POLITECNICO DI TORINO

Master of Science in Computer Science Engineering

Master's Thesis

Next Generation SOC: Automated Operations and Machine Learning in Cybersecurity

Advisor
Prof. Antonio Lioy

Candidate
Riccardo Gracis

Corporate Tutor
Mattia Mercuri

Academic Year: 2021 – 2022

Alla Mia Famiglia,
Non perfetta...
ma pur sempre Mia.

Acknowledgements

Writing a thesis is never an easy task, and I would like to thank all the people who have been close to me over the past five years of my university career at the Politecnico of Torino.

First of all, I would like to thank my father, my mother and my sister for being there for me and allowing me to fulfil my duties as a student. Thank you for all the financial sacrifices, support, encouragement and patience you have shown me. Thank you for standing by me for five years despite everything, and thank you for always putting me back on track, I owe it all to you.

I would also like to thank all my friends and in general, all the people who despite my absence have decided to stay without failing in their duties as good friends. They are a fundamental part of this journey, specifically, I would like to thank Nicolï¿½, Matteo and Emanuele who have been there from day one. Thanks to those who, even in their small way, has made all this possible.

For allowing me to do my thesis work in an extraordinary and dynamic place, I will be eternally grateful to Nais, which allowed me to get involved in the world of work and made possible an experience that will be invaluable for my future. Thanks to my SOC colleagues: Andrea, Gabriele, Fabio, Mario, Ivan, Gian Michele, Alessio and Manuel for having supported me on this incredible journey, passing on all your teachings and for allowing me to fit in perfectly among you. Thanks to Mattia, my corporate tutor, who with perseverance and dedication has accompanied me on this path, giving me the opportunity to be able to reach my total potential within both the academic and professional worlds. Thank you for all the ideas, insights and encouragement, for all the difficult moments; to you who showed me the utmost humanity and understanding despite my few hours of sleep and who despite everything you have always spurred me on to do better, I will be always grateful to you.

Thanks to Politecnico di Torino and all the professors who have passed on to me the passion and teachings to be able to achieve the craved goal. These five years have made me an engineer, but more importantly, they have made me a better person. Thanks to my university colleagues Emma, Gioele, Stefano, Jacopo and Nelly who have shared joys and challenges with me in the journey toward the Bachelor's and Master's degrees. My sincere thanks to my supervisor, Professor Antonio Lioy, for his dedication, perseverance and helpfulness in supporting me over these two years, transferring its dedication and cybersecurity knowledge to a thousand students every day. For supporting me over the past eight months: thank you for guiding me in writing the thesis and for passing it on to me his professionalism and life lessons, which no guide or book could ever have conveyed to me.

Finally, I would like to thank the most important person for this incredible achievement: Myself. Thank you, Riccardo, for always believing, for never giving up, and for always getting back up, even after painful falls. Thank you for keeping every promise on time against all odds. Thank you for the knowledge that you have gained along the way and that I hope will always take you to the top because you are capable of it, and you know what the sacrifices made yesterday are worth for a greater vision tomorrow.

I will remember these moments because they will give me the strength to continue progressing with conviction and confidence in what I am capable of doing, whatever the future holds.

Summary

In the last three decades, cybersecurity is becoming one of the most important aspects that organizations must handle to guarantee business continuity and proper internal processes. Several technological providers have been offered many solutions to deal with those operations, equipping next generation SOC with proper support to manage cyberattacks. Maintaining a consistent security posture nowadays is not an easy task for different reasons, and to support business's challenges, several standards were developed to ensure all the requirements. GDPR, PCI-DSS, ISO-27001 and papers proposed by NIST are becoming perfect alliances to satisfy all the requirements and in many cases avoiding expensive sanctions and loss in stopping production.

In my experience in Nais, I was able to hand on SOC's processes: both red and blue teams will be integrated to maintain a secure status globally. As required in many standards, the security stack must guarantee proactivity and reactivity. Proactive security concerns all the techniques involved in preventing security issues in the system monitoring and assessing the entire infrastructure. Reactivity, on the other hand, aims to restore a secure environment in case of issues as promoted in CERT and CSIRT. That kind of process not only is becoming important to maintain businesses, but also to fight against cyber-warfare and hacktivism.

To support detection operations as required previously, machine learning and pattern recognition cover a strategic innovation to support analysts. As discussed in that thesis many models such as Decision Tree, Isolation Tree or LSH represent an optimal the starting point to understand internal processes and flows to the extent that it is possible to build malware detectors or URL phishing detection system with 98% of accuracy. As an obvious fact, those models can be engineered in neural networks to optimize and tune the system, and for this reason many providers such as Microsoft, VMWare or IBM are investing many resources in refining those kinds of solutions. XDR, SOAR, SIEM and other strategic component can support analyst's tasks with a good level of precision, offering automation and integrations with other systems.

According to PDCA Model provided in the standard ISO-27001, there is also a governance aspect to keep into consideration. Concerning that, playbooks and operative planning can support Incident Response Team to deal with cyberattacks; otherwise, they can support red teaming remediation. Thanks to the experiences performed in Nais, it was possible to track all benefits provided by API's integrations with respect to classical human intervention both in terms of Incident Response and VAPT remediation. As required in SOC, all processes must be tracked, and thanks to TTPs, it was possible to implement all the processes starting from device isolation in cyberattacks until AD's user password restoration in Penetration Tests. From the moment that well-proven playbooks are easily deployable, it was easy to exploit Microsoft Graph and proprietary APIs, implementing a Python orchestrator able to send emails, block users, reset passwords or back up and update systems as well. The system developed offers all the benefits of speeding up all the operations with an aim to reduce the operator's intervention after a tuning period.

Thanks to that automation, a solution like that offers benefits also in SOC businesses, permitting the operators to focus only on relevant events and more services in parallel. Potentially automation can be also extended to the recovery phase if valid back-ups are protected that nowadays represents the perfect merge with cloud-based services. Speeding up operation means also increasing and making effective the frequency for assessment and hardening cycle exploiting shared Threat Intelligence provided by global CSIRT and CERT.

This permits the system to evolve adapting itself to incoming threats and, most important, evolving asymptotically to the state-of-art. Confident in progressing in applications and technological solutions can be considered the real essence in maintains a consistent security posture as required initially.

Contents

Chapter 1

Introduction

1.1 System Complexity

In the last three decades, there was an important growth in the Information Technology scenario concerning more and more production fields. Many processes that used to be managed by humans, nowadays have been replaced by automated solutions. Among them, there are also safety-critical processes, which require intrinsic security features such as confidentiality, authenticity, and availability. Simple examples can be bank transactions, car autopilot, health record management, shipment and sales management and so on... The need for cybersecurity operations and solutions is becoming more important day by day to the extent that there are more and more manufacturers that are proposing cybersecurity solutions and products to support other businesses against cyberattacks. Those solutions can vary starting from perimeter protection, audit systems, endpoint protection, and mail protection until more sophisticated ones such as User and Entity Behavioural Analytics exploit recent approaches like machine learning and so on.

In this artifact will be explained the necessity for automation and Machine Learning support to satisfy the security requirements according to network configuration, user working mode, and local compliance. Several factors increase the complexity of cybersecurity operations nowadays. Among them the most notable are:

- Entity Complexity and Variety: whether it is a computer or software, the higher the complexity of the system will be, the higher the cyber risk associated with it. More processes require more connections that in turn require more services, protocols, and open ports. All of this extends the attack surface to keep into account during the evaluations. All of those systems may support several platforms, OSs, and versions.
- Business Policy and User Operations: Nowadays company networks are composed of several components such as Servers, Desktop workstations, Laptop, Mobile and so on... Each of them again increases the attack surface. By the way, there are also topological configurations that may change due to user and employee behavior. First of first company devices can be in the external network without the same firewall controls and policies. Furthermore, the possibility to access cloud services (Office-365, Google, Azure AD) using a personal device instead of the company one.
- Local Compliance: according to local regulations, a business may need new technologies and infrastructure to be compliant. Some regulations such as GDPR require a strong strategy to satisfy resilient and business continuity constraints. To comply with that a backup system may be required. When a company decides to introduce this kind of system, the complexity inevitably increases again.

Compliance with regulations and standards proposed in cybersecurity fields represents certification of assurance regarding the security level adopted in the infrastructure and its related processes according to some specific and well-structured criteria.

1.2 Cyber Threats Impact and Standards

There can be different types of regulations such as GDPR and PCI-DSS which are mandatory to work with special types of data and, consecutively, providing services. While other certifications such as IASME and ISO27001 are voluntary ones to prove that security requirements are satisfied. Businesses have to take into account that being compliant with these kinds of certifications and regulations can have a huge impact avoiding being a victim of a cyberattack. In this period due to several factors such as digitization, the pandemic situation, and innovations, companies must be up-to-date about cyber threats to guarantee business continuity. A Lot of companies does not understand this fact until they are a victim of attacks. An attack or a data breach can be fatal for business activities.

For a small-medium enterprise located in Europe and subjected to GDPR, the following can be some effects resulting in a data breach:

- Business Stop: if an attack is aimed to stop and corrupt some business process.
- Recovery Effort: effort derived to restore the production system, reconfigure and tune the environment if a backup system is not available.
- Loss of Image: many customers can leave or dismiss their business operations in the company due to the bad reputation of the company itself after an attack or a data breach.
- Direct Loss of Profit: according to the previous point a loss of image and, consecutively, of costumers converge in a loss of profit.
- Indirect Loss of Profit: derived from the effort involved in system restoring, policies hardening, production stop, and maybe regulations sanctions in case there is no compliance with the standard required.

In the worst scenario, all of this loss of revenue may cause the complete bankruptcy of the company. Follow some statistics reports arguing the deadly impact of a data loss on the business process of a company. According to the University of Texas [1] after a catastrophic data loss, 94% of companies does not survive. Specifically, 43% never reopen, and the remaining 51% close within 2 years. These trends are also confirmed according to the National Archives & Records Administration in Washington which state that 93% of companies that lost their data for 10 days or more due to a disaster, filed for bankruptcy within one year.

A security breach is derived from several issues in configurations, monitoring, response, and backup strategy. Companies do not take recovery and backup strategy seriously so according to Contingency Planning and Strategic Research Corporation 96% of the business workstations are not being backed up [1]. Furthermore, 50% of all tape back-ups fail to restore according to Gartner [1]. This artefact aims to demonstrate the necessity for automation and machine verification to satisfy the requirements required by those standards to exploit all of their benefits.

Nowadays, there are a lot of standards able to provide the right approaches and evaluation schema to lead companies to the state of the art regarding the cybersecurity requirements implementation. As cited in the previous sections many standards are available according to different aspects such as:

- Locality: they are standards recognized in specific localities such as GDPR in Europe and HIPAA for medical record in the USA.
- Business Scope: such as PCI-DSS which is a certification required to deal with credit and bank records. This standard is used in e-commerce to implement secure transactions.
- Voluntary Certification: required by a business to certify its security standard. An example can be ISO27001.

All those classes of standards have some common aspects, but they will be analysed in a detailed way in the next subsections.

1.2.1 ISO/IEC27001

ISO/IEC-27001 is a standard regarding the requirements needed to govern an ISMS (Information Security Management System) [2].

This standard belongs to the ISO27000 family which contains 8 sub-standards. The most famous is the ISO27001 which is an evolution of the English BS7799:2 standard concerning information security matters. ISO acquires its guidelines in the ISO1799 standard while the second part the ISO27001, concerning the standard itself, was released in 2005.

Nowadays, ISO1799 is resumed in the ISO27002 guidelines which outline hundreds of potential controls and control mechanisms, which may be implemented, in theory, subject to the guidance provided within ISO27001. The standard not only concerns the logical protection of Information Systems but also treats physical aspects such as fire protection, unauthorized physical access to the server room, and so on.

In this artefact, it is posed attention to the logical and technical aspects, then the requirements provided by the standard will be the reason why modern systems require automated logic to satisfy those requirements. ISO-27001 nowadays follows the guidelines defined in the update released in 2017 which is based on one of the major version released in 2013.

According to the standards, data are considered an asset for all intents and purposes, to the extent that their loss or compromises can represent a serious issue for the business. Sometimes a loss can lead to drastic scenarios as explained in the previous section. The principal aim of the standard is to protect data to provide and satisfy the Availability, Integrity and Confidentiality requirements. ISO standards are often related among them; for this reason ISO27001 is strictly related to ISO9001 for Quality Management and ISO14000 in order to treat Environmental Management.

Furthermore, ISO27001 is based on Risk Analysis processing and evaluation recursively to promote a continuous improvement, converging to the state-of-the-art. This recursive approach is defined as the PDCA model, and it is based on 4 steps PLAN, DO, CHECK, ACT which will be described in the next sections.

PDCA Model

PDCA Model [3] is widely known nowadays since it allows us to analyse the current state, evaluating the issues that must be resolved. Then, recursively, the process is performed again resolving issues iteration after iteration if needed.

This approach converges to the state-the-art, and it permits the system to evolve consistently to new requirements and threats. The main phases are:

- PLAN: establish aims and final state according to the improvement needed by the current state. The current state is evaluated during the risk analysis according to the probability and severity of a threat that may impact the system. It evaluates also a small set of the entity on which perform the operations also called the Piloting Phase.
- DO: it consists of effectively performing the mitigations and troubleshooting on the defined small set. There is an important phase of report and data analysis in this phase to evaluate the process in the check and act phase.
- CHECK: revision of signs of progress, and based on the analysis performed on the DO results, it considers the consequence of involved actions.
- ACT: can be considered the implementation of the strategies discussed previously, including the improvements discussed in the CHECK phase. Again, are evaluated possible divergences between the expected results and the effective ones.

The PDCA cycle is massively used in ISO standards such as ISO9001 which poses some fundamental aspects even in the ISO27001 standard. Following is a reported image concerning the quality improvement through PDCA iterations.

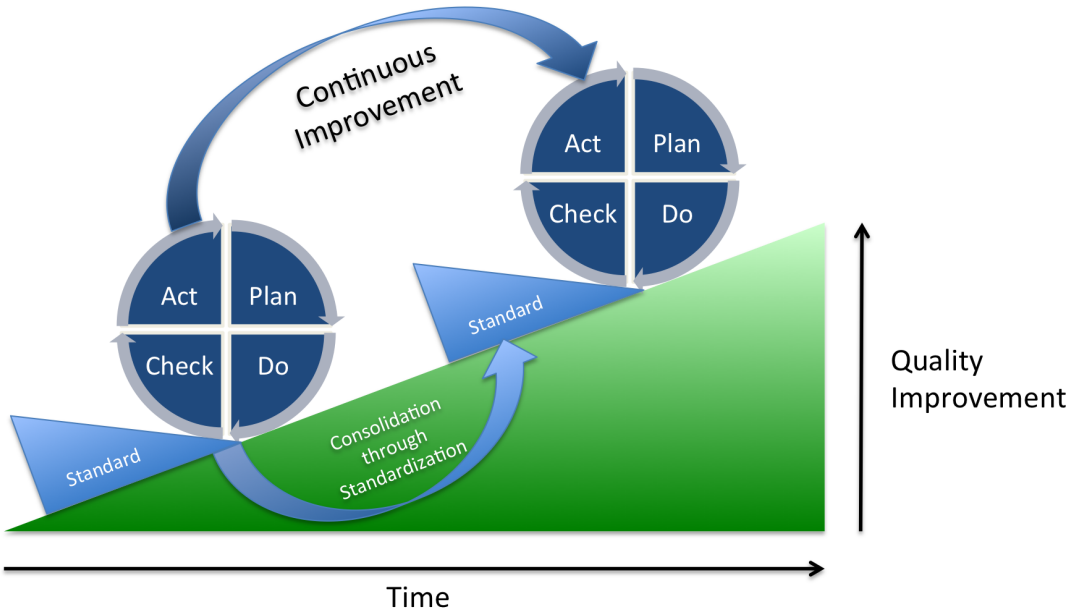

Figure 1.1. PDCA Quality Improvement

Operations and Domain

The standards provide 4 main phases strictly related to risk analysis and management:

- Planning
- Implementation
- Monitoring
- Maintenance and Improvement

As it states those phases are inherited from the PDCA approach, specifically the planning phase can be resumed in the following sub-phases:

- Risk Identification: know my system and how it works according to internal processes and approaches.
- Analysis and Evaluation: regarding the risks previously identified
- Evaluation of objectives: definition of activities to control and manage the risks
- Definition of residual risks: define a final state where the related risks can be acceptable
- SoA (Statement of Applicability): definition of the link between risk assessment and risk treatment in the enterprise.

Nowadays there are specific controls to evaluate compliance with the ISO27001:2013 standard [4]. There are 114 controls, divided into 14 categories, outlined in the Annex's checklists. Follows a detailed list of the controls:

- Annex A.5 - Information Security Policies (2 Controls): ensure that all policies regarding the information security is written according to the organization's requirements.

- Annex A.6 - Organization of Information Security (7 Controls): definition of a management framework and assignment of information security roles for how the controls will be implemented. In addition, it designs the security guidelines when employees access, process, and modify information while working out of the office.
- Annex A.7 - Human Resources Security (6 Controls): ensure that all parties understand their requirements during the entire term of employment. The latter involves conducting interviews and investigations to know the user's background, adhering to security policies , and training or implementing a formal disciplinary process to protect and conserve the business interests.
- Annex A.8 - Asset Management (10 Controls): Ensure that an organization identifies, protect, and classify all asset and information treated by the asset itself. To do that must be defined the acceptable use of the assets employing a classification scheme outlining all the procedures to handle and implement a secure processing and storage of data according to their nature.
- Annex A.9 - Access Controls (14 Controls): Limit and prevent unauthorized access to information through access control policies based on rights, privileges, authentication methods strength, and many other criteria to protect any programs with override capabilities on data. Furthermore, also define the responsibility for authentication information protection such as PIN, Password, OTP , and a combination of them as in the MFA strategies.
- Annex A.10 - Cryptography (2 Controls): Ensure encryption and key management is used to maintain Integrity (Authenticity), Confidentiality and Availability for sensitive data. Set the key policies related to key strength and expiration period.
- Annex A.11 - Physical and Environmental Security (15 Controls): As cited in the previous section this category concerns the prevention to compromise assets through loss, physical damage, or theft. All these operations require the definition of a physical security perimeter and protection of the equipment inside the protected areas as well as external ones.
- Annex A.12 - Operational Security (14 Controls): ensure information integrity during the processing phase. Latter involves the protection of the facilities and systems which process data from malware and loss of data. All those operations are tracked through a log system which requires intrinsic protection itself which allows audit activity and forensics analysis. Operational Security involves also a documentation phase where all the processes performed in the organization should be described to differentiate common operations from anomalous ones. Again, modification to process operation must be correctly updated in the documentation.
- Annex A.13 - Communications Security (7 Controls): Monitor Internal and external communications to identify and prevent data exfiltration using transfer policy. According to other controls such as Cryptography (Annex A.10), communications should be encrypted to satisfy the confidentiality requirements, some strategies such as TLS inspection must be applied to understand which data are transferred in encrypted communication.
- Annex A.14 - System Acquisition, Development, and Maintenance (13 Controls): ensure that information security management is established during the entire system lifecycle, especially during the update and the introduction of new systems. Furthermore, it ensures that the development process follows the best approaches concerning the security of data treated in the processes.
- Annex A.15 - Supplier Relationship (5 Controls): concerns all the strategies adopted to ensure that any valuable assets accessible by the third party remain protected according to the formal agreement which defines all the security policies adopted.
- Annex A.16 - Information Security Incident Management (7 Controls): based on the Incident response strategy to reactively manage an incident to limit and contain and eradicate the threats avoiding data exfiltration or compromise. This phase is managed by Incident Response Team, and there are specific frameworks which describe all the operations needed

such as the NIST Cybersecurity Framework or the guidelines explained in the NIST SP 800–61 (Rev. 2) papers that will be explained later. This operation requires also a reporting system to know and patch any known vulnerability according to Vulnerability Assessment and Penetration testing techniques.

- Annex A.17 - Information Security Aspects of Business Continuity Management (4 Controls): ensure resiliency an nd Business Continuity strategy in case of critical events or compromise. Those aspects concerned specifically are derived basically from the availability constraints and can be implemented employing back-up strategies, systems, verification of back-ups, and efficient recovery systems.
- Annex A.18- Compliance (8 Controls): Avoid security breaches and ensure those information security aspects are carried out in a compliant way according to local, legal, or business area regulations.

In the next image is reported how PDCA and Annex basics are related:

Figure 1.2. PDCA and Annex relationship

As will be explained in the next sessions, ISO27001 can cover the main aspects reported also in other regulations such as GDPR and PCI-DSS. By the way, differences and further requirements required in other standards will be further discussed.

1.2.2 GDPR

Another important regulation followed by all European countries is the GDPR (General Data Protection Regulation) [5] and it is coordinated locally by supervisory authorities and designed at the national level. It poses the requirements to treat sensitive data defined in the Art. 9. Differently from ISO27001, it is a regulation composed of recitals, defined in legal terms, but further technical requirements are provided to implement them correctly.

It also defines the operative phases to handle a data breach as explained in the Art. 33. The 99 articles defined in the GDPR can be divided into 11 Chapters. The following section will be presented all the chapters highlighting the most important article.

- Chapter 1 - General provisions (Art. 1–4): defines the material, territorial scope, and the definitions used in the regulation.

- Chapter 2 - Principles (Art. 5–11): defines the principles of data processing and defines different types of data. As cited before Art. 9 defines the sensitive data and Art. 10 concerning the processing of criminal convictions and offences data.
- Chapter 3 - Rights of the Data Subject (Art. 12–23): defines the rights involved in all the data collection, processing, and storage phase. It guarantees the transparency of data collection such as the motivation why those specific data are needed. Then inform the user about the rectification and modification of the data. Furthermore, it describes the period within the data storage and the procedure to erase the data after that period. They have described the operation performed in automatic user profiling performed by those data.
- Chapter 4 - Controller and Processor (Art. 24–43): It defines the responsibilities of the controller and description of the processing mechanism as well as audit and record of operations performed on data. Furthermore, this section also defines the security of personal data through intrinsic processing security and notification according to Art. 33. There is also the designation of a DPO (Data Protection Officer) who oversees all the tasks according to Art. 39. Finally, there is a list of code conduct and certifications aspects to guarantee GDPR compliance according to Art. 42 and Art. 43.
- Chapter 5 - Transfers of personal data to third countries or international organizations (Art. 44–50): it defines the constraints to follow in case of sharing personal data in the country inside the EU, promoting international cooperation. Furthermore, it poses the bases to share data in other continents not covered by GDPR legislation. This is a big issue in the cloud computing era, the reason why cloud providers are moving their data centres also in Europe in order to be compliant with GDPR.
- Chapter 6 - Independent Supervisory Authority (Art. 51–59): defines the guidelines related to independence status, competence, and tasks required by Supervisory Authority.
- Chapter 7 - Cooperation and consistency (Art. 60–76): ensure cooperation and consistency among supervisory authorities in order to keep aligned. In this evaluation is involved also in the European data protection board itself.
- Chapter 8 - Remedies, liability, and penalties (Art. 77–84): defines the juridical operation against the controller, processor, and supervisory authority. Furthermore, define the penalties in case of a data breach according to the measures taken to be compliant with GDPR.
- Chapter 9 - Provisions relating to specific processing situations (Art. 85–91): defines processing to information, official documents, and obligations of secrecy.
- Chapter 10 - Delegated acts and implementing acts (Art. 92–93): the exercise of delegations and committee procedure.
- Chapter 11 - Final Provisions (Art. 94–99): revisions and relations with other agreements.

According to the requirements explained lot of these high-level definitions can match ISO27001's [4] requirements and consecutively can be implemented thanks to Annex specifications. For example, Chapter 4 of GDPR can be covered by many definitions of Annex A.12 concerning operational security. Highlighting differences and similarities among standards is not the goal of this artefact, reason why this consideration is left to the readers.

1.2.3 PCI-DSS

Payment Card Industry Data Security Standard [6] is a specific defined to deal with credit cards information, powered by Security Standard Council. It is a widely used standard to ensure data protection and fraud prevention in payment systems and e-commerce. The standard is ensured through validation according to the annual number of transactions of the applicant. There can be 3 different cases:

- Self-Assessment Questionnaire (SAQ): Offers several types of questionnaires starting from 22 questions up to 329. It can be also associated with an Attestation of Compliance to highlight new requirements. It is applied for low volumes of transactions.
- Qualified Security Assessor (QSA): performed by a certified third party after an examination of the system. It is applied to medium volume.
- Internal Security Assessor (ISA): Global test and Assessment of the system, it involves issuing a Report on Compliance (ROC)

The standard provides the guidelines and requirements needed to ensure compliance. As will be reported below, again there are lots of similarities among the presented standards. There are a total of 12 points summarized in 6 macro areas which will be related to the Annex requirements [4, 6]:

- Development and management of the security system and network (concerns A.12 and A.13):
  - 1. Install firewalls and manage their configurations
  - 2. Do not use default password in information systems
- Card Holder Data protection (concerns A.12, A.13, and A.14):
  - 3. Data protection on physical support
  - 4. Data protection within communications
- Vulnerability Assessment and Management (concerns A.14):
  - 5. Protect endpoints and keep the system up to date
  - 6. Develop and handle a secure system
- Access Control (concerns A.11, A.12, and A.13):
  - 7. Limiting access to cardholder's information
  - 8. Track and authenticate access to a system component
  - 9. Physical access
- Monitoring and Testing (concerns A.6, A.12, A.13, A.14, and A.18):
  - 10. Monitor and perform regular audit
  - 11. Test and stress system to highlight weaknesses
- Adopting and Handling Information Security Policies (concerns A.5):
  - 12. Adopting valid and well-structured policies to guarantee data protection.

1.3 Security Operation Centre - SOC

In today's society, cybersecurity aspects are becoming more significant day by day so companies have to protect their assets against cybercriminals and cyberattacks, as well as be compliant with the regulations described before.

The most important point is that nearly always those companies cannot build up internal cybersecurity department or, even worst, does not care about that kind of risk. For this reason, in the last decade, many cybersecurity enterprises were born in order to support the third party in cybersecurity assurance, prevention, and consulting to satisfy the security requirements provided by the best practices. Many of the most essential services provided by cybersecurity consulting companies are ones related to the Security Operation Center (SOC) operations are divided into two big branches: Red Team and Blue Team. As explained in the next sections red teams is referring to the offensive security part while the blue team to the defensive part.

Since security operations provided by SOC are becoming a core process nowadays business for an organization, there are two possible solutions for a company. The first is the one related to a CAPEX (Capital Expenditure) business model where SOC is managed internally to the organization, and it requires a significant initial investment. The alternative can be supported by an external SOC adopting an OPEX (Operation Expenditure) solution where costs are exclusively related to the service offered by the external SOC.

1.3.1 Blue Team

As stated in the presentation of this chapter, SOC works into two big scenarios. The first is the Blue Team whose aim is to prevent malicious tasks through monitoring and analysis operations. This is also known as the defensive approach because the idea is to prevent attacks: monitoring logs, ingoing and outgoing network traffic searching for suspicious activity, unauthorized access, malware execution, etc... In that way, all the logs produced by network devices such as firewalls, switches, routers, NAS as well as network traffic will feed that system. Correlation among all of those data will produce a unified view of what is happening in the system at a given time. Based on this assumption the operative chain of a classical Blue Team is resumed in Collect, Detection, and Response operations, and they will be explained in the next sections.

1.3.2 Red Team

The other face of the SOC is related to the Red Team which aim is to investigate the security level of a system by means of vulnerability assessment and penetration testing. The first is performed through automatic assistants, which can detect vulnerabilities according to some assets, manufacturers, versions, and data sources (such as MITRE CVE); based on those shreds of evidence and on a general score given to those vulnerabilities, CVSS Scoring System can evaluate the security level of assets. Starting from those considerations assessors can evaluate consecutively the impact in the case related to assets.

By the way, not all vulnerabilities have a known exploit and for this reason, it is the purpose of the penetration test to highlight the vulnerabilities that may be exploited to perform malicious activities on the system and its impact. In conclusion, those two techniques are used to simulate an attack to highlight and mitigate possible flaws in organizations.

1.3.3 Purple Team

As it is stated in the previous sections there are two macro areas: Red and Blue team which aims are test and monitor a system, implementing proactive and reactive operations as well. According to the requirements introduced in the previous section, a secure system requires audits came from operations provided by the blue team and red team.

Nowadays, the Purple Team ideology was born in order to protect the system in a compliant way. As stated before, this approach stresses a system, looking for weaknesses, and finally addressing issues and vulnerabilities as proposed in the blue team operation. Purple teaming operations can evaluate Incident Response and Recovery operations, training and make aware Blue team operators on how an attack works based on well know patterns. Some of them will be analyzed in the MITRE ATT&CK Matrix later. Similarly, can be evaluated the efficiency to respond to an attack and recover the system in case of a successful one. This is, for example, a requirement presented in Annex A.17 according to business continuity and resilience.

Assuming that a good Incident Response Team will block and eradicate threats in an efficient way, a Purple Team operator can exploit some specific frameworks in order to handle incidents effectively. The next Venn diagram figures out the main activity and services offered by a Purple Team. To deal with advanced threats and compliance standards is not possible to speak about Red Team and Blue Team as separated instances. Blue Team helps Red Team in terms of data analytics and threat intelligence in order to recreate attacks and patterns that in turn, stress the Blue Team detection and response aspects. For this reason nowadays many SOCs including the

one present in Nais are now migrating to the new definition of a purple team, where proactivity and reactivity are considered two shapes of a bigger scope.

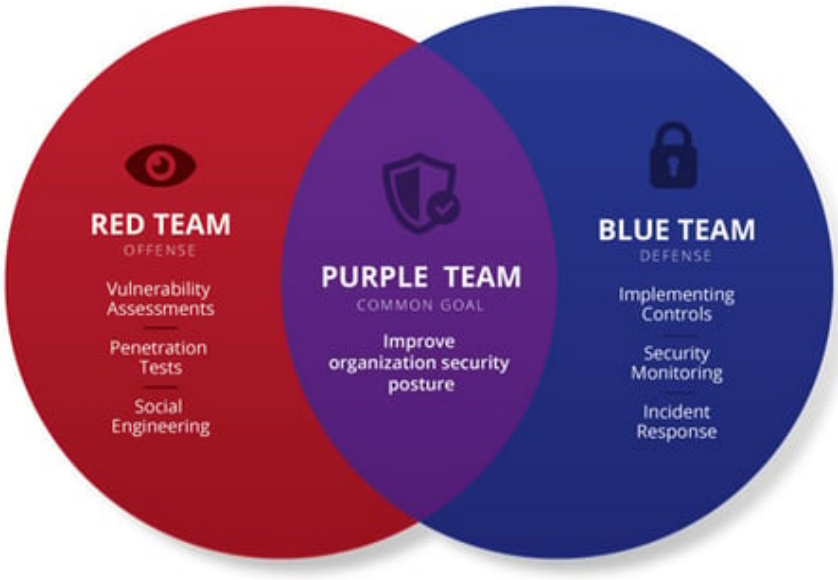

Figure 1.3. Purple Team Resuming Venn Diagram

As explained in the previous section many frameworks can support an organization to implement an awareness and compliant security systems according to their business scope and resources. At the same time, there are companies able to provide security services and support such as the one proposed by a SOC which are basically composed of three macro services:

- Read Teaming: involved in perimeter assessment aimed to discover flaws and misconfiguration
- Blue Teaming: Monitoring and Incident response
- Management: Compliance, Recovery, and Business Continuity

Operations performed in Purple Team are basically the ones explained before aimed to maximize the cooperation among those areas. Cooperation is provided by the Blue Team aspect such as monitoring and detecting security events according to standards expressed in ISO27001 (Annex A.12 - Operational Security) and pattern recognition provided by MITRE ATT&CK for behavioural analysis and CVE analysis, then Incident Response and Recovery can be executed according to Annex A.16 and A.17.

The ones concerning Red Teaming are strictly related to the testing and assessment operation belonging to Red Team the operation such as penetration tests and vulnerabilities assessment. Regarding one of the major problems expressed at the outset system complexity made coverage and scope of the assessment difficult to be issued by a single tester. In addition, according to MITRE ATT&CK framework, techniques engaged nowadays by attackers is the one concerning lateral movement. Those types of problems will be explained later, the basic idea is the necessity of automation also in proactive operations in order to support the operator.

There are also time issues to take into account to implement all execution phases provided by the PDCA model. This is becoming a major and challenging constraint to satisfy and for this reason, automatic and technological tools and architecture are implemented to support an operation like that. Manufacturers are committed to implementing compliant tools intrinsically protected and secure once again adhering to standard as ISO27001, GDPR etc...

Nowadays there is also the possibility to exchange knowledge among organizations creating a shared Threat Intelligence System. There are located many local and global regulative organizations able to provide this kind of information to track coherently a malicious entity on the internet such as IPs, digests, or URLs. This information is used in order to prevent specific situations or attacks according to data analysis over business sectors cyber threats actors. Once again MITRE Matrix is a good starting point because it offers also an excellent repository concerning intelligence related to cyber-warfare groups. Again according to those considerations, the reader can be aware of the relationship among those operations which belong to different macro areas. In the next section will be presented the three approaches and technologies involved.

1.4 Proactive Operations

In this section will be presented all the operative technologies involved in the monitoring and incident response scenario such as SIEM and their evolution; SOARs are widely used in extensive organizations to monitor and response to some critical events in a distributed environment, then move through the device security strategies expiation.

The main operation that can be considered proactive is the related to penetration testing and vulnerability assessment, which permits to a system to evaluate its flaws and components potentially vulnerable in case of attacks.

As introduced in the previous section, Proactive operations concern also system monitoring and analysis in order to detect security events Among them the most important are the ones related to some well-know attack patterns or the ones that may be related to some dangerous situations. As stated in the NIST Glossary Proactive Cyber Defense is a continuous process to manage and harden devices and networks according to known best practices. Implicitly, harden operations are derived by threat hunting evidences or identification of misconfigurations that may increase the cyber risk related to the system. The other concerning the management of devices and networks regards the Incident Response Team and Detection engine.

In the following section will be presented in a detailed way all the promises done above according to the local standards and compliance.

1.4.1 VA-PT and Compliance

In this section will be briefly introduced Vulnerability Assessment and Penetration Test, and their compliance implication in the ISO27001 Mandatory Clauses. They are 10 clauses describing how ISO27001 can support the integration of an ISMS, focusing the attention on the needed concerning Red Team operations.

ISO27001 - Mandatory Clause in ISMS for VA-PT

According to ISO27001 management system clauses, penetration test and vulnerability assessment are located in section 4 related to improvements, and consecutively proactive operation. In that case according to the ISMS [4] are examined internal and external issues that may be relevant to the business and to the achievement of the objectives of the ISMS, including confirmed interesting parties and scopes (ISMS mandatory Clauses - point 4). This assumption poses the baseline for penetration testing in order to stress the system and highlights problems (in that case vulnerabilities).

Then The same Management Model provides definition order to address those type of issues already defined in the previous phase. Important points are the ISMS mandatory clauses 6 which defines how the organization creates actions to address risks (evaluated in VA-PT), including setting information security objectives. In addition, point 8 requires planning and definition how processes will be executed, and most important, production of detailed documentation in order to describe those processes.

Finally, the last mitigation regards the actuation of the mitigations according point 8 concerning the form of the documentation and starting from this point, how organization will monitor, measure and evaluate the ISMS according to point 9. The final phase is related to point 10 which requires implementing a corrective and continual improvement, referencing PDCA model for Improvement of quality.

Vulnerability Assessment

In this phase the Red Team member used a vulnerability scanner (such as Nessus or Nmap) in order to perform a scan on some machines. Those tools generally try to enumerate open ports and possible services running on it. Furthermore, those tools are able to detect OSs and the relative version evaluating the response in the TCP connection by means of some flags combinations. An important aspect is that not each vulnerability is linked to a known exploit, reason why typically the assessor focuses on the vulnerabilities that can be exploited by means of a known exploit so that the companies can update an adequate fixing.

Manual Penetration Test

It is a simulation of attack where the pen tester tries to obtain unauthorized access to the target assets as he impersonates a real attacker according to the scope designed by the requester. The requester third party can also decide if the test is performed following a black box approach where the pen tester does not know anything about internal scenario (such as network topologies, credentials, application source code etc.) or white box where more internal information about the target are known to the pen tester. Frequently also a hybrid approach known as grey box is used in order to give some basic knowledge to the pen tester about the target avoiding time consumption in information gathering and reconnaissance.

During a penetration test, the tester tries to conduce malicious action on the target in order to evaluate the maximum impact on the system in terms of business value. In order to understand better that concept take into consideration a scenario within a double extortion ransomware that has successfully encrypted some sensitive data; even if the ransomware-hit company are equipped with back-up system in order to restore data, avoiding ransom bill. A malicious actor can disclose anyway sensible data in order to hit the company under public image and reputation point of view. This may lead to a loss of client trust and consecutively in loss of business. This is the reason behind penetration testing, in that case the tester always evaluates the worst scenario to make evaluation about the risks and business impact. This basic system can be correlated to the difficulty of implementations to describe a better evaluation of security risks.

Manual Penetration Test Limitations'

Nowadays, customers require technological solution in order to overcome the basics limitations concerning classic penetration testing and issues related to duration, scopes and costs. Behind a penetration test there is a human being which manually assess the system starting from a set of vulnerability, trying to find out the ones which represent potentially the greater impact for the system.

Assuming an old approach, a penetration test requires a set of resources such as time, knowledge and most important its cost. The old approach requires to pay a high expertise operator to stress the network for a long period. Furthermore, penetration tests can evolve in disservices which can lead to loss of profit or work out of business-hour by the operator which means higher cost. In each case it represents a relevant costs for the company. After the penetration test occurs operators has to produce reports and useful hints in order to prioritize mitigations and most important implement and validate them. This is translated into huge costs and duration to perform a set of activities which should be performed periodically. Again, those observations are translated into negligence of company and loss of interests in benefit and procedure involved in Red Teaming operations.

There are also others limitation regarding the system complexity in penetration testing that is Lateral movement: One of the most dangerous operations is to use the compromised host to perform network discovery and lateral movements. In this case the attacker exploit the access gained in the previous phase to effectively execute operations of various type. For this reason for a human operator is impossible to cover all the possible spaces generated by lateral movement for multiple reasons, at first for knowledge and time consumed. As we will discuss later, this reason becomes the principal reason why nowadays classical penetration testing is becoming unfeasible to cover big networks and organizations.

NIST RMF

How to handle risk in front of Lateral Movements?

NIST Computer Security Resource Center [7] again has the solution, it is a simplified model extracted from its SP-800-39 [8] Controls to implement a Risk Management Framework according to controls defined in 7 points:

- Prepare: Identify core activities for the organization, define risk tolerance and risk monitoring processes.
- Categorize: Evaluate the impact according to issues in Availability, Integrity and Confidentiality for resources identified in the Prepare phase. This results in a detailed document describing what previously reported.
- Select: Identify controls and security objectives in order to handle the risk evaluated before for each component.
- Implement: Implement controls previously defined
- Assess: Then assess those controls in order to highlight issues and validating its effectiveness.
- Authorize: Provides responsibility (similarity with GDPR DPO Role), and asking for system revision and integrations.
- Monitor: Proceed with monitoring of what has been build.

Note that many times Categorize and Prepare statement are grouped in categorize single phase as it is reported in image 1.4. In the same image 1.4 are reported standards where NIST RMF takes its principal baselines.

According to this framework Lateral Movement has to be intended has cumulative forms of compromised entity by an attacker, for this reason with attack complexity and resources lateral movement are becoming a real problem in risk management. They represent one of the most important problems for a company and Cyber-security operators. Ransomware can disrupt Availability on infected machine, but basically lateral movement can involve disruption of Integrity, Availability and Confidentiality for each entity (user included) in the network, with consequent explosion of the risk.

1.4.2 VA-PT Automations

As was stated in the previous discussions, the main problem related to classic approach to penetration testing concern its limitations in terms of lateral movement exhaustive coverage, knowledge and durations. In addition to those limits there is also the problem related to disservices and production stop in case of error and most important, economic aspects which are out of the scope of this artifact. For the moment the section focuses its attention on the other problems and try to figure out a solution based on automations.

Figure 1.4. NIST Risk Management Framework - Baseline

Operations and Models

The model that will be presented shows the main phase in order to conduct an automated penetration test exploiting all the computational power of the resource. The process starts with a pre-assessment in order to evaluate what is the real requirements basing on a target identification; to do that are useful interview to understand critical assets, also based on Threat Intelligence Information and scope. In the second step is discussed and designed the successful criteria based on the attack results. Those metrics are strictly related to the attack itself for example number of hosts compromised, weighted average respect to vulnerability CVSS score and frequency, cracked password and many others. In this designing phase can be also identified successful criteria to assess the detection and response system in case of attack. Finally, the attack is simulated considering various aspects such as business continuity in order to not compromise a host running on production and evaluate feasible time intervals to perform the assessment.

Machine Learning Driven Weaknesses Scan

As stated in the previous introduction vulnerability scanners are adopted to highlight possible vulnerability in the software. The major vulnerability database known today is the MITRE

CVE [9]. As stated in the Detection chapter this kind of solutions are fragile from the moment that they work on static information as OS, SW's version and others.

Behind those kinds of record there are researchers that spent their time to assess and test components such as libraries, software and more generally code in order to find a possible bug or defeat that can be exploited or can degenerate into a vulnerability. The main taxonomy behind the CVEs are the CWE [10] which instead works directly on well know dangerous code such as BufferOverflow, Out-Of-Bound Read, use before free with pointers and so on. Those kinds of weaknesses are particularly important because are provided with well known functions and code patterns that hide the real vulnerability. Starting from those assumptions, it is simple to understand why instead of a static Vulnerability Scanner can be used a h learning based approach in order to detect flaws directly in the code. In a similar way code scanning using machine learning is more fault-tolerant to pattern variability and can identify with an higher accuracy rate some weak code snippet. This solution is already widely used, for example CodeQL promoted by GitHub in its code review operation involves those kinds of technologies to build a feature extractors to detect fault code. The process behind is similar to a NGrams feature extraction for malware detection that will be presented in the blue team's operations, in that case instead to search for common pattern in malicious code, model will search for common pattern in known software which suffer for that specific vulnerability. Starting from vulnerability the model knows also the Weakness and that is it.

In order to get a good training set is not difficult for GitHub which has access to millions of repository, and among them there will be for sure some vulnerable snippets. Code QL and many other framework exploit some well-known queries in order to detect weaknesses, this can be considered as label 1 while other code which does not are vulnerable according to those queries are considered 0. This can be a good metrics to know how data works. Now machine learning can feature those elements focusing on their basics feature and discriminates among weak pattern and common ones. When ML-generated alerts are enabled by repository owners, CodeQL computes the source code features for the code snippets in that codebase and feeds them into the classifier model. The framework gets back the probability that a given code snippet represents a vulnerability, and uses this probability to surface likely new alerts. According to GitHub a based system like the one presented is able to provide a better performance respect classic and static vulnerability scanners, but pay attention, as stated many times in this artefact this means to merge the two approaches in order to have the classical high accuracy due to well-known flows of vulnerability scanner detection as well more general models to detect vulnerabilities.

Pentera - Automated Penetration Test Framework

Pentera [11] nowadays represent one of the most influential and technological solution in order to assess the network independently on its dimension. Pentera has many peculiarities and among them stands out the possibility to create local file copies without impacting the production or access production files in any way. This according to the evaluation made before is one of the most critical point which cannot be satisfied by a manual penetration test without slow-down again the entire process. Product can be configured in order to exploit computational power provided by other nodes in order to crack hashed passwords. In this phase, it can be decided to involve external nodes with GPU based cluster in order to provide computational resources to HashCat password Cracking tools. Password cracker may be also supported with a predefined dictionary, as the well known rock-you dictionary as well as other basic words related the organization and so on.

The primary metrics designed are the following:

- Number of Password Cracked: this is one of the most important aspects. Many passwords cracked are synonyms of weak password policy that an attacker may exploit, this aspect should be also related to the available computational resources, most important, when it can be adopted by an attacker.
- Number of Total Exploit: high exploit available can open up new attack chain and operations.

- Number of Administrative User Compromised: This is another important aspects because with high level privileges attacks can become powerful.
- Number of Vulnerabilities and related Score: System with lots of high vulnerability does not survive to an attack. Furthermore, can be an evidence of bad patch management.
- Achievement: stats about partial and totally compromised hosts, users and devices.

Pentera offers also an important set of configurations in order to deal with assessment shapes and scopes: Among them stands out:

- Stealthiness: which select the noise of enumeration and port scanning. Some times can be useful to perform stealth enumeration to assess also the detection system integrated in the target.
- Approval Action: each time that Pentera understand that a possible vulnerability can be exploited, ask operator for taking action: run or cancel the exploitation.
- Distributed Password Cracking: when possible tools can be integrated with external GPUs. Useful to crack NTML tokens.
- Bruteforce on major services (Not recommended for sneaky assessment): such as SSH, SQL, FTP etc...
- Allow Worst of Both Worlds Exploit: authentication as SYSTEM Authority on Windows Active Directory environment using a combination of NTLM relaying and Kerberos delegation techniques to achieve this.
- DHCP MITM Attack: The DHCP MITM attack enables the platform to receive authentication requests, to crack NTLM hashes and to perform relay attacks when LLMNR, mDNS or NetBIOS name resolution is disabled.

Pentera executes the assessment based on configuration and finally is able to provide a detailed report with result concerning all the metrics reported. Among the reports there is a useful one reported all the exploitable vulnerabilities. The real power of Pentera is that based on the action approved is able to generate a prioritization related to the vulnerabilities present in the system.

Starting from this report a patch-man can implement mitigations based on severity and resolution surface. Those are the major benefit which are not evaluable in the same time by a human and definitely it is the reason why manual penetration testing is overcome.

Pro and Cons

In the previous section was analysed the main features offered by different approach in penetration testing. As we can see there is two main aspects that are visibility and human interaction in penetration testing which permits to have a global visibility of the infrastructure that only in few cases is available to an automated solution. This affirmation regards the fact that a user or a penetration tester, especially in white box, has a view of assets as well an automated solution, but latter does not have any idea about processes and most important assets. This is the first observation that can be easily disrupted by the fact that to emulate a real attack, penetration testing follows black box approaches without any other information except for the ones gathered by means of OSINT operations and Reconnaissance.

In other hand, automated solution offers many benefits, among them are remembered the most important:

- Efficiency
- Duration
- Scheduling

- Costs
- **Prioritize and Automize Patching**

The last point is the most important also under a compliance point of view. According to NISTIR 8011 [12] which presents the basic automations support in Information Security Continuous Monitoring (ISCM) systems, is possible to work directly on NIST Cybersecurity Framework presented in NIST SP-800-37 [8, 13] in order to automatize two specific phases among the six presented. Those phases are the assess by means of Automated Penetration Testing and consecutively and Monitoring by means of detection System and Blue Teaming.

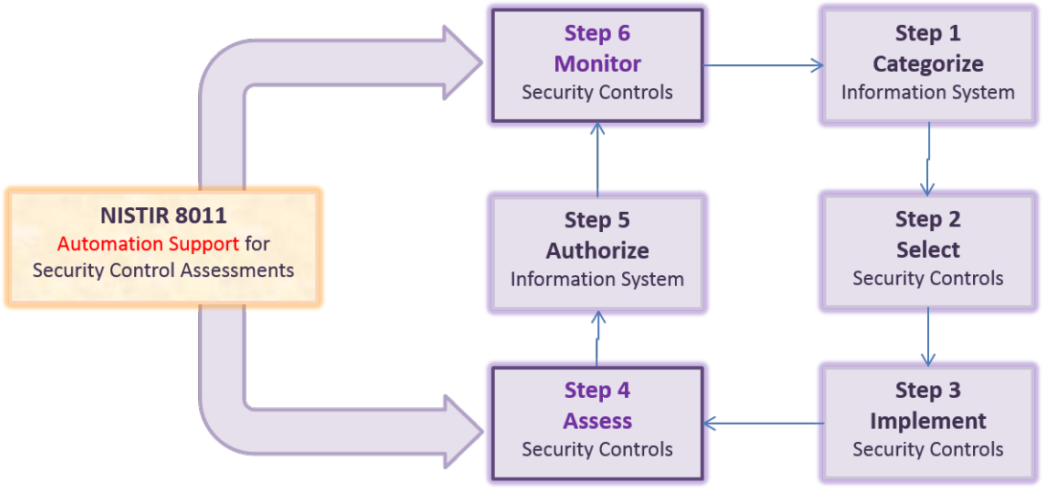

Figure 1.5. NIST Risk Management Framework - NISTIR8011 Automation Support

For effective automated assessment, testable defect checks are defined, and they bridge the determination statements to the broader security capabilities to be achieved and to the NIST SP-800-53 [14] security control items themselves. ISCM Security Capabilities are defined as set of security Controls Item (defined in NIST SP-800-53) which purpose is to collaborate to block some specific attack phase.

As stated before Automated Penetration Testing assess the lateral movement path that can be destructive for a company. What is stated can be represented with a matrix of security capabilities and attack phase.

As the image shows Attack phase can be interpreted as a general form of MITRE ATT&CK framework and it is not detailed on techniques but also on a general attack kill-chain. The vulnerability management can suppress the attack phase defined by NISTIR Notation as *Expand control - escalate or propagate* that cover also lateral movements. Again according to the matrix Vulnerability Management is one of the Managed Operations and effectiveness related to its security controls can be assessed, according to NIST SP-800-53A [14] in 3 different way Examine, Interviews or Test. The easier one is the Test which is defined as *"The process of exercising one or more assessment objects under specified conditions to compare actual with expected behaviour."* Again it is a basic assessment which aim is to compare a desired state with the actual one as described in other standards such as ISO27001.

According to those definitions, Vulnerability Assessment and Automated Penetration Test become indispensable Managed Operations to Test security controls defined in NIST-800-53A. As already presented in this artefact, automations in red teaming operations as well as mitigations of vulnerability detected represents a way to implement a security validation that can be performed according a PDCA model in order to improve the global states of the system itself. Iteration by iteration the system can be potentially convergent to the state-of-art.

1.4.3 Blue Team Operations

In this section will be presented the main features related to Blue Team operations starting from the idea of monitoring system to proactively detect and the response to cyber-attacks, until the post-incident activity and recovery.

Pay attention that Blue Teams operations are divided into Proactive and Reactive concerning Incident Response Team which shared the main activities stack with the classic monitoring operations. To support the considerations and operations explained will be also presented the main standard and framework involved in this scenario.

Collect

The classic Blue Team focuses its attention on events, because starting from those, team is able to identify each kind of activity and will be analyst's task understand if it is a normal network traffic or attackers' operations. Furthermore, all the information collected can be useful to guide the environmental threat intelligence in order to know specifically which logs and events are involved in specific attacks, also known as attack pattern. The source of that collector are the entire relevant devices, able to generate and share a log, in the network. Given the growth of devices connected in a network a system, as the one described before, it is able to generate a huge amount of log in short time. For these reasons, solutions for log management must be adopted, especially to handle big data produced by logging systems in an efficient way. Based on those considerations there are two important design aspect strictly related to average data amount that a system is able to handle: real time analysis and data storing. The first concerns the detection phase, because a system will trigger alerts based on some policy logics, differing normal and routine events out of attack pattern or malicious action trying to minimize the number of false positives. While the second one is related to storing, management and pruning after a decided period.

Assuming that the log system will produce a huge amount of data, storing architecture must be able to store that data and recover them in an efficient way. Another issue presented in that phase is related to time interval within the logs are maintained in the system. For this reason, an efficient system of Network Attached Storage (NAS) or a Cloud based solution is required in order to store this huge amount of data. At the same time, we have to protect intrinsically all this information providing at least integrity and authentication (to prevent any unauthorized alteration). In addition, encryption (or just a simple anonymization) is required if data log is considered sensible; this will make storing GDPR-compliant data or satisfaction of other regulations required by the business scope.

Storing all the logs in a secure way is important during the forensic analysis after an attack. If an attack succeeds without any alert or action taken to stop it, SOC must understand why the system was not able to detect that kind of attack. In order to know that, the Blue team has to understand the attack in all its shapes. In order to do that, logs can reconstruct the attack in all its phase, defining the attack pattern, if the logs are corrupted the attacker is able to hide completely its traces and, most indispensable, a reconstruction of the attack itself will be unfeasible. Once those design choices are addressed, the system is able to collect, and consecutively store, in an efficient way the huge amount of data related to the log. Follows the description of the core Blue Team operations, which consists of analyse those data in real time to make prediction and act in case of needed.

Prevention Triad: Identify, Protect and Detect

Once Blue team has collected all the data, it must perform a detection in case some malicious operations are detected. This can be conducted in two ways: reactive and proactive. Reactivity concerns all the operations in which the system, based on a real time analysis, is able to detect suspicious actions in the perimeter minimizing the false positive alerts rate and maximizing accuracy.

In order to do that, analysts often use some Security Information Event Management system which based on detection engines send alerts if some events match an alert policy based on

malicious behaviour in the working environment. Another way to perform reactive analysis is to automatize the procedure thanks to agent on the endpoints (Board Agent) or simply looking for possible suspicious packet going in the firewall (Agent-less), discarding them according to network's policies. From the moment that for an analyst is unfeasible to check at millions of packets per seconds, all of this system is implemented via software, it is executed on a specific node and may expose that node to some vulnerabilities. If the node is not properly protected an attacker is able to take over the node modifying its normal behaviour. This means the attacker is also able to modify the configuration files and the behaviour of the agent present on that node. Suppose that the node taken in exam is an Intrusion Detection System (IDS), if an attacker was able to take over that node and consecutively change the policies for the alerts triggering, then he would be able to connect in a $'$stealth$'$ mode with other hosts without being disclosed. This is a silly example in order to highlight how cybersecurity is not a product or a packet but is a set of assets, approaches, knowledge and criteria aimed to ensure information security. If there is a node with a powerful detection and correlation engine working on top of it but that node is not intrinsically protected, then the security level provided by the system can no longer be assured. If the detection engine works well, analyst is able to detect threats early. Purpose of IRT (Incident Response Team) is to detect malicious activity, investigating and excluding false positive and legal actions. Proactive methods instead involve a human interaction by means of threat hunting operations, which consists of evaluating possible threat in the system before their attacks. This approach is relevant different from Forensic operations which work after an attack as well as PT and VA which simulate an attack from outside. In that case, Threat hunting is performed with the idea that a threat may be already present, then lateral movements and indicators of compromising (IoC) are related in order to highlight a possible attacks and anomalous behaviour concerning the scenario. Detection is useful when behind there is a risk based management framework which has associated identified assets and entity with a logical risk, then based on the risk the entity is protected in a consistent way. As it is stated before those are two of the most important actions also referenced in the NIST Risk Management Framework where Identify is the respective for Prepare and Categorize. At that time, detection will be effectively consistent and based on the risk indicators and tolerance level can be generated whenever something malicious is detected.

In conclusion, all of those operations are used to trigger alerts, which can be handled automatically, or manually, it depends on the criticality of the scenario.

SIEM and Log Correlation

Security Information Event Management is a system able to support the analyst with his monitoring tasks. SIEMs collect lot of information by the log generated in other applications and security components. Among them are relevant firewalls, web servers, access managers and many others; basically, whatever device able to store information about its internal status and timestamp. SIEMs are basically high-level interfaces able to collect and manage data.

Logs of several types can be correlated in order to discover a possible IoA (Index of Attack) according to some user's policies. SIEMs offers quite often some preconfigured pattern in order to integrate huge amount of data from general solutions such as firewalls, AD services, Authentication servers and many others. Data correlations can alert the analyst if some policies are broken.

SIEMs allow to investigate the machine in real time and looking for some evidence or attacks patterns inside the real-time log analysis. During the post incident activities, forensic analyst investigate logs in order to understand the attacks' dynamics. This is a good solution to exploit the static part of a SIEM; at the same time it requires a good storage and back-up solutions. This is used especially to be compliant with some GDPR requirements.

1.4.4 Detection Automations

In this section will be explained which are the main implementation strategies behind detection systems.

Nowadays system complexity does not make possible to view the whole status of the system and analyse if something of malicious is happening for a human operator. Furthermore, if something

strange is detected there is no time for evaluations and analysis otherwise there is no way to contain threats. This is the reason why automated operations are needed.

As was introduced in the previous section, nowadays, there are automated solutions able to support automated detection with the classical approach, used in several antivirus act in a normal way based on hash classification. The problem of this type of identification is that hash classification is a very sensitive mechanism and an attacker can add simple NOP operation in the executable file to evade detection systems like that. Common antivirus block at priori the execution if a malware is detected, but nowadays, those solutions are not deprecated. Many times, system uses behavioural analysis in order to detect anomalies in order to do that Machine Learning is massively used in this field.

Follows the major algorithms [15]:

- Decision Tree Ensemble: the predictive model is composed by several decision trees generating solution such as Random Forest or Gradient Boosted Trees. There are tree in which every non-leaf nodes are evaluations while leaf are resolutions. Again the model evaluates the features of the item in order to categorize its relative class (binary or not). The following image reports a decision tree:

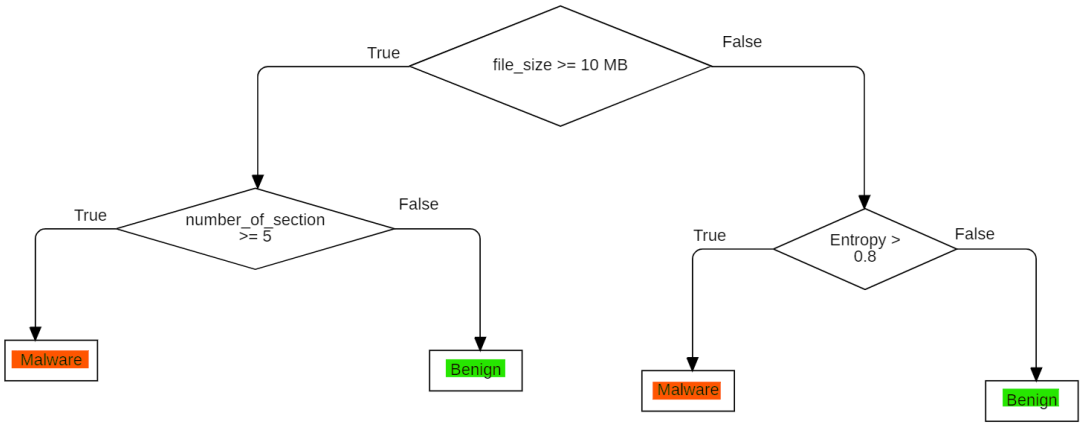

Figure 1.6. Decision Tree Model

- Locality sensitive Hashing - LSH: it is a method to detect similarities in some specific patterns. In order to do that, the system extracts file features and by means of orthogonal projection learning (cost based) it is able to identify the most important features. Then tanks to ML-Compression, set of similar features are transformed into similar or identical pattern.
- Behavioural Model: in the same way as executable files analysis is performed, even Behavioural Analysis exploit Machine Learning Compression in order to detect malicious activity in observed log data our model compresses obtained sequence of events to a set of binary vectors and trains the deep neural network to distinguish clean and malicious logs.

In the next section will be analysed the actual challenging in order to deal with pattern reorganization and malware detection involving Machine Learning solutions.

Hashed Based Malware Detection

Malware detection operations can be applied to local system such as working computer but also to communications, try to think to malware spamming by email or malicious code download from

internet. From the moment that many times organizations can have Mail Gateway and Web-Gateway, those act exactly as a node offering further defense lines where traffic can be analysed before reach its final destination. The perfect example is the one concerning the Web Gateway that basically act as a Proxy. The client connects to the proxy that in turns perform the client requests acting as a middleman, then the request is analysed in the proxy which is able to decrypt the TLS ciphertext (assuming a simple HTTPS connection) and analyse the content in order to check if malicious code is present in the response.

Before the Machine Learning advent, malware where classified based on its cryptographic hash. Antivirus scan works exactly in the same way check all the hash related to files and executable in the system looking for some well know hashed considered as malicious. This technique has its benefit in the sense that whenever detection system matches something that is considered malicious by an oracle (AV signature DB) there is the 100% of assurance that this is a real malware, potentially malicious for the system.

By the way, solutions like that are quite tolerant to minor alterations which do not alter the real code behaviour. Obviously, as stated before, cyberattacker can easily elude the hash detection system just discussed changing its structure but not its behaviour. Suppose that an executable malicious file is composed by the following set of instruction:

```
pushl %ebp
movl %esp, %ebp
subl $0x4, %esp
```

The following code present as simple situation in which CPU save content of *%ebp* register into the stack, then override move content of *%esp* into register *%ebp* and finally performing a subtraction like this: *R[esp] = R[esp] - 0x04*, where convention *R[regName]* indicates the content of the register.

By the way, the same result can be obtained with this other set of instruction:

```
nop
pushl %ebp
nop
movl %esp, %ebp
nop
subl $0x4, %esp
nop
```

Differently from a high level programming language where comments are trimmed out during the compilation phase, NOP operations are persistent also in the executable altering the final hash produced. This assumption is confirmed by the fact that NOP operation present OP-CODE as well other "real" Machine Operations such as SUB, ADD or PUSH. For example its value in x86 CPU Family is 0x90.

According to those facts, a system like the one described before is vulnerable to those kinds of attacks and may not recognize threats. The example presented shows also the major resolution in order to address those issues. Detection system has no longer to evaluate structure of a program, but otherwise they examine the behaviour. In order to do this we need a model and in the next section will be described some important innovation in the Machine Learning scenario able to address this kind of situations.

Machine Learning Based Malware Detection

In the previous section was stated the reason why hash based malware detection are considered deprecated nowadays, but in this section will be give solid assumptions to understand and approve the definitive win of machine learning approach in cybersecurity. Machine learning techniques presents various algorithm to statistically analyse the executable binaries in order to understand some common patterns. All the aspects presented will find practical implementation in the code developed and in the technical considerations discussed in the Appendix A attached to this artefact.

Among the solution involved in the pattern recognition there is the one known as N-Grams which aim is to understand and predict statistically the N-index possible item of a sequence based on math's assumptions. Follows an example to understand better the scenario:

```
string = "I am a human being"
N-Grams(1, string) = ["I", "am", "a", "human", "being"]
N-Grams(2, string) = [["I", "am"], ["am", "a"], ["a", "human"], ["human",
    "being"]]
N-Grams(3, string) = [["I", "am", "a"], ["a", "human", "being"]]
```

Basically this is the model that is used to process natural languages understanding that composition and structure of complex sequence of string or in our case machine instructions. Those considerations are mixed with statistical analysis that can be made on a binary file. Assuming to deal with PE (Portable Executable) widely used on Windows System, some metrics can be:

- Number of Sections
- Imported DLLs
- Type of Section

Now using a supervised approach, a binary classifier (Malware or Not-Malware) implemented with RandomForest supported by N-Grams Analysis and PE stats, can assure an accuracy between 96% and 99% [16] based on the train-test-split sets definition and data accuracy.

This detection system is no longer tolerant to structure's minor alterations and can be integrated with classic hash based systems. In addition, whenever new datasets with threats are composed, model like that can be continuously be trained to be able to detect and classify correctly new malicious entity. This means again that the system can be calibrated based on scope threats or past evidences leaved by malicious code, and from the moment that it computes behavioural analysis can recognize 0-day threats and new malware. They represent features not feasible and implementable for a simple hashing detection system.

This is just a simple example but modern dataset permits producing models which works on multiple classes, not only with binary ones. This is able to recognize threats in a more accurate way and a specific pattern to a specific category. This is what is done in MITRE ATT&CK Framework where based on techniques or well-structured operations a threats is matched. Those probabilistic models permits also to give a percentage of matching for each class: for example a sample can be categorized as follows for 90% Ransomware, 7%Trojan and remaining as 3%Rootkit.

Anomaly Detection, UEBA and Isolation Forest

Machine learning and statistical analysis can be involved also to detect anomalies in the network. This is made by solution known as Isolation Forest which are composed basically on the same aspect of Random Forest but after feature extraction they select isolated points according to random split in the hyperplane. Those type of techniques permits detecting isolated point because they are lonely respect to other ones around a centroid.

Random Forest are ensemble model that works like follows: a feature is selected and the range of value is represented as numerical, then it is split based on a threshold value [16]. If the two partitions do not contain single point the process is repeated based on another feature until a single point partition is created. The partitions can be mapped as a binary tree and lower will be depth from the root for a single partition point, higher will be is anomaly score. Will be created N of this tree and then based on a statistic decision such as how many times the point has a high anomaly score will be defined the result. The main ideas is that common points requires more partitions to be isolated respect to lonely points as shows in the image 1.7.

This kind of system are widely used to detect network anomalies, and more in general component anomalies. Especially in user behavioural analysis', Isolation Forest model and data analytics are involved in DLP processes and data-oriented-protections.

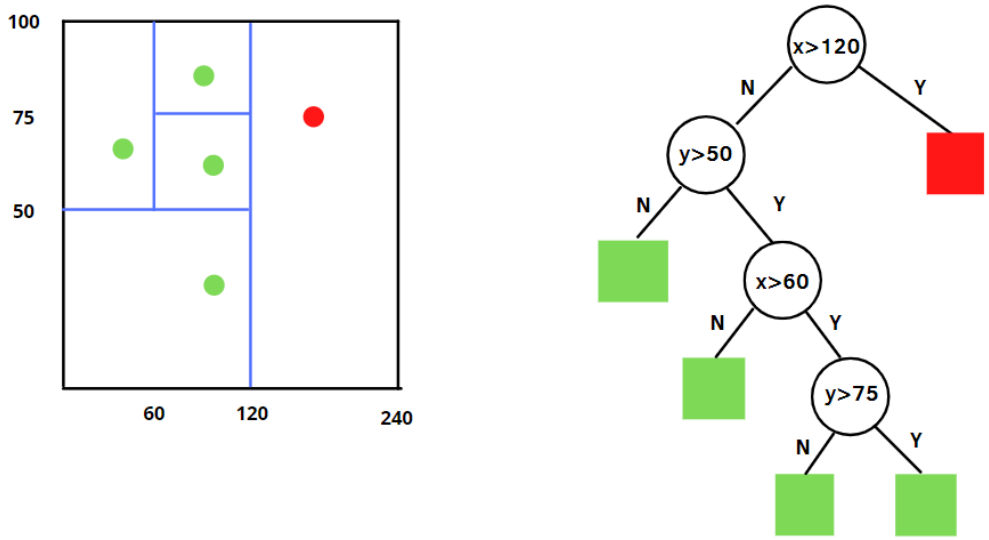

Figure 1.7. Isolation Tree Model

For example if an employee deletes 5 files as average in a month and then after his firing the DLP system detect a bulk deletion of 500 files maybe it is a possible armful situation for the company's data that will be destroyed. Solution like that can also detect data anomalous data exfiltration based on what is common for a system that basically is easier to produce respect what is uncommon. For this reason manufacturers as Lookout are defined by Gartner 2022 magic Quadrant for SSE (Security Service Edge) as Visionary; because the product developed poses its attention on DLP and Data Protection by means of technological implementation as Random-Forest. DLP solutions covers a central role to support system to be compliant with GDPR requirements. Viewed as an anomaly the flow is detected and blocked, at the same time can be possible normal situations because another important concept behind Anomaly detection in cybersecurity is that not all anomalies are real threats, while (in well tuning systems) every threat is considered an anomaly.

Those system are able to block also 0day attacks if the pattern and network traffic involved is strange for the system itself [16]. The tuning aspect cover a central role in order to avoid false positives and recognize effectively anomalies, those kinds of operations are highly sensible to background noise and data quality. In the Appendix A.2.2 will be described an example of Random Forest functionalities with KDD-NLS dataset: a network based pattern identification used to test IDS in production. This concludes the Machine Learning aspect concerning the detection phase.

Investigation and Forensic Analysis

In this section will be presented the main operations concerning data investigation and forensic analysis in case of a successful attack. This phase is of vital importance in order to understand from data and logs, which is the point in time when a system was compromised and which is the root causes regarding the compromise.

In that case, post incident investigation which works exactly as a detection, but in addition the analysis are performed on collected data instead of a close-to-real-time analysis as the one performed by a SIEM or and EDR during their normal activities. The review after a successful attack is crucial in order to understand dynamics, what did not work, how was possible to elude detection system or why no action was taken by automated response to deal with this specific threats.

In addition, in EU country there are also compliance's aspects to keep in mind that, again, regards the data integrity, availability and confidentiality from the moment that as stated in the

initial part of this artefact, companies must notice data breach within 72 hours according to GDPR. Then to implement also Resiliency and Business Continuity aspects', the company has to understand which is the instant when the resources were compromised, analysing legal alteration by user and trying to obtain consistent status of data otherwise has to considered all the data edited legally after the attack as inconsistent. In front of this scenario company has to re-collect data in order to verify its validity and consistency.

This is the reason why data and log collection are not important only to detect possible threats in the system, but also to define valid recovery points in time, minimizing the possibility to recover again a compromised version of the server whenever a recurrent back system was up and running in the system. Finally, for proactive reason is important to assess the system understanding what can be improved not only in case of successful attack, but also in case of well-contained attack. This is the real essence of the PDCA model, there can be always improvements margin if data are collected and useful information are extracted.

1.4.5 NIST Cybersecurity Framework

NIST Cybersecurity Framework [17] was released first time in 2014 by the US National Institute of Standard and Technologies and was originally a document designed for operators of critical infrastructure. In 2017 was released version 1.1 for a public comment and in 2018 the final release was available.

The framework is based on evaluation of the current status according to evidence of an investigation, and it tries to evaluate the risk of the system; if the latter is not acceptable some action must be taken in order to reach the designed status according to the desired risk level. Furthermore, this framework offers also the possibility to prioritize improvements in order to make continuable and repeatable the entire process. This is one of the most important features, and it inherits the recursive model offered by the PDCA model. The progress is assessed by the evaluation of the target state toward its phases permitting communication among internal and external stakeholder about cybersecurity. This fact made this standard integrable and extendable with assessment expressed in ISO27001 [2] etc...

The framework is a risk-based approach to handle cybersecurity, it is composed by three principal models that will be explained in the next subsections.

- **Framework Core**: It provides a set of operations to achieve a specific cybersecurity outcomes including technical reference and examples aimed to implement the requirements in order to satisfy those outcomes. It can be defined as a matrix composed by four principal element Functions, Categories, Sub-Categories and Informative References. The first element is represented by the Functions that are identified by 5 macro requirements to be implemented in order to manage cybersecurity risk, and they extend the three main operations stated in the Blue Teams operations explained before. They are listed below:
  - IDENTIFY: Develop an organizational understanding to manage cybersecurity risk to systems, people, assets, data, and capabilities. This activity resume the importance to know the environment identifying crucial assets in order to evaluate correctly the risk associate with it and, based on the evaluation performed, prioritize the improvement in order to maximize the efficiency according to the desired final states. Again also the final states definition is challenging without knowing the current state of the system. The successful completion of operations are based on how detailed and accurate is the IDENTIFY phase.
  - PROTECT: Develop and implement appropriate safeguards to ensure delivery of critical services. The PROTECT phase aims consist of limit or contain the impact of a potential attack by means of Identity Management, Access Control or Protective Technologies.
  - DETECT: Develop and implement appropriate activities to identify the occurrence of a cybersecurity event. This phase concerns the one explained in the Blue Team operations, and it is the initiator of the Incident Response operations.

  - RESPOND: Develop and implement appropriate activities to take action regarding a detected cybersecurity incident. It is the Core of the Incident Response Operations in order to contain and mitigate the threat according to best practices.
  - RECOVER: Develop and implement appropriate activities to maintain plans for resilience and to restore any capabilities or services that were impaired due to a cybersecurity incident. Recover normal operations and activities in order to guarantee the normal business continuity and resilience of the system. In this phase there is also an important phase of review and learning in order to highlight what has been done in the right way in contrast of what can be improved. This phase concludes the cycle and based on evidence collected in the last phases there can be a further iteration in order to improve the system in a consistent way according to the requirements.

  NIST Cybersecurity Framework, based on those functionalities, provides the second element of the four listed before that is identified by the Categories. Categories are the subdivisions of several cybersecurity outcomes based on the Functions they belong. They are strictly related to programmatic needs and activities. Then categories are divided into Sub-Category. They provide a set of result that, while not exhaustive, help support to achievement of outcomes in each Category exploiting a divide-et-impera approach.

  Finally, Informative Reference are specific sections of standards, guidelines that illustrate a practical implementation in order to satisfy the cybersecurity requirements expressed by category and subcategory to achieve the outcomes desired. They are based upon cross-sector guidance most frequently referenced during the Framework development process. As the next figure shows, there are many Categories for each function. As stated before many times standards as ISO27001, COBIT or PCI-DSS can be used as Informative Reference in order to use them as guidance to implement requirements as reported in 1.9.

- **Framework Implementation Tiers**: Aim of this section is to define context on how an organization feels cybersecurity risk management according to sophistication in risk management process and operations. There are four total Tiers and its selection is based according to several factors such as threat environment, risk management practices, legal and regulatory requirements etc... Tiers are meant to support organizational decision-making about how to manage cybersecurity risk, as well as which dimensions of the organization are higher priority and could receive additional resources. Progression to higher Tiers is encouraged when a cost-benefit analysis indicates a feasible and cost-effective reduction of cybersecurity risk. Successful implementation of the Framework is based upon achieving the outcomes described in the organization's Target Profile(s) and not upon Tier determination. Still, Tier selection and designation naturally affect Framework Profiles. There are four tiers, starting from the simpler one they are listed below:

  - Tier 1 - Partial: Tier 1 have an ad-hoc and reactive cybersecurity posture to protect data and assets. They have little awareness of cybersecurity risk and often plans implemented are done inconsistently.
  - Tier 2 - Risk informed: Cybersecurity risk organizations may approve cybersecurity measures, but implementation is still fragmented. They are aware of risks, have plans and have the proper resources to protect themselves from data breach but there is no proactive operations implemented.
  - Tier 3 - Repeatable: The third tier is called repeatable, meaning that an organization has implemented CSF standards in all main business sectors and are able to repeatedly respond to cybersecurity events. Policies are consistently applied, and employees are informed of risks, improving global awareness.
  - Tier 4 - Adaptive: Called adaptive, this tier indicates total adoption of the CSF. Adaptive organizations are not just prepared to respond to cyber threats, there is also a massive proactively operations in order to ensure an appropriate protection level.

  Each one is designed according three main metrics that are Risk Management Process, Integrated Risk Management Program and External Participation. Aim of this artifact is to pose the basic aspect in order to see how they can be satisfied in automated operations. Further information are available on the NIST Cybersecurity Framework White Paper.

- **Framework Profile**: Framework profile aims is to relate consistently the elements presented in the Framework Core to the business requirements, risk tolerance and resources of the organizations. Profile definition are useful in order to design the current state and the target one. Current state is evaluated based on cybersecurity outcomes achieved while target states is defined to achieve the desired cybersecurity risk management goals. Comparison of Profiles (e.g., the Current Profile and Target Profile) may reveal gaps to be addressed to meet cybersecurity risk management objectives. An action plan to address these gaps to fulfil a given Category or Subcategory can contribute to the roadmap described above. One important aspect to consider is to prioritize the mitigations required in order to cope with costs and time frames needed to implement those requirements. Follows the information required to define the target profile that are Priority and Sub-Category involved, they are evaluated by the initial state which in turn is defined according to business operations, regulation and other aspects as stated above.

Subcategory	Priority	Gaps	Budget	Activities (Year 1)	Activities (Year 2)
1	Moderate	Small	$$$		X
2	High	Large	$$	X	
3	Moderate	Medium	$	X	
...	...	...	...		
98	Moderate	None	$$		Reassess

Figure 1.8. Profile review

Function	Category	Subcategory	Informative References
RESPOND (RS)	**Response Planning (RS.RP):** Response processes and procedures are executed and maintained, to ensure response to detected cybersecurity incidents.	**RS.RP-1:** Response plan is executed during or after an incident	**CIS CSC** 19 **COBIT 5** APO12.06, BAI01.10 **ISA 62443-2-1:2009** 4.3.4.5.1 **ISO/IEC 27001:2013** A.16.1.5 **NIST SP 800-53 Rev. 4** CP-2, CP-10, IR-4, IR-8
	Communications (RS.CO): Response activities are coordinated with internal and external stakeholders (e.g. external support from law enforcement agencies).	**RS.CO-1:** Personnel know their roles and order of operations when a response is needed	**CIS CSC** 19 **COBIT 5** EDM03.02, APO01.02, APO12.03 **ISA 62443-2-1:2009** 4.3.4.5.2, 4.3.4.5.3, 4.3.4.5.4 **ISO/IEC 27001:2013** A.6.1.1, A.7.2.2, A.16.1.1 **NIST SP 800-53 Rev. 4** CP-2, CP-3, IR-3, IR-8
		RS.CO-2: Incidents are reported consistent with established criteria	**CIS CSC** 19 **COBIT 5** DSS01.03 **ISA 62443-2-1:2009** 4.3.4.5.5 **ISO/IEC 27001:2013** A.6.1.3, A.16.1.2 **NIST SP 800-53 Rev. 4** AU-6, IR-6, IR-8
		RS.CO-3: Information is shared consistent with response plans	**CIS CSC** 19 **COBIT 5** DSS03.04 **ISA 62443-2-1:2009** 4.3.4.5.2 **ISO/IEC 27001:2013** A.16.1.2, Clause 7.4, Clause 16.1.2 **NIST SP 800-53 Rev. 4** CA-2, CA-7, CP-2, IR-4, IR-8, PE-6, RA-5, SI-4

Figure 1.9. Response details

1.4.6 ML and Detection Requirements inside NIST Cybersecurity Framework

Reviewing the main aspects introduced in the Anomalies and Events of NIST's Detection phase [17], anomalous activities should be recognized in order to evaluate the potential impact of threats.

To evaluate the impact can be assumed that each entity in the network has a business value in terms of data managed, and core functions performed within the business cycle. A server or a laptop can have assigned a numerical target value which means the criticality of that host. See specifically the subcategory for this macro-area:

NIST DE.AE

This a governance oriented requirements in which there is the necessity to monitor and prevent asset and resources looking for anomalies or attacks. Those activities can be the real communications needed inside the network for two simple reason: the first is to reduce background noise in the network, the second is to have a clear picture of what is happening inside the network, especially in big environment such as company and organizations. As expressed in the ISO27001 A.12.1.1-2 the operative architecture must be documented in order to understand real necessity in the network designing a surface, minimizing the unusual sessions and be proactively reactive to changes as patches or reconfigurations. Furthermore, network and security must be implemented also to highlight weak ciphers or session involving old versions of TLS protocol as stated in ISO27001 A.13.1.1-2. For this purpose anomaly detection systems can be implemented with network monitoring in order to highlights anomalous flows in the ones documented. In this section is also described the necessity to detect events and understand attack proactively. Detection system able to collect huge amount of logs can help in, remembering SIEMs features company can easily collect data from multiple sources in order to expand their visibility and correlation engines referring as stated in ISO27001 A.12.4.1. While in the ISO27001 A.16.1.4 is expressed that once a security event has been reported and logged, it will require to be assessed in order to evaluate the best course of action to take. Those requirements poses the bases for an effective and efficient Response phase. Once solution like that are able to categorize attacks can be useful to extract stats in order to understand critical and frequently targeted hosts, in case of 0-day attack known by CERT and CSIRT Intelligence sources is able to prioritize the patching machines based also on this information, again this parameter can be involved in the classification of the risk.

Now that the risk is generated involving various shapes of the company processes and status is possible to classify the impact of an event based also on involved devices' risk score. It is simple to see that to do those type of considerations a company need huge amount of data regarding its company network traffic, structure and architecture. So machine learning based models can be involved in order to classify a treat or to predict possible impact in similar pattern according to old events and attacks. From the moment that not all anomalies are threats a good machine learning model must detect anomalies and then has to classify the anomaly based on the flow pattern highlighting what can be assumed as a treat. This increment the risk score which related its impact can determine a severity level in order to prioritize the entire process. From the moment that log and information can be used as evidence for attack and juridical operations, company has to apply this governance process in order to implement requires express in A.16.1.7 which is out of scope respect machine learning applications.

NIST DE.CM

In this section will be presented the support needed to implement the Continuous Management macro area. Latter implies to support all the processes required in order to collect and correlates logs. According to ISO27001 A.12.4.1 requirements, the system must be able to defense-in-depth operations. As stated before SIEM can be good alliance to make sure all logs are collected from multiple sources and then correlated into a single brain. Implementing specific features extractor is possible to detect anomalies which can result in data breaches or dangerous situations. At the same time for forensics operation log must be protected in order to have redundancy in case of recovery. Those logs can be analysed with a regular cadence in order to highlight compromises, reducing the analysis interval can be globally increased the efficiency and the response in this scenario. For this reason ISO27001 A. 12.4.3 requires that logs must be accessed only with administrator grants and where possible the unique access is in append mode in order to remove possibility to delete log entirely.

At the same time those kinds of considerations must be performed also locally at device level and user level. Exploiting machine learning and user activity is possible to detect access to unauthorized resources based on well-structured policies. Otherwise, is possible detect anomalous access performed by user analysing the file accessed by others users with the same role. This can also be a good detection system in order to detect anomalous attempts to access log by uncommon users referring the Annex A.12.4.3 task. DLP system otherwise offers user behavioural detection in order to analyse anomalous deletion in bulk as well as a malware detection system is able to recognize suspicious shadow copies deletion suggesting a ransomware behaviour before encryption. In order to reduce false positive those system will correct BIAS by means of well-declared exceptions for example deletion of the shadow copy after the retention time. At the same time at local level is important to detect code execution and prevent armful code for the business and the processes which governs the core functionalities. In the previous sections, it has been showed the limitation of AntiVirus, nowadays they can be integrated with machine learning pattern detection system with statistical analysis in terms of number of sections, contents, sections and DLL invoked to improve efficiency in malware detection (ISO27001 A. 12.2.1). This can be also implemented following approaches expressed in the A.12.5.1 in order to verify the signature and manufacturers behind a software installed on a deployment machine as well as the guarantees provided by the manufacturer in case of audit on third party components. Again Machine learning and automations can prevent installation of banned-list software as P2P or TOR Browser on company assets, those restrictions be easily set up with an XDR locally on the machine or with network controls. By the way is better to act at device level in order to be compliant also with A.12.6.2 regarding formally *Restrictions on Software Installation.*

One of the most important aspect concerning the identification of malicious code execution are the variability of the vectors, those kinds of considerations has to be extended also at email and web level. Email gateway can be adopted with malware detection system as well as web gateway, those solutions can be then supported with specific machine learning based detection tools such as phishing detection system, banner as well as anti-spam filter in email scope. At the same time solution such as anomaly detection to recognize peak in the traffic to detect DoS or to recognize frequent pattern in HTTP request to detect similar packets highlighting cyber actor's signature.

This solution has another useful characteristics that intrinsically secures and monitors also the connections with the external entities. In that way especially in interactions with external providers can be used prediction models to understand when the request are legal or not and more important, in order to monitor third party security level according to impositions expressed in A. 14.2.7 and A.15.2.1. There is one last aspect to take into account, and it concerns the DE.CM-4 which discuss *the monitoring of cybersecurity event potentially malicious for the system.* In this case anomaly detection system are not unquestionable Oracle able to predict malicious activity with 100% of accuracy. As we stated in the previous section the example based on the KDD-NLS dataset demonstrate that based on the train and test split partitioning accuracy can change but always maintaining an accuracy grater than 90% with a good cutoff value. That means, operation like that always requires a human interaction in order to discriminate between threats and abnormal activity. By the way according to the results described, those models offers an indication of anomaly that is better than guess, so very precious for the analyst.

The last point discussed in the Detection's Continues Monitoring point concerns A.12.6.1 which is related to Management of Technical Vulnerabilities, and it is strictly related to response operations whenever a vulnerability is found in the system.

NIST DE.DP

As in NIST DE.AE, it is a governance oriented set of tasks related to responsibility and processes involved in the system in order to ensure accountability of the entire system. This macro-area concerns detection system testing in order to ensure awareness in the system and keep this system up to date and improved based on the requirements and behaviour observed in to the wild.

Related to automations and machine learning approaches are out of scopes controls expressed by A.6.1.1 and A.7.2.2 concerning Information security Roles and Responsibility as well as Information Security Awareness, Educational and Training. Actually, there can be a minimal implementation based on data analytics which can highlight user training and education based on

the most common bad practices and attacks pattern mined in the logs, a classical example are Phishing campaign.

Most important are the controls expressed in the A.14.2.8 concerning guarantee that a given detection system is able to detect anomalous event as well as is efficiently calibrated and designed to support detection operations. In order to do that external entities can assess and test the component, in our example a machine leaning model is trained with KDD-NLS dataset that as stated in the previous section is widely used to test IDS. Those aspects are strictly related the one expressed in the A.18.1 family concerning the compliance aspects, also for accredited testing company and standards.

The last point concerns the improvement in the detection system according to new threats and data collected, in this way prediction models can be easily extended with more recent and newer dataset in order to keep into considerations new malware and attack patterns. This is an operation that can be easily extended introducing cooperation among company and organizations as CERT and CSIRT which are able to provide useful information and dataset to do this kind of improvements, as well as internal detection and audit.

1.5 Reactive Operations

Reactive operations consist of all the actions taken to deal with incidents or dangerous situations especially during a cyber-attack, once it is detected; as well as maintain business operations also in case of a successful attack implementing a Resilience and Business Continuity strategy.

Again the involved phases, according to NIST Cybersecurity Framework [13], are the last two: Response and Recover. The response phase is strictly related to reactive operations, also known as active cyber defence namely a Synchronized, real-time capability to discover, detect, analyze, and mitigate threats and vulnerabilities, according to NIST Glossary, to contain and eradicate an attack once it will be by the strategy discussed in the proactive operations. Detection and consecutively a well-planned response depends on the detection system as the NIST Documentation states; again the cooperation among proactive and reactive operations is the milestone to build up a secure posture for the system according to the major standards.

The other two important aspects are dealing with reactivity operations. Those are related to Incident Response Team operations whose aim is to contain, eradicate and recover the system in case of issues. The first regards recovery and post-incident phases concerns can highlight what has been working and what can be improved by implementing the hardening requirements according to the Proactive Cyber Defense discussed in the previous sections.

The second concerns efficiency in terms of effectiveness and speed of reply. Latter can result in effective containment in case of cyber-attack reducing the intervention in the post-incident phase, concerning Hardening and Recovery, or can lead to a disaster recovery in case the malicious activity is not contained and eradicated as soon as possible.

This is the reason why nowadays many technological solutions can help the incident response operator to act as soon as possible consistently and effectively. Among those solutions there is XDR in fact, respectively to passive solutions such as AntiVirus and NGAV, XDR can decide in an autonomous way to block the execution of a malicious code or permits the possibility to remotely isolate the device as already discussed in the previous section. By the way, there is the necessity to have a Global view of what is happening in the system and that can be satisfied through log collection from specific traffic, device, and configurations changes as a SIEM does.

Then there are other important components strictly related to the SIEM that is the SOAR. It can automate and orchestrate a response to cyber-attacks based on the correlation performed by the SIEM, that many times is integrated with the SOAR itself. This system will be explained in the next subsection.

1.5.1 Incident Response

The response phase involved all the actions performed in responding to an alert triggered during the detection phase. As stated before, the nature of that response action may be both automatic and manual. The detection phase is an indicator of possible incoming attacks or potential malicious situations, which must be handled in the best way. From the moment that there may be several alerts to make supervise an analyst, a scoring system that schedules alerts based on their severity impact should be implemented. At the same time, modern SOC requires an ad hoc system to take automatic actions for alerts selected to not overload the analyst who can dwell longer on the critical alerts.

Now the basic question is "How to differentiate the critical alerts from the non-critical ones?" as an engineer probably responds the answer will be "it depends...". It depends on the scenario assuming some basic milestones. If the system detects possible DoS due to a high peak of requests from a specific IP, an automatic response can block the ongoing traffic for that host defining some temporal policies (quarantine(x days) or never expires). Otherwise, if an attack is detected on a SCADA system of a power plant, maybe halting the system in order to physically isolate itself can have a huge impact since can be created disservices in the power supply chain. In that case, a senior analyst has to handle the situation by evaluating possible solutions in order to minimize the disservice and the impact of the attack.

Summing up, the decision to take autonomous actions is strictly related to the environment and the safety-critical level of the operation itself but nowadays-useful tools such as machine learning and AIs can predict correctly the action to take to minimize the possibility of disservices and the response time. For this purpose important frameworks such as NIST Cybersecurity Framework, and ISO/IEC-27001 have been developed.

E/XDR and NGAV

The endpoint Detection and Response system represent a professional strategy to protect devices. They are agent-based solutions able to detect and respond to several threats involving workstations, laptops, servers, and whatever endpoint requires local protection.

Those platforms allow identifying possible threats using real-time monitoring and alerting. Whenever an alert is triggered those solutions allow the analyst to perform an investigation and provide some important information such as the kill-chain of the event, involved machine, connections performed, the reputation of the executable ran, and many others stats. In addition, some policies can be set up in order to avoid the execution of some well-known malicious executables based on the SHA256 reputation of that file. In case a device has been compromised, those kinds of solutions permit to quarantine of the device in order to block all the ingoing and outgoing connections, except the ingoing through the secure channel instanced by the EDR's agent. From the moment that EDR collects a lot of data, this solution can be also used in order to perform several types of investigations such as proactive monitoring, threat hunting, and post-incident forensics analysis.

XDR is the evolution developed to cope with the IT scenario's demands. Nowadays, emails and Digital Identity are becoming basic aspects to be managed in all businesses. For this reason, Extended Detection and Response solutions extend the EDR concept in that way. They perform detection and monitoring based on telemetry provided by several sources such as, precisely, emails, access locations, and digital identity providing more visibility. Furthermore, the data collected can be correlated with others provided by data lakes to detect anomalies exploiting User and Entity Behavior Analytics.

Next Generation Anti-Virus are modern solutions to extend the old signature-based detection antivirus. They use in a massive way technologies such as Machine Learning and User Behavior in order to detect possible threats. They are not XDR because does not offer response phase mitigations, the reason why NGAV solutions are just prevention components.

SOAR

SOAR stands for Security Orchestrator and Automation Response, it is an extensible tool used to support Incident Response Team during their tasks. It offers a more general configuration in order to implement several scenarios and events following some playbook or procedure. Especially in big companies, it is a relevant tool to handle the heterogeneity of data and procedures involved. It is a good solution to handle global security operations and several subnets.

It offers also ticketing and reporting services integrated in order to process security alerts based on their severity. Latter can be defined based on the impact and the time elapsed and from triggering instant to take over. In that way an analyst can focus his attention on the high priority, tracking response duration by IRT.

The scope can be detected based on the scenario, for example, phishing campaigns, malware campaigns, or endpoints security events. This architecture allows also the correlation of redundant events, especially in the mail scenario where during the campaign the malicious sender can spread the email to several internal users. The analyst can block entities correlated to the mail such as the sender or a malicious URL and block automatically all the mail with that specific entity inside.

A Great feature offered by SOAR is the opportunity to parse automatically in a concise way all the information related to an incident exactly as a SIEM but, further functionalities provide the opportunity to act against all the possible entities considered as an IoC. The entity can be edited to match the pattern of logs and, for each entity, some actions can be automated such as a block of mail sender in the mail server, block IP in the firewall etc... This system is easily extended with some threat intelligence feeding systems to develop a proactive strategy and to detect possible threats before they attack the system.

1.5.2 Governance and Internal Management

Assuming to adopt security solutions, thanks to the ones previously explained a company can be able to detect and consecutively respond to cyber-attacks. In this section will be explained the operative chain used by the Incident Response Team in order to handle and mitigate an incident in an efficient way. Once is clear the escalation chain, will be explained the main technologies involved to support analysts in their tasks.

Furthermore, they will be discussed in order to underline the main TTP (tactical techniques and procedures) with a glance at the help of the playbook in security incident management. Playbooks are useful in those scenarios because they can represent with a high level of formalism all the kill-chain adopted to contrast and mitigate the attack. Thanks to their formalism playbooks can be the building block for an automatic response orchestrated by SOARs. If a new event or hidden shapes of an attack are discovered playbooks must be updated in order to stay consistent and, more important, promote hardening and continuous review activity; again related to the PDCA model to improve the quality and efficiency of Incident Management.

A well-managed Incident Response Team is composed of two or more operative lines to separate the analysis and the actions to be performed in order to handle the incident. Often there are two incident teams: the first layer is responsible for primary analysis, highlighting the IoC and then taking action according to the evaluation performed previously.

The term IoC means a possible compromise related to a specific entity such as IP, mail sender, file hash, and URLs. The first line will send a detailed report to the second line which has more privileges: such as performing blocks of URLs, mail sender, domain, file hash, IPs, and, more precisely, blocking every entity that is considered malicious according to threats data and evidence. Otherwise, in case of compromise, it may opt for logical or physical isolation of the involved machine and other more invasive actions.

Finally, to respect commercial aspects and contracts the service manager has to prove that timing in operations meets the ones described in the SLA (Service Level Agreement), which basically is an agreement between two parties concerning the duration, phases, and operative

workflow in case of an incident. For example, there can be the same take-over time and response time based on the severity of the attack detected. Many times inside this document is also defined the RACI Matrix which associates human resources with their operative role, this is covering a crucial aspect in the satisfaction of GDPR document required for compliance answering to *Who does what?*. RACI acronym stands for:

- Responsible: who performs the activity.
- Accountable: who has the responsibility for the results produced by the activity. It is uniquely defined.
- Consulted: who collaborates with Responsible to perform the activity.
- Informed: who must be informed on execution and completion of activities.

In this way roles are well-defined and, models like RACI can be applied in several situations, making it usable in different responsibilities definition according to GDPR.

1.5.3 NIST Incident Response Framework

In this section will be explained the NIST Incident Response Framework in order to highlight how all those reactive operations can design strong proactive and predictive hardening strategies.

According to NIST SP 80061 (Rev2̇) publication, incident response teaming is not only related to incident response. Other major sections presented in the document concern also organizational and capability aspects to design the team and resources, and, more important, also a robust knowledge-sharing system according to the predictive approach concerning threat intelligence and threat hunting fields.

NIST Incident Response Lifecycle [13, 18] consists of 4 phases, and it is resumed in the following points:

- **PREPARATION**: Incident response methodologies typically emphasize preparation, not only establishing an incident response capability so that the organization is ready to respond to incidents, but also prevent incidents by ensuring that systems, networks, and applications are sufficiently secure. Although the incident response team is not typically responsible for incident prevention, it is fundamental to guarantee the success of incident response programs.
- **DETECTION AND ANLYSIS**: Incidents can occur in countless ways, so it is infeasible to develop step-by-step instructions for handling every incident. Organizations should be generally prepared to handle any incident, but they should focus on being prepared to handle incidents that use common attack vectors. Different types of incidents require different response strategies.
- **CONTAINMENT, ERADICATION, AND RECOVERY**: Containment is the primary action to perform before an incident overwhelms resources or increases damage. Many incidents require containment, so it is an important consideration early in the course of handling each incident. Containment provides time for developing a tailored remediation strategy. An essential part of containment is decision-making (e.g., shutting down a system, disconnecting devices from a network, disabling certain functions etc. . .). If incidents disrupt some internal functionalities a recovery strategy can lead IRT to remediate and restore normal activities. Such decisions are much easier to make if there are predetermined strategies and procedures for containing the incident. Organizations should define acceptable risks in dealing with incidents and develop strategies accordingly.
- **POST-INCIDENT ACTIVITY**: One of the most important parts of incident response is also the most often omitted: learning and improving. Each incident response team should evolve to reflect new threats, improved technology, and lessons learned. Holding a "lessons learned" meeting with all involved parties after a major incident, and optionally after lesser incidents as resources permit, can be extremely helpful in improving security measures and

the incident handling process itself periodically. Multiple incidents can be covered in a single lesson-learned meeting. This meeting provides a chance to achieve closure concerning an incident by reviewing what occurred, what was done to intervene, and how well the intervention worked.

Infographic 1.10 is shown to understand better the execution flow concerning the incident response operations.

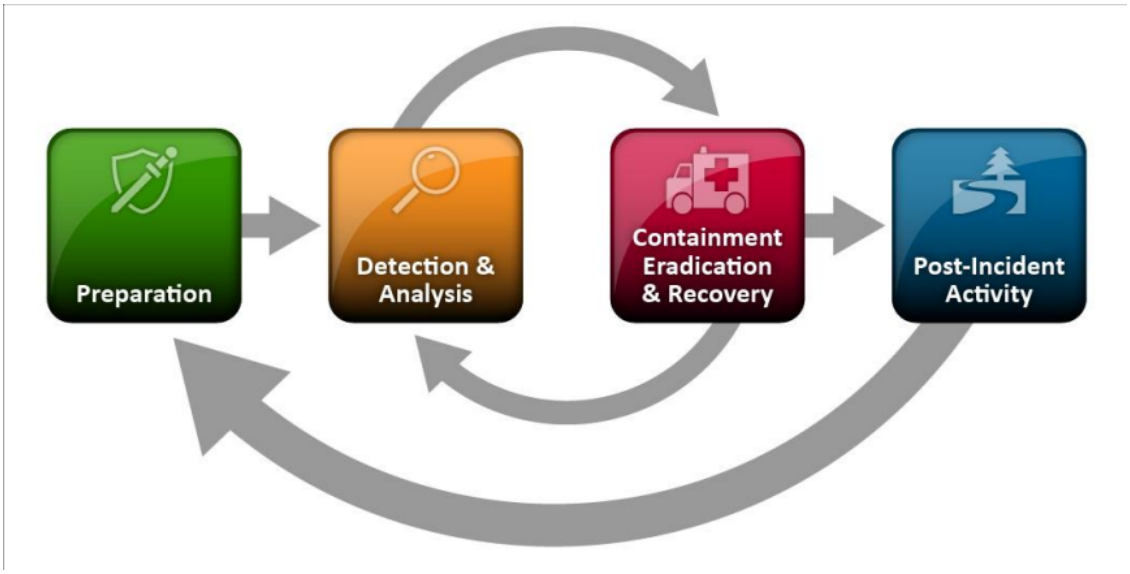

Figure 1.10. NIST - Incident Response Life Cycle

In addition, the incident response guidelines provided by NIST, latter promotes also the cooperation among organizations to share common situations and incident response plans, and most importantly, share IoC and malicious actors or entities.

Organizations often need to communicate with outside parties regarding an incident, and they should do so whenever appropriate, such as contacting law enforcement, fielding media inquiries, and seeking external expertise.

Another example is discussing incidents with other involved parties, such as the Internet Service Providers (ISPs), vendors of vulnerable software, or other incident response teams. Organizations may also proactively share relevant incident indicator information with peers to improve the detection and analysis of incidents.

The incident response team should discuss information sharing with the organization's public affairs office, legal department, and management before an incident occurs to establish policies and procedures regarding information sharing. Otherwise, sensitive information regarding incidents may be provided to unauthorized parties, potentially leading to additional disruption and financial loss. The team should document all contacts and communications with outside parties for liability and evidential purposes.

In the image 1.11 are summarized the involved actors concerning communications after an incident. All these information-sharing operations can be viewed as part of Post-Incident actions to be taken to improve awareness among Incident Response Teams and to be compliant with local regulations such as GDPR which requires informing the Supervisory Authority within 72 from the data breach.

1.5.4 Response Phase Automations

As already discussed for detection automation, also for response automation can be applied same baselines.

Figure 1.11. NIST - Communications with Outside Parties

The principal scope is to speed-up events that can be automated to be more efficient and improve the global effectiveness of response phase according to this statement: *"Earlier will be performed response and mitigation, lower will be the impact"*.

In response, the main aspect concerns reactivity and timing from the moment that Incident Response belongs to Reactive Operations, where according to NIST Incident Response Framework Containment and eradication are two of the core functionalities to manage the incident in the most effective way. The reason behind those considerations is that, unless configuration problems, an automated solution based on API interconnection between a SIEM and an XDR is many times faster than a manual one. For example, in a company, the SIEM deals with detection while EDR with incident response locally on the device. By the way, EDR does not have visibility about network traffic while SIEM does. SIEM detects a loud anomalous outgoing data traffic by this device and correlated the event with a possible data exfiltration. A classical solution that follows the work-flow takes time to raise the alert, it waits for the analyst's takeover and considerations, then he has to switch the console, identify the device, and request an isolation or block.

In the meanwhile, a possible malicious operation has exfiltrated GBs and GBs of traffic against many statements of GDPR and ISO27001. Again the problem is related to human performances concerning automated ones. On the other hand, an automated solution can be implemented blocking all network interfaces when a situation like that is detected thanks to direct integration among the Detection system and Response agents on the machine. This is just a simple scenario, but in that case, it shows the real powerful automation in detection and response phases to mitigate close to real-time security events.

By the way, there are many points to keep into consideration in automated solutions, which require attention in design to avoid disservices, manage false positives and so on.

False Positive Management

The automatic response is a two-edged sword because even if on one side can be useful to speed-up incident response and containment of threats, on the other hand, it can create a disservice if it is not well-calibrated and tuned. This is just because playbooks behind that automation are basically deterministic and sometimes can not design perfectly all the scenarios, for this reason, many times false positives may occur.

From the moment that is not possible to understand At Priori which situations can occur, response action should be designed also considering the false positives impact. For example, if the web gateway detection system detects an internal server that exchanges traffic with an external entity that can considered as a possible IoC concerning Intelligence Data we can block entirely the server. Assume that it is a domain controller, this means all related internal active directory operations. Unfortunately, it represents a huge cost to pay in case of false positives. Many times, services such as Active Directory are crucial business continuity aspects for a company and its productivity, and for this reason, models should be designed to keep into account also those kinds of considerations concerning the impact of a wrong detection (false positive). A good trade-off can be matched with a less restrictive action that considers also internal business continuity. From the moment that the malicious activity can be viewed as an external exfiltration instead of blocking the server, the orchestrator can simply deny that outgoing traffic blocking it at the network level, employing a Firewall or Web-Gateway for example.

By the way, according to best practices, the Domain Controller should be segregated into a private internal network and can be accessible through Privilege Access Management and ACL verification.

Within the actions taken for an automated response, false positive management must keep into consideration also the risk associated with the machine. Sometimes, especially for employees' devices, which represent a huge attack surface concerning well-configured domain controllers, they can be totally isolated from the network without incurring into the blocking of vital services. As was stated in the previous sections, XDR offers that kind of possibility.

Rollback and False Positives Management

In case a false positive was detected, a system like that should offer the opportunity to roll back operations as soon as possible. For this reason, a system like that must be implemented involving an action formal model as the playbooks do.

A formal model provides the possibility to develop solutions like that, and thanks to APIs which are designed to offer *UNDO* operations in evolving systems. Those can be viewed as an important aspect in designing automated systems compliant with business continuity requirements also in an unfortunate case of false positives. Obviously, to be consistent and to avoid this problem, the main requirement is to design and model a solution with a minimal internal false positives' ratio. To conclude the circle, false positives cases data analytics can correct the model in order to improve its accuracy.

1.6 Introduction Review

In this section will give a global view of the automatism that can be implemented inside a security perimeter to enhance its reactivity and proactivity aspects of the system following hints and requirements included in major standards controls.

According to the solutions deeply analyzed in the A, it has been implemented solutions based on a machine learning model able to provide a high accuracy rate in the detection of anomalies, malware, and phishing URL. Then, thanks to the API interaction is possible to create a specific model for the organization in order to automize the response system, based on security stack's components. This global system must have a low false positive rate not just in detection but also in response mitigation choice in case it requires choosing among different mitigations.

1.6.1 Problem Definition

The main issues related to this system are that there is no standard that deeply describes how that type of system should be implemented. For this reason, in the next chapter will be analyzed how penetration test remediation can be used to start a model to respond to a real attack in order to create and implement a model able to respond by itself to an attack. This evaluation will be made considering all the implications introduces in this chapter, furthermore, the benefit's analysis will be evaluated.

The process presented will be the inverse of the one presented in previous sections: instead of collecting information from penetration tests to proactively adjust the system itself, will be considered penetration testing assessment to recreate and implement a model able to act as a human probably do, minimizing false positives and disservices. This is the real essence of the purple team, mixing approaches, considered previously separated, in order to improve the global system in reactive operations. This is another main important point because reactive operations, as the word suggests, are highly influenced by time consumption. Speeding up the entire process will be viewed as a benefit with respect to old manual responses.

Chapter 2

Purple Team and SOC's Services

2.1 A Unified Vision

In this section will be covered several operational approaches on possible services offered by SOC in terms of visibility about the security requirements needed to implement what has been discussed in Chapter 1. This assumption is based on regular services that can be offered normally by a horizontal SOC: Penetration Test and Vulnerability Assessment as well as complete Incident Management and Detection Strategies.

Many times organization out of the ICT scope requires SOC's Professional Services to deal with standards impositions and business continuity. The reason behind customers' requests is that nowadays business continuity depends relevantly on cybersecurity posture. As it has been already stated in Chapter 1, many organizations go bankrupt due to cyber-attacks every day, and this fact is becoming an effective problem for organization's business, making them obliged to protect their assets and processes against cyberattacks if they want to survive.

2.1.1 SOC's Managed Services

For this reason, SOC was born: to create a business model based on security protection, and its scope is to lead other organizations through security hardening processes. Many times SOC can be integrated with other solutions, such as NOC and managed services able to give a 360-degree view of the entire network perimeter and IT internal processes. Understanding how core processes' lifecycle works is the starting point also for many cybersecurity standards explained previously. Among the services offered by SOC, there is also security architectural stack design, that is based on the requirements of the organizations or is gathered by performing assessments' driven architectural composition.

Knowledge, Compliance, and Performance

SOC organizations, designed to support the customer in all of their security services lifecycles, are basically formed by operators with IT background and knowledge in cybersecurity offence techniques as well as defensive and reactive approaches. The purple team aims to make those two fields converge in order to create a team with a unified approach to resolving issues and manage IT networks against threats.

The solutions implemented in both the Red and Blue Teaming approaches, intrinsically, produce a huge amount of data collected by audit, logs, detection system, and IRT reports, as well as be compliant even with the less restrictive standards. To produce and maintain smartly all of this data, can be useful not only for compliance aspects but also for machine learning and data analytics. From the moment that APIs integration can be a driven process in the composition of the security stack, many processes can be automated with basic programming skills. This explains the *"IT background"* requirements at the beginning of this subsection.

By the way, if the predictions extracted are coherent and useful in terms of cybersecurity aspects, the team can involve them to improve the global system in terms of Resiliency Score and Business Continuity. Many predictions can be automized with machine learning algorithms once understood the model behind it, giving birth to a system able to evolve with the dataset which in turn represents the evolution scenario and knowledge itself. Data now represents again its strategic role as a crucial asset on which an automated model can create and extract knowledge.

All the processes described match also the requirements presented detailed in the forms of Response Frameworks Controls and Operations. If SOC can guarantee established and solid detection systems according to ISCM definitions that will be provided in section 2.2, also compliance aspects can benefit from that improvement. The scheme behind shows as an iterative model can be implemented to auto-feed with local data, keeping updated also employing Threat Data provided by external sources.

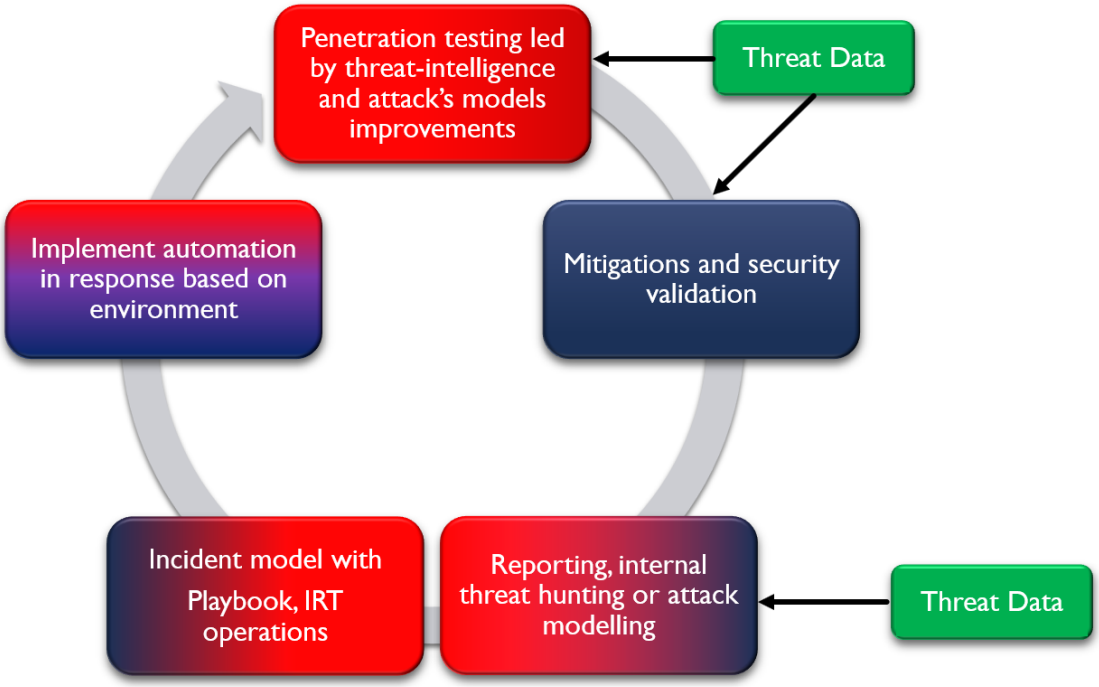

Figure 2.1. SOC's lifecycle

The figure shows a system able to exploit red and blue teaming operations in two ways:

- Red Team (Producer) - Blue Team (Consumer): Red Team is the security validator for detection and automated response system through testing operations described in NIST SP-800-53.
- Blue Team (Producer) - Red Team (Consumer): Blue team and intelligence involved to design attacks.

In support of those operations, there is also the Threat Data provided by Threat Intelligence and the community. A system like the one explained can also be able to address generic issues generated by penetration tests by automatic patching and device assessment as described in appendix A.5.

Threat Intelligence, CERT and CSIRT

Threat Data are all data related to some potential threats to an organization. Technically, the phase described in the previous sections is strictly related to the categorization of Threat Data.

As described in appendix A.3 the research and investigation performed on the raw sign-in data in order to detect a possible brute-force, represents the process on which, based on some evidence, the analyst can categorize an entity, such as an IP, as a possible IoC. The entity becomes a new Threat Data. Now threat data can be correlated to perform intelligence operations and prevent in a proactive way possible attacks.

Threat intelligence is the art of correlating threat data in order to detect and prevent attacks in specific business sectors. The main idea is to create a community where knowledge and experience are shared to increase awareness of some common attacks and trends. If companies A and B are operating in the fintech sector, and they are targeted by attacks designed by the same IoC (a malicious IP for example) and following a specific IoA (Index of Attack); the company C (which operates in fintech as well) may expect the same attack against itself. In a shared community the company C based on the evidence provided by the other two companies can act in a proactive the way in order to prepare defensive and responsive strategies to deal with those attacks effectively.

ENISA (European Network and Information Security Agency) is the top-level authority to fetch threat intelligence correlated data about threats in Europe. Given its crucial role in the European scenario, ENISA collects information from all the EU countries, furthermore also from other local top-level authorities such as NSA (National Security Agency) and CISA (Cybersecurity and Infrastructure Security Agency) which deals with the same activity in the United States.

In previous years, there was also an enhancement with some military organizations in order to deal with cyber-warfare operations. Latter is the example of the cooperation between NATO (North Atlantic Threat Organization) and ENISA [19] to share military intelligence related to cyber-warfare and hacktivism operations. This is a big step for the EU because involving NATO implies also cooperation and threats to data sharing not only with NSA and CISA but also with big companies such as CISCO, Microsoft, and AWS that can collect and correlate a huge amount of data, to detect not only new threats, actors, and vulnerabilities but also mitigations, patches and resolutions following the best practices. Those aspects become relevant in order to deal with the Ukrainian-Russian crisis to detect potential IoC related to Russia's attack against NATO's alliance.

All of that shared knowledge can increase in a relevant way the awareness of companies and organizations respect to the current cybersecurity scenario. By the way, an organization such as ENISA is not able to deal with all possible and national requests in entire Europe, for this reason at the local level, such kind of activity is managed using sublevel organizations such as CERT and CSIR.

CERT (Computer Emergency Response Team) and CSRIT (Cybersecurity Incident Response Team) [19] are two organizational model involved in national and local scope cybersecurity operations. Basically, they act coordinated by the ENISA at the local level, and in some cases, they can also provide consultancy service to improve the security assurance of an organization. The main difference is that CERT act at a more local level than a CSIRT. CERT has been involved in the process of improvement in incident response best practices according to local legislation and constraints. CSIRT is more operative and in some sense, its aim is to deal with operative mitigations and consultancy. Big organizations may have their own CERT to correlate security-related events and aware the entire community about the risks and some common situations. In Italy the most important CERTs are the following:

- CERT-GARR: CERT deals with security events on the national ultra-wide net GARR, involved in education and research.
- CERT-DIFESA: managed with the C4 command of the Italian Army and the Ministry of Defense, it handles security-related events in military and cyber-warfare operations.
- CERT-AGID: managed by the AGID (AGenzia Italia Digitale) is involved in monitoring, preventing, and responding to incidents in Italy.

Now the role of the CSIRT is of significant importance because it is the organization whose aim is to be the middleman between the private sector and the local CERT. In that way, CSIRT deals with a lot of operations including dynamic analysis of risks and incidents, incident response,

and the issuing of early warnings, alerts, and announcements. Its aim is the dissemination of information to interested parties based on risks and incidents. The most relevant CSIRT in Italy is the one managed by the CAN (Associazione per la Cybersecurity Nazionale).

From the moment that those actors are not able to collect and handle all the possible threat data in the world, other ′open source′ solutions based on the community feedback are developed. It is the case of the OTX platform (Open Threat Exchange) where reporters can report some malicious entity, and its score (or reputation) is based on community feedback. These platforms are widely used nowadays to know if an entity detected for malicious activity against specific organizations are already been reported by another member of the community.

Benefit

From the moment that reactive phase is implemented, the system can be led by Purple Team's developers in its first face to evolve into an automated framework able to respond to issues in terms of patching and incident response, potentially without human interaction. The benefits reported by those system has already demonstrated their strategic aspect from many points of view, among them the most important efficiency and response time with respect to human intervention.

Two approaches must collaborate among them to provide those types of flows, respecting compliance aspects and hardening the security operations. In the next section will be given further details about the relationship between those solutions and customers' needs. Finally, thanks to strategic support provided by CERT and CSIRT threat data about vulnerabilities, threat actors' operations, and attacks are always available, promoting collaborations and feeding those systems. The operations explained before follows the PDCA model, and they can provide security validation at each iteration. In the iteration *i-th* the cycle starts with penetration test assessment which dives visibility to security validation and improvements for patching, mitigations, and automation response operations implemented in the previous cycle (*th - 1*). This approach represents a real improvement to the overall system provided by a purple team managed services.

2.2 Customer's View

In the previous section, it was presented the technical improvements that can be offered by a purple-teaming-oriented SOC to exploit the powerful convergence between classical blue team and red team operations. The system described can address important aspects concerning continuous monitoring, security validation assessment and the efficiency provided by automation support. Assessments are enhanced considering also threat data gathered at the global level employing CERT and CSIRT organizations presented before.

2.2.1 Management and IT/OT Convergence

As stated in section 2.1, among the services offered by competitive SOC there is also another important aspect less oriented to security operations. Classical AD services, integrations, and network administration can be easily implemented through innovative systems and console to expand the view on the organization's infrastructure by the SOC. This became a baseline also in a highly automized environment such as an industrial system based on ICS and SCADA technologies. Attacks against those infrastructures can block entirely production and, for this reason, it is required to operate also on intrinsic security of those applications. In literature are reported many attacks against ICS infrastructure and some examples can be reported in the Ukrainian attacks actuated on December 23rd, 2015, where the Russian APT group SandWorm performed an attack against 30 substations: **7 110kv substations and 23 35kv substations were switched off, and about 230,000 people were without electricity for a period from 1 to 6 hours.** Attack has executed thanks to Black-Energy-3 Malware which works by means of HTTPS for connection to R&C server enabled thanks to phishing mail vectors, then the system seizing SCADA under control, remotely switching substations off.

In this case response's effectiveness and automation represents the last defence line against devastating disservices to power plant distributions. Those aspects can be also applied to other production fields such as automotive factories, petroleum refineries, and more in the general production plant of any type, from the moment that core processes are consistently automated according to IT innovations. These are the reasons why nowadays Operative Technologies must be integrated with Information Technologies to apply the same approaches (monitoring, response, auditing, reviewing, and assessing) not only to the internal network but also in a production system not related to classic ICT services.

Operative Technology

OT focuses on the management and control of physical devices existing and operating in the physical world. The control of real-world devices is as old as industry and manufacturing itself. The introduction of electronics and digital technologies over time also found plentiful uses in operational control systems, such as computerized numerical control machining systems.

While IT inherently covers communications as a part of its information scope, OT hasn't traditionally been networked technology, meaning connected to a larger network over the internet. Many devices for monitoring or adjustment weren't computerized. Those, with computing resources, are generally used closed proprietary protocols and programmable logic controllers rather than technologies that afford full computer control. The systems involved often relied on air gaps for security. As IT reaches more OT systems, air gaps can't provide adequate security for network communication and OT data. Organizations driving IT/OT convergence must educate and train staff to understand and implement adequate security. This often involves a mix of cross-hiring, cross-training, and close observance of regulations such as the GDPR, in the context of both IT and OT in terms of data security, privacy, and transparency.

The main differences between IT and OT are reported figure in 2.2, and briefly explained below:

- IT includes any use of computers, storage, networking devices, and other physical devices, infrastructure and processes to create, process, store, secure, and exchange all forms of electronic data.
- OT, traditionally associated with manufacturing and industrial environments, includes industrial control systems such as supervisory control and data acquisition.

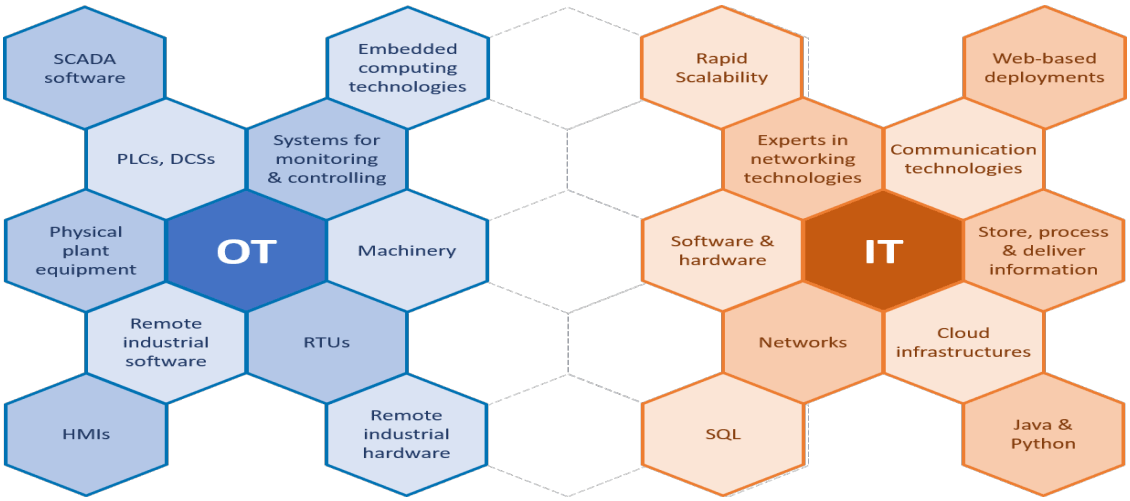

Figure 2.2. IT vs. OT

The grey area that connects IT and OT is the development and deployment of IoT devices. IoT devices include a wide assortment of sensors for gathering real-world conditions, such as

temperature, pressure, and chemical compositions. IoT devices also include an array of actuators that translate digital commands and instructions into physical actions, such as control valves and moving mechanisms. Each IoT device is designed to communicate over standard networks, allowing them to exchange OT data with IT resources, sometimes over considerable distances.

2.2.2 ISCM over ISMS Review

Now that customers' operations and needs are described it is possible to understand how the Information Security Continuous Monitoring baseline can be applied to an Information Security Management System highly focused on automation. NIST Special Publications documentation can offer many important hints and considerations to keep into account during the designing of those systems.

ISCM Automations

According to NIST SP-800-137 [20] which describes operations adopted by ISCM for Federal Organizations: *"Maintaining ongoing awareness of information security, vulnerabilities, and threats to support organizational risk management decisions.* ***Note: The terms "continuous" and "ongoing" in this context mean that security controls and organizational risks are assessed and analyzed at a frequency sufficient to support risk-based security decisions to adequately protect organization information".***

The same Special Publication provided by NIST affirms in section 2.3 that *"Automation serves to augment the security processes conducted by security professionals within an organization and may reduce the amount of time a security professional must spend on doing redundant tasks, thereby increasing the amount of time the trained professional may spend on tasks requiring human cognition."*

Those two affirmations reported by NIST are perfectly compatible with the consideration exposed in the first section of this chapter, proving again the global system's compliance not only regarding NIST assertion but also with PDCA models and improvements widely referenced during the artefact.

ISCM and ISMS Integrations

In the next subsection were presented principal aspects to keep into account in designing detection and monitoring strategies effectiveness. Those solutions concern all the aspects questionable by a *"How?"* statement, because guidelines as the one proposed by NIST put basics to implement a security system concerning frequent assessment and continuous/ongoing improvements.

By the way, there is another important question to answer, and that is the *"What?"*. What a system must monitor to be effective is specifically described in the Information Security Management System as the one proposed by ISO27001. Specifically, those aspects concern all the requirements defined in Chapter 1 regarding the IS27001 Annex. Following the kind of requirements and notions described in ISCM, there can be the possibility to create a system like the ones exposed before able to improve its security posture in terms of assessment frequency and automation to speed up response operations.

The last step which is the most expensive in terms of effort in designing the solution will be described in the next section and implemented in the Appendix A. It concerns the practical integrations between these two worlds in order to design the entire system based on the assumption provided by many standards as ISMS by ISO27001, ISCM by NIST SP-800-137 [20], and finally all the operative procedures described in Purple Teaming.

Chapter 3

From Offence To Defence

3.1 VA-PT Mitigations

According to the requirements and definitions that are given in Chapter 2, it is needed an automated solution to respond to malicious scenarios with efficiency and effectiveness. Efficiency aspects can be highly improved by exploiting AI integrations and automation provided by Software tools in order to reduce all human interaction in well-defined and structured situations. Effectiveness instead, can be guaranteed by continuous assessment and validations of what was considered resolved.

Validation is another important aspect that will be covered in this chapter, and basically, it consists of demonstrating, through test solutions as exposed in the NIST SP-800-53 [14] assessment for the controls, that the current status is now patched mitigated respect to the previous one. In this section, the artifact will focus its attention on the first big set of mitigations that are concerning the Red Teaming operations.

As we introduced previously, there are many ways to mitigate a vulnerability detected in VAPT assessment, but nowadays, the most important are the ones correlated to automation and integrations with a basic asset inventory and patching systems as required by several standards [21] to guarantee global visibility of the system and all connected devices. After a Penetration Test and a prioritization pipeline for the vulnerabilities resulting in the assessment, it is possible to interact with those systems to take action. Many times, almost the totality of vulnerabilities found in the system concern out-of-date components or misconfigurations. Based on those assumptions, it can be simple to exploit asset inventory and Mobile and Device Management features to update consistently the system and trigger also applications update. This is due to the fact many times employee does not care about those type of issues on their organization's devices, and this fact makes life simple to attackers who may exploit those kinds of vulnerabilities to take initial access to the internal network.

But how it is possible to formally describe mitigations based on some VAPT results? Playbooks can be the solution, and they can give a huge benefit not only in automated mitigation but also in describing and modeling a peculiar system. In the next subsection will be presented all their benefits and relevant features.

3.1.1 Patching Benefit and Strategies

Gaining access to valuable data resources from a compromised host on the network is often not difficult because people have not fixed things properly, leaving a bug relatively easy to sniff out and trace. Problems occur when machines are not registered correctly or users want not to restart them; you can force a restart, but that usually annoys users, and it is completely impractical on servers. Then you may find machines that you thought were repaired, but are actually quite outdated.

For this reason, nowadays, there are many automated ways that offer the possibility to automatically track and force patching and updating on systems widely used.

Nowadays, many organizations use production tools such as Windows OS, Office Adobe etc... Those systems can be automatically updated following a hierarchical strategy starting from OS's version and then pass to applications. Microsoft provides many solutions such as WSUS (Windows Server Update Services) utilities in order to handle update aspects for Windows-based servers in order to guarantee regular patches and updates. In the same way, Microsoft provides also a tool to handle personal devices such as Intune and Microsoft Endpoint Configuration Manager. Again all of those systems can be joined into a unique console known as SCCM (System Center Configuration Manager) able to give a unified view of the patching state in the system.

The Patch Management provided by NIST [21] lifecycle can be described by several states to guarantee the effectiveness of mitigations derived from patching:

- **OT Asset Inventory**: Important aspects concern understanding what assets are composing the network, where they are located, and what software is running on them. This census can be done by means of agent-based solutions, but it is difficult for IoT devices with limited computational resources, where an agent-less probes are preferred also for ICS/OT networks.
- **Identify Vulnerability Information**: The second challenge is the necessity to monitor which patches are available and required for all the devices visible in asset inventory. The core components of Windows, Linux, Unix, Office and other products like Adobe are straightforward to keep monitored. For this reason is useful to identify assets as well as approved software that can be installed on the network's machines in order to monitor specifically for updates. Operators need to research patches to determine what, if any, security components are addressed.

  The sheer volume of these apps makes the task exponentially difficult and for this reason if well-designed this can be a good candidate for operations to be automized. In addition, patch availability is only half of the equation. Effective patch management requires robust vulnerability assessment capabilities. Traditional IT tools with scan-based approaches are not effective and/or safe for OT/ICS systems due to the sensitive nature of the devices and their internal firmware.
- **Identify Vulnerability Relevancy**: Now the system has to identify which assets must be patched based on OSs, versions, applications and most important, vulnerability assessment results. Again those types of aspects can be automated by means of filtering processes that are significant to speed up the analysis of which patch is required and on which systems.
- **Review, Approve, and Mitigate Patching**: Once identified the devices to patch are, there is the necessity to implement mitigations approved by vendors. Sometimes especially in 0-Day attacks, the vendors suggest a workaround while the patches will be designed and developed by the manufacturer itself. For this reason, also that kind of considerations must be taken into account when approved patches are available in order to undo the workaround and apply the mitigation in the regular production environment.
- **Test and Deploy Patches**: To implement patches one of the best approaches is to split the device to patch into a small set to validate the effectiveness of the patch, then if results are consistent with the one expected the process can be applied to all other devices within deploy mode. This is an important aspect the moment that, if an organization implements an ineffective patch on all devices, then they fail to prove the mitigation provided, the organization must undo the patch in all devices with a huge waste of effort.
- **Profile and Improve Patching**: As widely discussed, at the end of patches cycles there is a final validation and review in order to improve internal patching management according to the PDCA model.

As the lifecycle reported in image 3.1 shows, good patch management is able to cover other useful aspects for an ISCM such as Asset Inventory needed to understand which patch must be applied based on system composition.

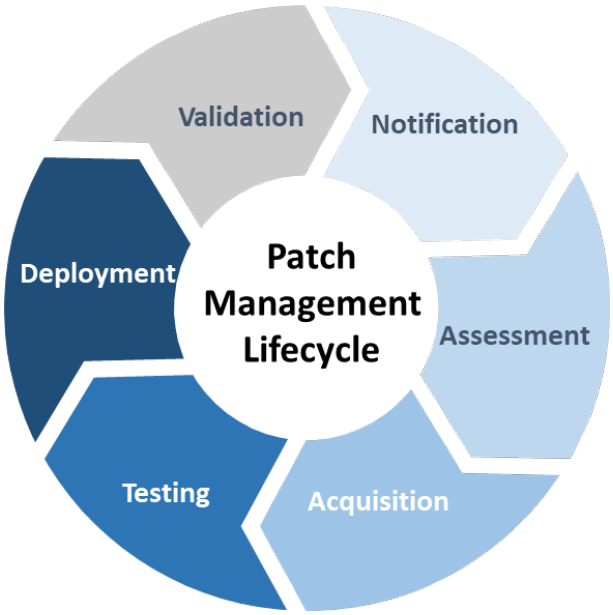

Figure 3.1. Patch Management - Lifecycle

In addition, patch management takes into consideration also Vulnerability Management and Vulnerability Assessment frequency and their results as well as internal configuration management. Configuration management is a basic aspect to guarantee that the internal component of an organization work according to a well-known and expected behavior, appropriately by a specific configuration. The next image shows dependencies to perform effective Patch Management:

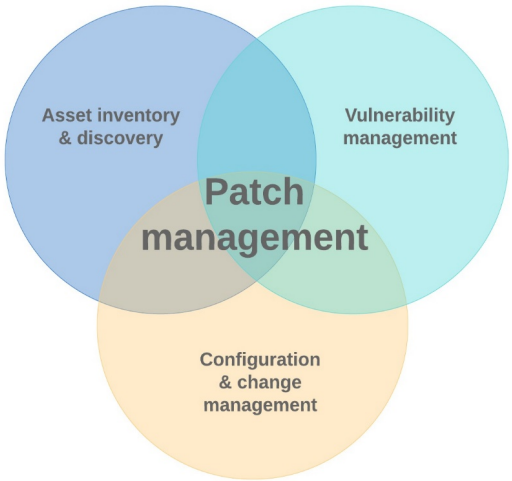

Figure 3.2. Patch Management - Dependencies

3.1.2 Variability and Further Considerations

In the previous introduction, it has been presented basic aspects concerning the patching in Microsoft environment. By the way, there can be situations where there is also another type of complexity in the network, and from that, those devices increase the attack surface they must be managed securely as well as Microsoft devices. Those devices are referring to Unix-based ones, which must guarantee patching and update at the kernel level and packages used by applications. In a Unix-like system, a good alternative can be viewed in the solution offered by Red Hat or other private competitors that try to integrate all the systems described before are in a single console which must give visibility on the security posture of the system based on the results gathered from the assessment such as Vulnerability Assessment and Penetration Test.

Furthermore, there all the IoT and network devices must be updated at a firmware level to guarantee legal behaviors, and operations. Those kinds of devices represent a vital subsection of infrastructure which in case of compromise due to out-of-date firmware exploitation can result in a complete block of production. At the same time, firmware is always viewed as not useful and many times it is not aware by the manufacturers themselves which may be public the new firmware on its website, without informing CERT and CSIRT and threats sharing organization.

Finally, the important aspect is that visibility of patching status and actuation's a complex organization is more difficult, and it must not be undersized. To resolve those aspects, a compliant system has to track all the ICT, IoT, and network devices present in the network in order to unify a common console in order to have global and detailed situations concerning the patching status. Starting from this visibility aspect can be possible to implement automated solutions specifically designed for each OS or firmware architectures.

3.2 Playbook and Validations

A playbook can be viewed as an execution flow model to be implemented in case of a specific vulnerability is found as a result of penetration test and vulnerability assessment. Many times playbooks are designed in a similar way to flow charts but instead focus on classic programming statements as assignment objects creation, they work on well know building blocks based on the architecture. They can be considered as high-level phases consolidated and working for this specific system by means of API integrations, script integrations or simply more general automation.

For this reason, playbooks were originally used to resolve networking issues or configuration, describing step by step all the actions to take to evolve through target states. For this reason, playbooks were born in the ICT scenario as a checklist to follow in order to implement strategic configurations in networking systems simply, and more important can be used as many times as if the preconditions permit its applicability.

3.2.1 Playbook and Cybersecurity

According to the aspects introduced before, playbooks are designed to evolve a specific system from one status to another one in order to implement what the playbook is designed for. This definition can be viewed as key in modelling the mitigations described in the major standards and concerning the improvements that a good ISCM must periodically perform. Playbooks are formally viewed as a list of steps to implement that can be re-designed to be applied to cybersecurity scopes.

To do that, there must be several pre-conditions that are useful to identify their applicability, avoiding implementing wrong features or degrade the entire system which can have the opposite result.

The consideration reported before are not only applicable to mitigate VAPT issues where at 98% of cases playbooks are limited to perform components updating and misconfiguration resolutions; they can also be applicable, in a more complex way, to incident response operations in order to automize processes as required before in the global system descriptions. In this way, the playbook's design is more complex because pre-conditions for an effective application can

become very challenging to identify generic situations taking into account relevant aspects and minimizing the false positives rate. Concerning the example made in the section 1.5.4 about response automation, playbooks can be used also to create an automated incident response line that is triggered after an alert is generated by the detection engine.

At that time the system has all the ingredients to respond consistently to an incident because he knows the incident and the threats behind this scenario, as well as a playbook response action to implement to mitigate, eradicate permanently the threats restoring a benign context. Then there are two opportunities: the first is to react to the incident, following manually the operations listed in the playbook otherwise, where possible, the playbook can automate all the operations by means of local security products and configurations.

Playbooks can provide operations concerning different phases of the NIST RMF discussed in the section 1.4.4 and for this reason, can be useful also for a junior analyst to have a formal model to handle security incidents and scenarios. Playbooks are an important knowledge baseline that can be used to train human operators even under a governance and compliant aspect of incident response processing, as well as a machine learning model to implement an intelligent system able to manage the incident in an automated way.

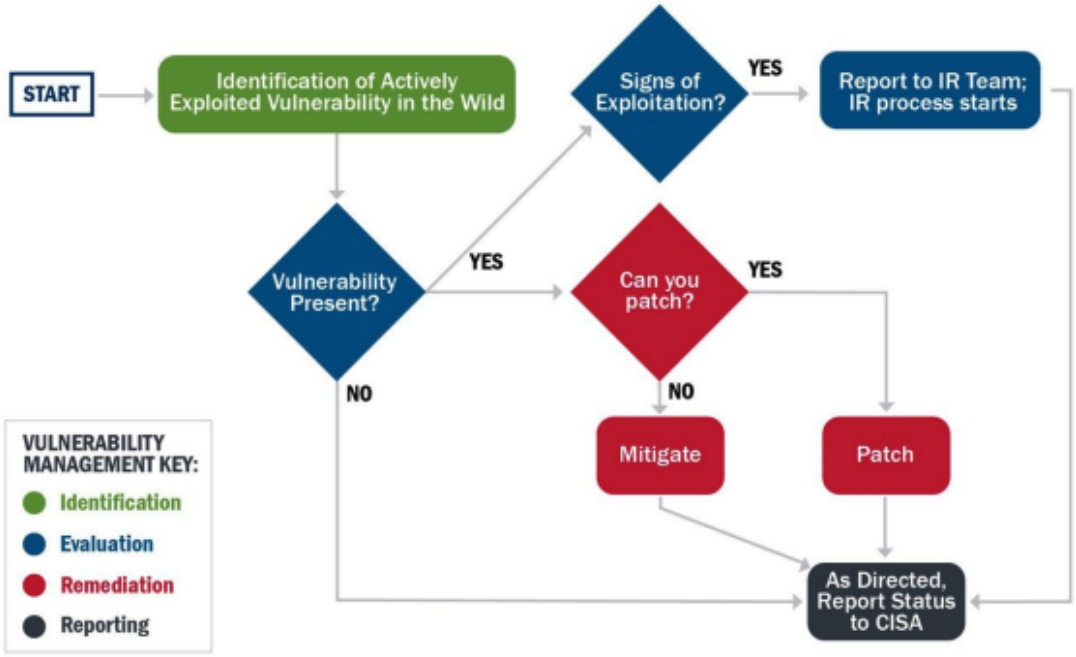

Figure 3.3. Playbook

As the example reported in image 3.3, integrations among Intelligence, Blue Team in detection and monitoring as well as Incident Response Team in order to understand how to respond to a specific vulnerability, again can be formally modelled through a playbook.

As stated before its operational view remember the one of a flow chart, but the main difference is that each block is more complex than a single instruction or simple *IF Statement*. In the example, the conditional block *Vulnerability Present?* is actually a VAPT to understand if the vulnerability can be viewed as a real threat to the organization and to evaluate a possible impact for it as well as design a risk management program sized for its interfacing with the incident response team. If the threats are critical, IRT operations can even block the machine and set the intervention response time as little as possible. In this system, a playbook is a key point in order to automate and respond formally to each possible threat in a coherent way, and for this reason, a playbook has to be kept updated, available and most important protected. At the same time playbook addresses also documentation aspects declared into major standards, they can provide a highly formal structured document to respond to incidents and to manage the escalation process in case of relevant security incidents.

3.2.2 Playbook Management

In the previous section will be presented the support gained involving the playbook and its benefit in designed ISMS. By the way, there is the necessity to keep logic and hierarchy for internal playbook management in order to have a global view of the new playbooks as well as consistently update the existing ones.

New scenarios often present relevant importance in describing new threats, attacks, and techniques; for this reason, can be exploited all the benefits provided by an IRT to handle new cyber-attacks. After the security event handling, can be used the Post Incident Phase review in order to design a new playbook, understand also the modus operandi of that attack, and optimize based on the IRT human response with critical opinions. The entire process can be adapted to be integrated with agent and API. These important operations can have a doubled benefit: first of first playbook's tasks can be now used to automize Incident Response more efficiently; furthermore, based on the attack review is possible to design an attack pattern to validate the effectiveness of the playbook. According to the model shown in section 2.1.1, is possible to introduce new data sources to design the playbook model and consecutively the entire automation of the response, based on the attack handled as already stated.

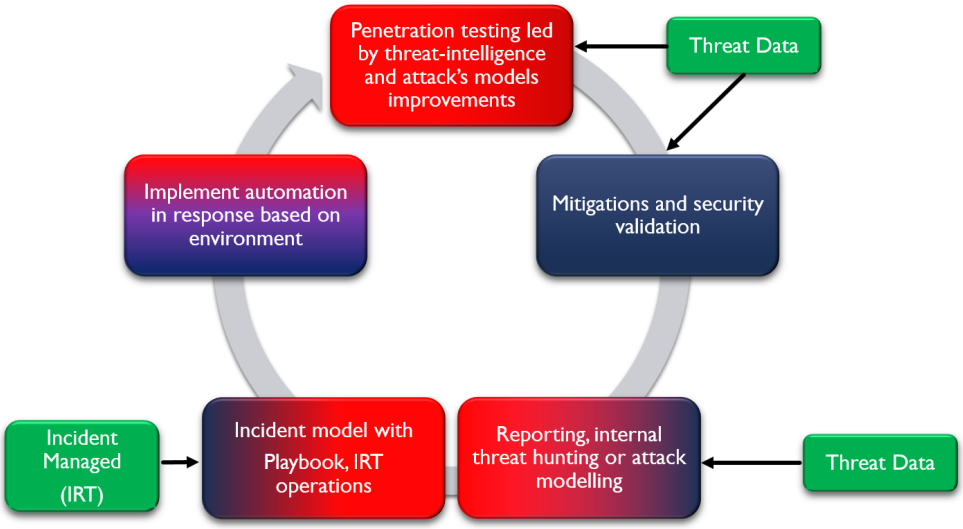

Figure 3.4. SOC's lifecycle

Playbook must be kept also updated in order to take into consideration and improve use cases or boundaries conditions that may escape from the previous review sessions. In this case, an update can be easily validated using penetration testing or specifically designed threat-hunting operations in order to evaluate internal reaction and, more important, that response automation works well for the designed security event. Playbooks' update is also useful to model attacks in all their stages and to understand how complex operations work and evolve over time.

3.2.3 Validation Methodologies

As stated in the previous section, now a SOC can provide an initial assessment to understand the vulnerabilities and internal security posture of the organization. The operation can be performed through Risk Assessment with VA-PT. The organization based, on its internal complexity can reduce the attack surface by integrating a patch management system able to provide a global vision of what is patched and what is not. To do those operations there are many considerations to keep into account where the most important, as discussed before, is the variability of the environment.

After a primary assessment, with a good security posture is also possible to integrate those systems with some detection engine and Incident Response tools such as NDR (Network Detection

and Response) or XDR as well. In case of detected attacks, the IRT eradicates threats and restores normal operations, analyzing response efficiency and tasks in post-incident activity. At that time there is the necessity to implement a playbook and mitigations that will be discussed in the next sections in a detailed way.

Now the system is completely integrated with an automated playbook but how to test the real effectiveness of target status concerning the previous one? Playbook effectiveness can be tested and validated with red teaming assessment: the main idea is to reproduce the attack operations to stress and evaluate the response of the entire system within playbook integrations. The attack can be simulated very close to the original one thanks to detection logs and post-incident activity where, as discussed before, the attack's techniques are understood and modelled.

Then the simulation can be planned to verify the playbook behaviour, in case of problems or unexpected behaviour, the playbook model can be tuned in order to address the issues occurred simply by repeating the assessment phase. By the way, during deployment, there may be many playbooks and attack models to assess, for this reason, many times more than one playbook are stressed into a single assessment considering also the disservices that may be issued.

To complete the playbook's validation effectively, the best approach requires implementing the playbook following a manual interaction, then only when the playbook addresses all the security events that it was designed for, can be automated considering the system specifications. So there are two assessments to implement a good automated response. The first is to model the best playbook and then validate its behaviour against threats, finally, automation can be implemented and another assessment validates its reactivity, analyzing the effective speed-up in the response phase.

The next image shows how the playbook's validation can be implemented considering also the management aspects discussed previously. Three main logical threads can be implemented by the Purple Team members: Playbook definition, automation, and deployment as reported in the figure 3.5.

As the flow chart shows after the Post Incident Activity, one team can create the playbook, the other can design the APIs and infrastructure resources involved in the automation of SOAR integrations, script XDR's APIs, and many more based on the environment. The design aspects in both cases keep into account the impact in case of false positives and so on. Commonly, those assumptions are considered in the playbook's design from the moment that it is already known possible target of the attack is analyzed before. Finally, if the playbook behavioural and automated validation pass the assessment, they are deployed and evaluated in the real attack world. If attacks or boundaries condition do not work, the playbook is re-examined and improved, then validation will consider all the issues related to this process.

A process like that permits keeping into consideration also aspects exposed in ISCM, playbooks are then periodically tested by the assessor and by real-world attacks statistics can be continuously monitored for their real effectiveness.

3.3 Playbook Implementation and Validation's Automation

As stated in previous sections, playbooks are the key to integrate formal models, efficiency and adaptivity inside complex systems that require automation for incident response basic operations.

In this section and the following will be presented all the technical aspects encountered, to keep into consideration while designing an automated response system. So, the point now is that, supposing to have a well-calibrated detection system able to detect each security event in the organization, the system must respond also to those events when alerts are triggered, considering all the aspects such as the impact of the event, the business value of the target asset and false-positive costs for the action taken.

Again will be exploited relevant and common features offered by the automation tools for red-teaming operations such as automated vulnerability scanner and AI-based Penetration Test Framework as the one presented in the previous sections.

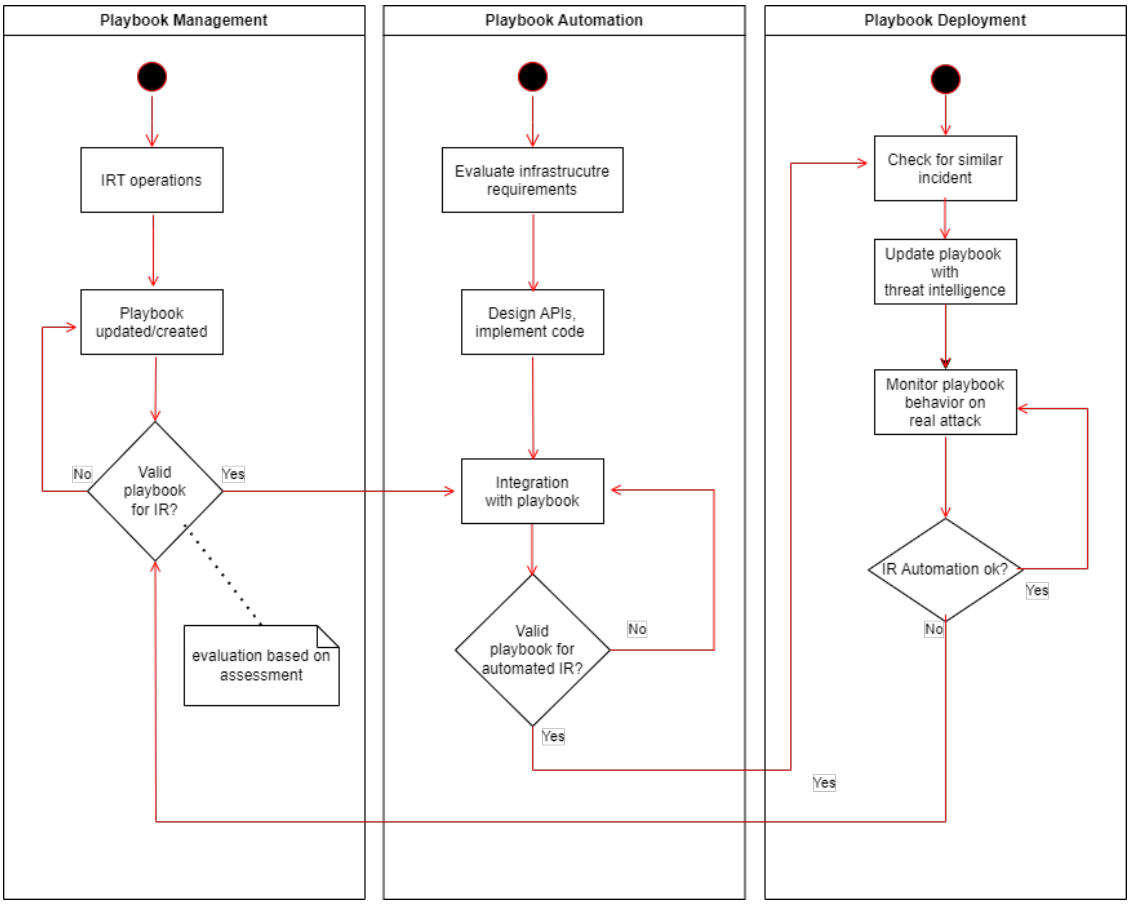

Figure 3.5. Playbook validation process

3.3.1 Blue Team and IRT Reaction

It is important to keep into account how the infrastructure is designed to exploit the security features already available in the system. From the moment that the Detection System offer many entities to monitor the internal network and devices we assumed to have in the system the following solutions:

- **XDR and NDR**: They are two basic systems to detect malicious activity in case of attack. XDR and NDR are able to give global visibility both at the device level and network level, in addition, they exploit massively machine learning-based solutions from the moment that has visibility of a huge amount of data. Specifically, XDR can view all the local machine logs such as windows logs while NDR can access all the network's traffic, classifying events and evaluating if they can be considered legal or not.

  Keep into account that, concerning XDR, NDR must be designed and located strategically: designed in order to support the close-to-real-time processing of a huge amount of logs, avoiding shadow log in case of under-sized sensors. At the same time, there must be a sensor that will interact with the main brain which must be located in a strategic network location based on topologies to have a segregated collector node.

  Another important feature is that nowadays many XDR permits actuate basic response action, further offering a huge set of API endpoints for the interactions.

- **SOAR**: As already presented SOAR can become the central node of computation in order to actuate the playbook. Modern SOAR allows integrating automated responses with a custom script, those kinds of platforms are by default integrated with a common interpreter such

as Python or Javascript. This is a relevant fact from the moment that they can potentially implement whatever a developer can produce using the supported programming language.

Furthermore, SOAR's functionalities can be extended with specific plugins that many times simplify drastically the development effort, allowing a faster deployment, which is another important benefit given by these advanced systems.

- **SIEM**: As a milestone of the detection engine there is also the necessity of SIEM to guarantee a correct global visibility, log collection and storage as specifically required by Annexes points concerning collection, storage and securing logs. Logs are important also to reconstruct the attack and understand how to model it to validate playbooks for the incident response.

  Understanding how attacks work is one of the most important operations to perform in the Post Incident Activity. Many times SIEM collects also remediation action and this aspect gives the possibility to understand which patterns are behind an attack, creating explicit policies to trigger alerts and at the same time link them to those a corrective playbook. Logs and responses among systems rigorously monitored by a SIEM can be also a validation check that the playbook is applied correctly.

- **Network Perimeter Defense**: Many times response correction requires blocking traffic, logically isolating devices from WAN as well as from the inside, within the LAN.

  From the moment that Perimeter Security System is the one which decided if a connection can be propagated among several networks, governing and automizing its reaction, represents the baseline to implement a consistent and effective automated reaction system.

- **Back-Up and Recovery System**: In this artefact will be also briefly discuss the integrations with backup systems and recovery systems to support resilience and business continuity with backup and recovery strategy in case of disaster. The basic aspect is that the backup system can be scheduled to get snapshots of the device images mounted on it. In case of a huge compromise, there may be the necessity to restore a working image on a corrupted device: for this reason, An automated backup system can again improve the global system resilience score. Snapshots must be scheduled at a reasonable interval time, considering the retention and air gap in the entire Storage Area Network where image databases are contained.

  In case of an attack, the system will select the last image not affected by malicious activity, and it will proceed to replace the compromised one with a legal one. Finally, that kind of solution permits also the detection of problems analyzing direct incoherence in data as IBM provided in its suite of products with the safeguarded copy (copies are encrypted to be compliant with major standards).

The image 3.6 shows how Detection and Reaction components can be integrated with this complex system, the image does not consider network defence that may be included in the object governed by SOAR in automation and response.

3.3.2 Red Team and Threat Hunting Reporting

In addition to classic Blue Team operations concerning Detection and Response actions, there is also the necessity to integrate the vulnerability scanner reports and the patch management system to implement all the aspects concerned in section 3.1.

As stated, the key in this operation is to have a well-structured report that permits to extract of all the critical vulnerabilities detected, the data will be then parsed and linked to their CVE. By the way mitigations, especially during the first assessment must be performed coherently, so that all the critical vulnerabilities can be patched earlier. The prioritization in this case will be made specifically based on the CVSS score related to the threats, but many times this is not a good indicator. The problem behind CVSS score is that it is a static indicator of impact, by the way, vulnerability scanners are not able to provide enough information about the effective

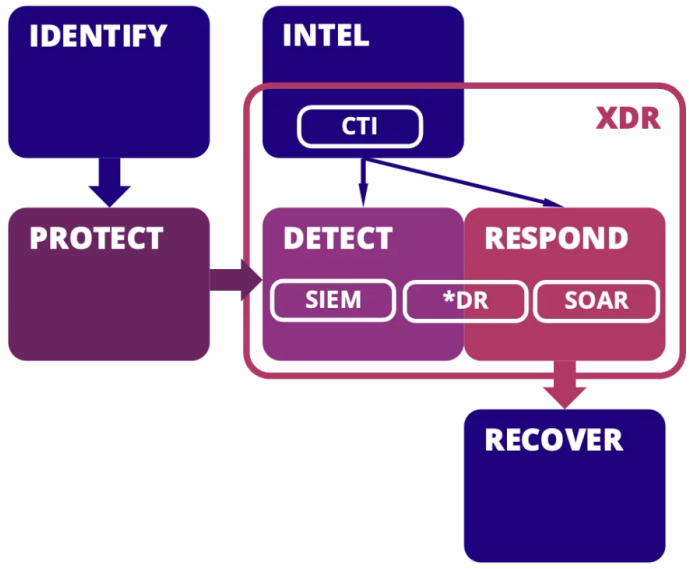

Figure 3.6. Environment For automated Detection & Response

exploitation of the vulnerability itself. This evidence is evicted from penetration test assessment, and it gives considerable information on which produces a coherent prioritization queue.

Reports produced in this phase become the most important data source to highlight flaws and correct the system, in addition they are the snapshot of the current status on which they will be designed the target one. Then based on the new reports (and consecutively the system status) is possible to implement the Test validation between the two states to verify that the new status corresponds to the one designed as a target before taking corrective actions.

3.3.3 Prioritization and Queue Evaluations

The basic idea behind that kind of evaluation is to assign a coherent value to a single vulnerability after the result based on the assessment performed before. The main considerations to keep into account are the following:

- **CVSS**: the CVSS score can understand the potential impact of the vulnerability based on loss of Confidentiality, Integrity and Availability and other considerations as described in version *3.x*.
- **Involved Asset Business Value**: another important aspect to keep in consideration is the business value of involved devices. In a more granular way, there can be impact indexes based on classic CIA parameters and the finest evaluations can be produced to keep into account CVSS details and impact for specific Information Security features Confidentiality, Integrity and Availability.
- **Effective Impact**: as stated a vulnerability can be assumed as malicious especially when it can be exploited. This evidence is given by Penetration Testing which can produce an effective report of which security features are bypassed.
- Number of Occurrences: another important aspect is to keep into consideration also vulnerabilities present on many devices. In order to prioritize the patching even on those aspects, and apply the recurrent patch to a big set of assets.
- **Vulnerability Correlation**: to optimize the patch management is also important to consider if there are situations where a vulnerability can be addressed by patching other vulnerabilities. This is the most difficult aspect to evaluate and for this reason, an automated

solution can offer support in evaluating all the possible patching paths, looking for the most efficient.

Now is possible to keep into consideration all the qualitative aspects described to give a unique numerical (quantitative) value to the vulnerability score to prioritize the patching. A possible solution is to multiply the CVS Score by a factor based on the percentage of assets affected by that vulnerability. Then the results obtained for each vulnerability are sorted to generate the prioritization order. In that case, must be kept into account the CVS score and the frequency of the vulnerabilities.

By the way, a more complex model can be created starting from the previous one and considering also CIA suffering impact for the specific asset and the risk score provided by the CVE. Those type of considerations are more difficult to implement because it needs all the impacts concerning Confidentiality, Availability and Integrity for each asset, and many times is the result of a preliminary assessment. A possible form can be:

$$P = \frac{m\ cvss^k}{m\ cvss^k + N} \tag{3.1}$$

Where:

$$m = \frac{asset_{AFFECTED}}{asset_{TOTAL}} \tag{3.2}$$

$$N = Asset's\ Impact\ Score\ Value > 0 \tag{3.3}$$

and

$$k = \begin{cases} 1, & \text{severity is LOW} \\ 2, & \text{severity is MEDIUM} \\ 3, & \text{severity is HIGH} \\ 4, & \text{severity is CRITCAL} \end{cases} \tag{3.4}$$

based on CVE details.

In that way is possible to keep into account many factors as the severity of the alert, the percentage of assets affected by this vulnerability and the impact score value that is a constant to keep into account the severity of CVE and risk evaluated for the specific assets affected by that vulnerability.

N is tricky to evaluate and for this reason, will be supported by the computation by python code to explain better its behavior. Assuming to have X assets and their risk values for Integrity, Availability and Confidentiality. All of those assets are affected by a vulnerability and their CVSS score, in addition, available information about Integrity, Availability and Confidentiality severity level. It is possible to define N as follows:

```
N = 0.00
for an in assets:
    if a['integrity'] <= cve['integrity']:
        N += 0.33
    if a['availability'] <= cve['availability']:
        N += 0.33
    if a['confidentiality'] <= cve['confidentiality']:
        N += 0.33
return N
```

Basically in that case N considers in a weighted way all devices in which the CVE presents a non-tolerable risk level. From the moment that N is always greater than zero, it verifies that the equation will not degenerate into a constant solution as in the case of N = 0. But the real importance of N is given by the fact that it is in the denominator and for this reason, it plays against growing of prioritization final score.

If there are few situations where risk is not tolerant will be generated a score lower than a situation where more situations are observed. Finally, this variable keeps into consideration CVE

and asset risk levels. Instead, k, is the polynomial exponent able to describe different global scores for the CVE, at the priority of assets and CIA impact levels. It keeps into account other measures which determine an increase in the global CVE score such as *scope* or Attack Vector, follows an example to demonstrate what has been described.

```
CVSS:3.1/AV:N/AC:H/PR:L/UI:N/S:C/C:H/I:L/A:L -- 7.7 (HIGH)
CVSS:3.1/AV:N/AC:H/PR:L/UI:N/S:U/C:H/I:L/A:L -- 6.1 (MEDIUM)
```

As we can see in the two vector string they are the same except for scope (S:) where changed (C) increase severity to HIGH while unchanged (U) keep it on MEDIUM.

By the way, there is another important aspect to keep into account which is the reliability of the risk assumed against the vulnerability found. If the assumptions are based on a simple vulnerability assessment is not a good way to implement that system. It is as if the system was oversize because there is no proof concerning the effective exploitation of the vulnerability.

Many times for several reasons a vulnerability cannot be completely exploited by an attacker, and penetration test reports and metrics must be involved to evaluate the real impact of an evaluation method like the one presented before. At the end of this evaluation will be possible to extract a report like the one presented below:

Now based on that sorting order is possible to patch vulnerabilities following specific criteria. The resulting extract will be injected into a patch management system. The operative flow is the following:

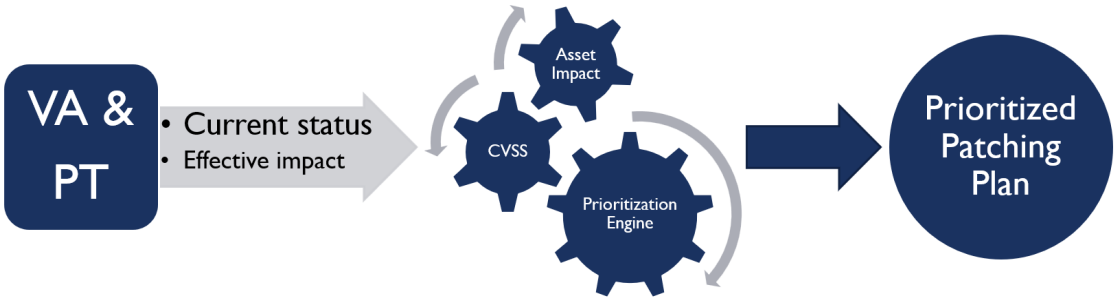

Figure 3.7. Operative Flow

Finally, the last step is to use some integrated system as Kaseya or SolarWind to implement the automation exposed before and patch many host from a unique console. Pay attention that this system must be intrinsically protected because in case of a compromised supply chain an attacker can propagate malicious resources inside the entire system from a single component. This is the case of a supply-chain attack that nowadays becoming a real problem from those type of architectures, in fact, the providers stated before has already suffered this type of attack with a huge impact on their customers.

3.3.4 Playbook Creation and Use-Cases

Now present all the ingredients to apply mitigations and patches based on the effective impact concerning a specific attack. But in this artefact are highlighted two main operative scenarios:

- VA-PT: evidence is provided by red teaming assessment to understand real criticality in the system. They can be prioritized based on the effective exploitation score evaluated in the penetration testing. Critical CVE does not mean for sure that the system is effectively vulnerable. To understand this fact a finer level PT assessment is required. Again is highlighted the key role of reports and evaluations performed.

- IRT: evidence is provided by Incident Managing, and intrinsically, they can map attack phases thanks to a detection system which matches the well-known pattern and internal security event. In addition, the response can be formally described as well as from the moment that is set of sequential actions is taken by a human.

At this point, a playbook can be created as a consequence of events that are observed and the IRT expertise offered by SOC services. Furthermore, as already discussed, CVEs can be mitigated basically in terms of updates and configurations corrections that many times are intrinsically resolved in updates. Now are required to design a centralized system able to implement both, interacting with an automated patching system and the security stack's components in order to automate all the operations.

CVE mitigations and automations

In order to deal with CVE, many solutions can be useful when a CVE is not able to provide an effective resolution implementation. Follows a basic example which reports a high vulnerability reported by Microsoft in its Microsoft Windows Support Diagnostic Tool. This is a critical vulnerability (*CVE-2022-30190*) because can be exploited to remote code execution impacting entirely CIA features and for this reason its CVSS score is 9.3. As stated before, MS updates the security issues exploiting the system update services inside Windows hosts.

By the way, Microsoft's website reports all vulnerabilities affecting its system and others widely used software. This is the revision tab for such vulnerability:

VERSION	REVISION DATE	DESCRIPTION
2.0	Jun 14, 2022	The update for this vulnerability is in the June 2022 cumulative Windows Updates. Microsoft strongly recommends that customers install the updates to be fully protected from vulnerability. Customers whose systems are configured to receive automatic updates do not need to take any further action.
1.1	Jun 3, 2022	Added an FAQ. This is an informational change only.
1.0	May 30, 2022	Information published.

Table 3.1. Automated Incident Response Global Stats

As the version indicates, there can be different situations: CVE is published but no actions are available to address it (Version 1.0), then there may be a phase of workaround that can mitigate the issues (Version 1.1). Finally, the official patch is released *"to be fully protected from the vulnerability"* as Microsoft stated (Version 2.0). This approach can be mapped with several types of vulnerability disclosure.

In that case, it is important to understand three cases of a disclosed vulnerability on which select the automations' behaviour:

- **0-Day CVE**: The vulnerability is disclosed but no patch is available when CVE is published, it can be viewed as Version 1.0 in the previous table.
- **CVE Without Official Patch**: CVE is published, there are mitigations based on a workaround but no official patch or security updates are available, it can be viewed in Version 1.1.
- **CVE Responsible Disclosure**: CVE is released with official patching references and version updates, this is Version 2.0.

For this reason, an automated patch managing system can be designed to handle simple situations automating patching Responsible Disclosures and checking for updates or replacing asset software with the newer version. Only when there is a critical situation, does the automation need to alert the operator that will handle manually the CVE. In that case, the operator can implement a workaround and the automated system will keep them in account to assume the

CVE is partially resolved but continuously check for updates in CVE's revision. When an official patch is available, it will inform again the operator that will dismiss the workaround in favour of official patches.

Then as stated before, the system will perform validation to verify the patch states and that has been applied correctly. At that point, the CVE can be assumed as completely mitigated and patched.

IRT: Mitigation, Eradication and Recovery

The previous section issues the operational requirements for the implementation of a patching system based on CVE and its availability resources to fix vulnerabilities based on manufacturer solutions. An optimal response system must be able to implement and automate incident response tasks based on the organization's necessity and environment.

There can be many use cases that an Incident Response operator may want to automate to focus its effort on complex incidents. By the way, an operator wants also to automize the recovery and backup snapshot in the organizations to complete the incident response as fast as possible not only in terms of eradication but also in terms of recovery, service restarts and reporting.

Concerning all the possible situations that may occur, will be analyzed the most useful:

- **IP Block:**
  This is a common situation where the attack source is viewed as a network entity described by means of its IP. Many times, especially in perimeter defence, there is the necessity to block an IP linked to malicious activity or reported as IoC.

  The connectivity to that IP basically is blocked from both outgoing and ingoing connections, and for this reason, this automation deals with several devices such as Firewalls, WAF, Proxy, and SWG that have to update their policy consistently.

- **Mail Sender Block:**
  Another important vector is the one concerning email traffic, so mail senders become potential threats actor in phishing campaigns, malware campaigns and reconnaissance where the most used technique is impersonation.

  Block reactively a sender mail can prevent malware infection and credential stuffing and phishing. This activity has a huge impact on organizations and for this reason, automation can consist in block mail senders if malicious content is detected. Calibrations to that automation can be provided by analyzing also sender domain, number of sent items etc... Another useful automation is the one that permits to user to report emails, emails will be automatically analyzed by a threat intelligence automated system and if malicious activity is detected, all the same mail from all involved mailboxes will be pulled down. There are lots of detection strategies to monitor emails, but they are out of the scope of this artefact.

  By the way, blocking malicious mail actors is equal, and sometimes can be considered an operation of basic importance as well as blocking malware execution.

- **Domain Block:**
  Especially in the email scenario blocking malicious domains can be useful especially in large environments to prevent infections and other malicious activity. As well as email sender and IPs, Domain can be viewed ad another network object to be blocked by perimeter defence infrastructure.

- **User Reset Password:**
  Users, especially in large Active Directory environments, can be compromised and consequentially used to perform lateral movements inside the internal network. Especially when users interact with a malicious phishing form must be vital to execute a reset password to not give possibility to use theft passwords for illegal activity.

  Many times reset password operations are linked to clean-up session activity which will be described in the next points.

- **User Clean-up Sessions:**
  Used to force logout and revoke all active sessions for a user. This is effective when a reset password is triggered from the moment that it permits to revoke of all authorization tokens issued with the previous password. So the user has to sign in again to restore all sessions; if the password is reset an attacker cannot exploit the previous session to continue its operations.

- **User Block:**
  In critical situations, there can be the necessity to disable users on Active Directory both for on-premises resources (on Domain Controller) and cloud ones (Azure and many other providers). The user then will have denied all activities performed and each attempt to log on will be blocked. This is a useful activity especially when the compromised user has high privileges.

- **Isolate Device.**
  Device Isolation is another of the most important response action to execute to contain malicious behaviour. This can be viewed as the milestone for device isolation response. Agent-on-board solution works well with common devices such as laptops and mobile devices where it blocks each computer activity until devices are not restored; by the way in large industrial systems, many times it is not possible to act from the moment that devices do not support interpreters or use domain-specific PLC systems.

  ICD and SCADA Systems can help to understand if a system is working in a compatible way concerning its expected model. In that scenarios, automated supports cannot be embedded directly on devices and for this reason, agent-solution must work to isolate logically that device for example blocking all outgoing connections.

- **Remove Malware By Hash:**
  As stated many times EDRs can remotely control the system blocking its activity and so on. Based on the principles of SSH based on a secure environment it can seek and destroy threats as normal antivirus operation does. In terms of detection and response, many commercial solutions offer automation and profiles to block the execution of malicious content, exploiting also UEBA operations.

- **Take Snapshot:**
  As described in the Recovery phase of the NIST Cybersecurity Framework Incident, Response Management needed also a good recovery system to restore normal activity after a security incident that many times can be massively devastating by cyber-attacks in terms of business production.

  To recover production the infrastructure needs a mature backup system able to create a snapshot of machines during their evolutions. Fixed the security incident at a specific instant, it permits the restoration of a functional environment before the compromise. In addition, it is possible to collect TB and TB information in case of forensic analysis or threat-hunting operations.

  Behind those aspects, again can be adopted machine learning algorithm to detect anomalies or strange behaviour inside the system's fluxes. Data Validation especially for DB represents another important security detection aspect that at the same time requires an important level of detail to implement business logins and domain-specific core operations. Backup systems nowadays are among the most recurrent automation by default for a company to provide Business Continuity as required in many standards.

- **Restore Snapshot:**
  To guarantee the efficiency of to restore system is required to ensure a reliable and replicable system. Migration to a cloud environment and multiplatform containerization permits us to restore snapshots basically wherever we want; automation of those processes provides support with a load balancer, and service replication manager to restore efficiently machines based on necessity.

- **Patch Program:**
  Finally, the system can also be implemented to support patching processes to mitigate

assessment results. Although it is considered proactive operations, within backup systems, represents a way to automize security posture growth for an entire organization.

Those are basic examples of operations that a SOC's service can automize to handle efficiently security events. Many times, based on use cases and scenarios, those basic operations can be involved in creating a pipeline to apply the model to a playbook in composing its specific security processes.

Now is possible to consider also technological solutions as already stated concerning device management, patch management to consider their interactions and benefit. As it has been stated, penetration testing and VAPT can be prioritized and automated to be performed frequently and to track progress in playbook implemented by blue proactive mitigations and automation. Operators especially when 0-Day vulnerabilities are disclosed have again to handle operation manually, by the way, based on the impact thanks to periodical assessments are able to decrease or increase its security level interchanged also the playbook selection logics. Playbooks are granularly and enhanced using day-by-day IR operations and analysis. Once the attack model is created it can be used to feed the PT automation by means of patterns and models.

In the next section will be clarified some practical aspects exploiting again all the power of machine learning and data analysis which can predict a workaround model starting from mitigations implemented through operators' actions. Starting from those assumption model presented below tries to resume all the strategical aspect based on Purple Teaming principles and automation point of view:

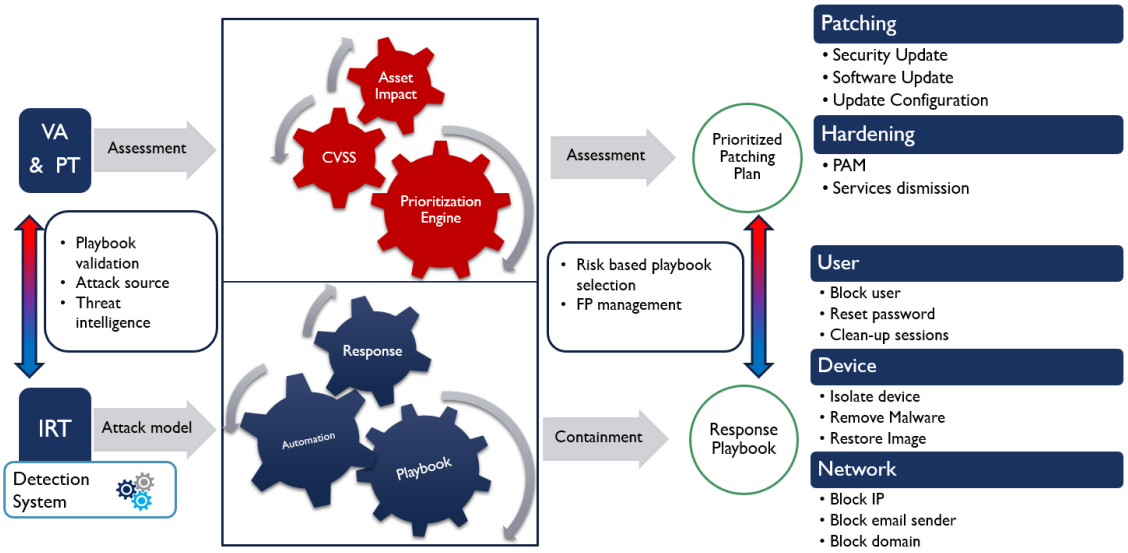

Figure 3.8. Purple teaming Integrations

3.4 Practical Considerations and Attack Mapping

Now there can be many aspects to keep into consideration while designing a system like the one presented in figure 3.8. There can be two different implementation aspects to discuss to give some logical details concerning what will be then implemented into the Developer's Manual.

The first is the Automated solutions within SOAR and XDR or Active Directory services. Many times, those types of automation can implement several requirements described before in a simple way thanks to APIs interactions. In the examples involved in Appendix B, there are many scenarios where API integration is considered the basic principle for effective automation on the endpoint, or on the User and Device Management platform that are the two main components to govern.

The second one concerns the orchestrator logic able to interface with the detection system, evaluate playbook response, and contacting the response orchestrator. Concerning those aspects, the focus is oriented on the internal logic and models to select the playbook. The basic strategy can be involved also in automated mitigation for penetration tests as discussed in section 3.1.

3.4.1 API and Automation's Integrations

As stated in the previous section, basically the main solution is to use a playbook and APIs to integrate the ideas described before into a system. There are many considerations to keep into account in designing those kinds of integrations:

- Centralized vs. Distributed Detection: detection system can be implemented in two different ways the first concern to centralize all the relevant security event through SIEM able to set logics that trigger an alert based on policy and correlated raw data in order to detect malicious activity. In that case the SIEM act as a real middleman which will redirect all the critical incident to the SOAR according to the policies defined.

  Otherwise, there is another scenario where especially adopting XDR or EDR there are distributed agents that performs detection directly on the device (no cloud computational support), in that case, the agent can be redirected directly to the SOAR when security events are triggered.

- Bottleneck and Replication: concerning the aspects exposed previously another criticality is to manage the replication and duplication of the system especially in a large environment both for performance and reliability.

  If just one node is present, it can be the vulnerable point of the architecture in case of faults. Replication permits to have more nodes involved in these operations guaranteeing fault tolerance and, more important, to implement a load-balancing system. Load balancing and replication are the main strategies against faults, and they permit the system to scale horizontally (more devices) instead of scaling vertically (more single performance).

- Leading Technologies: Many times orchestrator and automation systems are called SOAR, and they can support automated operations satisfying the other requirements already stated in the previous points. Many times those solutions permit to load directly in the main core of the application Python and other types of scripts able to be triggered if the conditions match specific policies.

Now present all the ingredients to complete the recipe, in the next sections will be presented the main design and implementation aspects to keep into consideration.

3.4.2 Playbook Design

As already stated, a playbook are simple formal model able to implement some sequential operations at a higher level than a simple flow chart. If a flow chart's node is based on simple programming instructions, the playbook model offers the possibility to abstract massively this concept and, as a result, nodes became operative blocks that can be represented by proven playbooks as well. An example will be explained as a playbook which aims to concern incident response in case of violations. The detection engine involved is the one concerned with the XDR which monitors the end user's device to detect malicious activity.

The end user device can perform several malicious actions but as stated in the previous section the most relevant are the ones concerning the malware execution or anomalous activity detected on the system. Based on the risk tolerance adopted by the infrastructure there may be several actions applicable, by the way in the next section will be explained playbook essence, the scenario presents the capability to interact with virtual machines' pool in a server farm.

- Detect anomalous and malicious activity

- Isolate device logically
- Block user on cloud Active Directory
- Run AV Scan
- Delete by hash

As explained before there are the main actions that can be useful in case of an attack but now the thesis will focus on how is possible to integrate that macro event into a classical Incident response Playbook? The next figure summarizes the operative procedure agreed upon with a customer to handle incident response starting from that assumption is possible to select the block that can easily be automated using API integration in the system. To not burden the artefact all the code developed in terms of API interaction is presented in the Appendix A which presents all the code needed to integrate this theoretical system like that with a technological solution such as VMWare and Microsoft Azure; in addition configuration aspects to set up the system will be described in the Appendix B.

The processes needed in case of an incident on non-critical servers and the Organization's laptop are the following:

As the figures show, there are already highlighted the main operations that can be automated using integrations already explained. The example is based on operations performed worldwide for a customer with a network in China, the United States and Italy. The main corporate is in Italy and monitors all the subsidiaries in a centralized way, this means that an internal CERT provided by the company needs 24/7 services for IRT from Italy, otherwise, there must be dislocated CERT inside every location, increasing resources, effort and costs.

Automations can work independently, they are not constrained to hours or time zones, and they can respond to incidents speeding up operations that can be reviewed by a centralized SOC within business hours. To support those considerations, it will be presented the differences between the same processes handled by a human being and then, once the playbook is deployed, by the automated system built.

This is the review incident table on two weeks for a classic Incident Response operations:

Incident-ID	**Playbook Applied**	**Take Over Time**	**Response Time**
Atypical travel activity for user - 0******@****.it	Block User	Sep, 02, 22:20	Sep, 02, 22:52
Suspicious Content Executed via WMI - B*******1	Block Device	Sep, 02, 00:08	Sep, 03, 01:15
Unfamiliar Sign-in Properties - T******@******.com	Block User	Sep, 09, 17:08	Sep, 09, 17:30
Atypical travel - Z******@*****.com	Block User	Sep, 09, 23:00	Sep, 09, 23:32
EMOTET malware - B******N	Block Device	Sep, 14, 00:28	Sep, 14, 01:29
Detected Suspected Stolen User Credential - 0******5	Block User	Sep, 15, 00:12	Sep, 15, 00:40

Table 3.2. Human Incident Response Stats

As the table shows there are several incidents handled in September and the average handling time for an incident is 38 minutes. In 38 minutes an attack can be devastating for organizations, but automation now can help incident response to speed up containment of attacks minimizing business impact.

Follow the table with the automated incident response system for two weeks:

Incident-ID	**Playbook Applied**	**Take-Over Time**	**Response Time**
RIG Exploit Kit URI Pattern - C******7	Block Device	Oct, 10, 23:56	Oct, 10, 23:58
Outbound connection to malicious IP - S*****C	Block Device	Oct, 15, 13:39	Oct, 15, 13:42
Multiple Unfamiliar sign-in properties - A*******e	Block User	Oct, 17, 20:52	Oct, 17, 20:55
Unfamiliar Sign-In Properties - B******.****@*****.com	Block User	Oct, 22, 17:04	Oct, 22, 17:04
Detected Suspected Stolen User Credential - 0******0	Block User	Oct, 26, 03:35	Oct, 26, 03:37

Table 3.3. Automated Incident Response Stats

As the table shows, there is a relevant speed-up concerning the automated approach that can improve global resiliency.

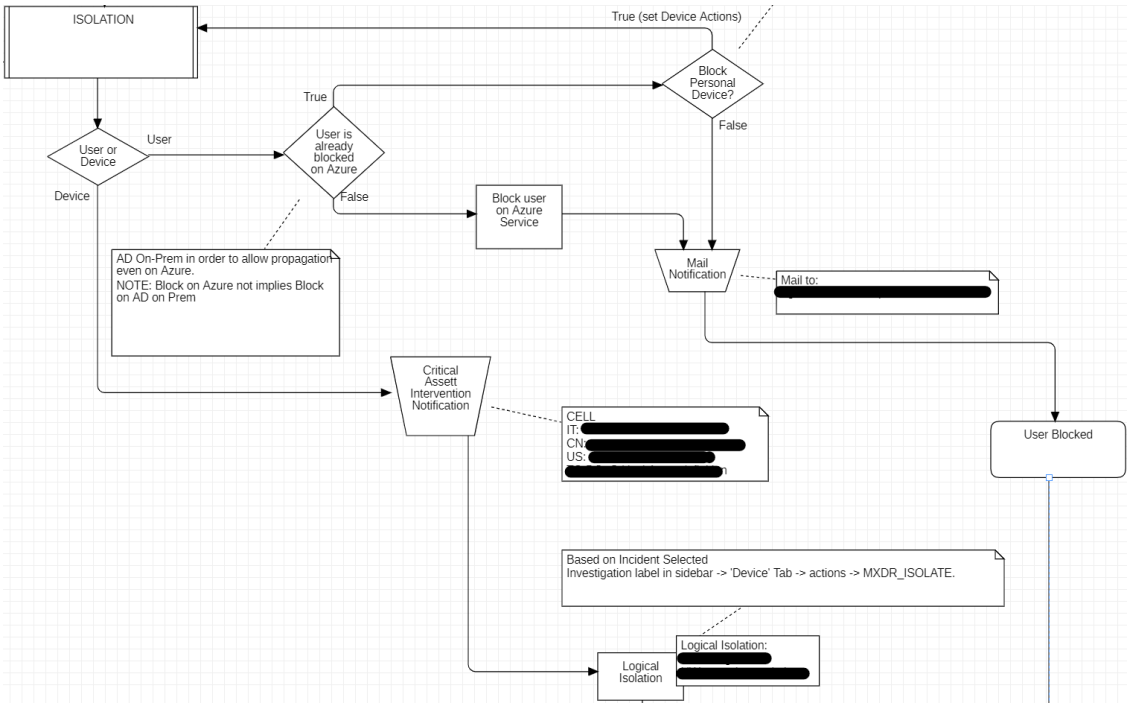

Figure 3.9. IR Playbook pt. 1

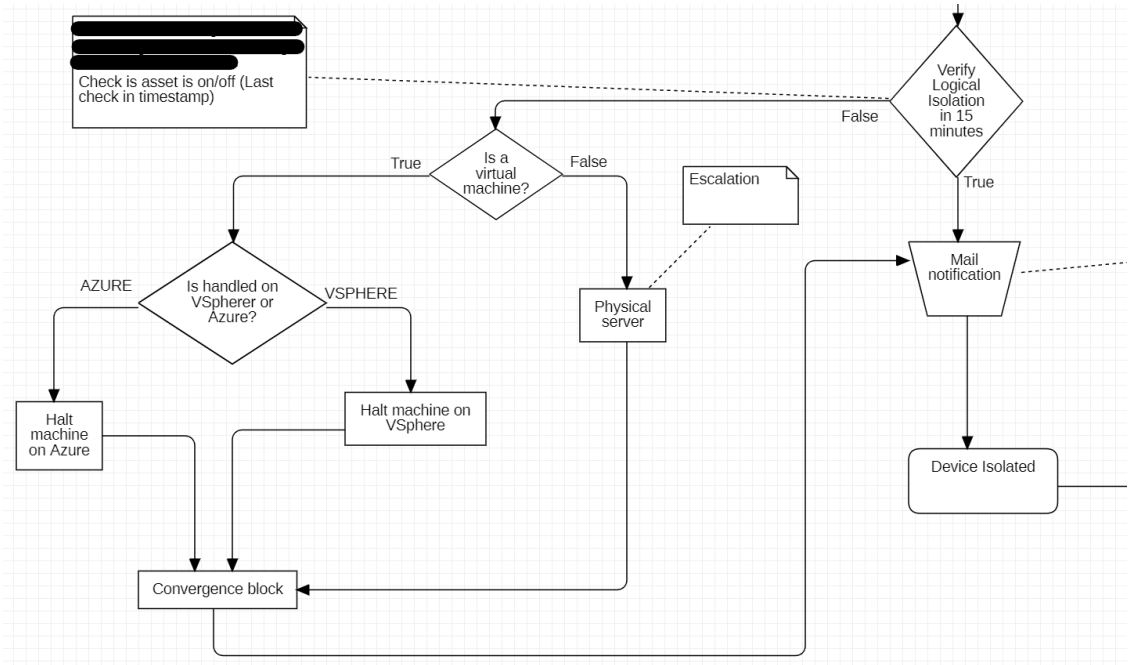

Figure 3.10. IR Playbook pt. 2

Handling an attack in just two minutes, and implementing all the actions required means reducing the critical impact windows to which the infrastructure is exposed.

In turn, that means reducing business impact in terms of earnings. In the same way can be involved just a single operator to discuss the post-incident activity, that as will be discussed in Appendix A, which can be again optimized in terms of recovery and automated backups.

3.4.3 MITRE ATT&CK Modelling

The MITRE ATT&CK [22] is a framework which relies on a matrix able to identify attack patterns based on the specific kill chain executed. This matrix presents some possible techniques that a cyberattacker can implement to reach its goal, describing operations and risks concerned. The first aspect is to understand the tactics used by an attacker during all phases involved in an attack. MITRE ATT&CK offers several sets of techniques based on the architecture under exam.

There are tactics patterns classified as Enterprise, Mobile, and ICS (related to Industrial Control Systems). All the tactics are resumed as matrix columns

- **Reconnaissance - TA0043:** related to the early phase of an attack, where an attacker is trying to gather information about the target. Several techniques are implementing this approach for example DNS, WHOIS record, Phishing information, OSINT tool etc...
- **Resource development - TA0042:** an attacker set up all the stuff he requires for its attack based on the evaluations assumed according to the information gathered. An attacker can DDoS the system by a botnet or obtain capabilities using a backdoor installed due to vulnerabilities as well as using compromised certificates.
- **Initial Access - TA0001:** Adversary tries to gain the access to internal network or DMZ using several techniques. For example, an attacker can exploit an SQL Injection to create a high-level privileged user in the domain, otherwise can use credentials were stolen by a phishing attack. For the moment is not important to focus on the techniques, by the way, this phase represents the initial foothold of the attacker inside the corporate network.
- **Execution - TA0002:** This is the real malicious code execution. In contrast with Initial Access, in this phase, the attacker can run malicious code to perform several actions. It can use various system shells to run scripts able to change system configurations, change local DNS records, and schedule feasible tasks according to its local privilege. One of the most dangerous operations is to use the compromised host to perform network discovery and lateral movements. In this case, the attacker exploits the access gained in the previous phase to effectively execute operations of various types. The effective attack starts and the next steps will be discussed in the following points.
- **Persistence - TA0003:** This tactic consists of a set of actions taken to maintain stable access to the network. It can be possible using the action performed in the execution phase: for example add SSH Authorized Keys, boot and logon autostart execution or hijacking execution flow using malicious DLL. In conclusion, this is a modus operandi adopted by an attacker to create a persistent foothold in the system, and it may not be related to the initial one used to gain first access.
- **Privilege Escalation - TA0004:** If the previous phase is accomplished, now, the attacker can access persistently the system. By the way to access sensitive data or edit critical system configurations such as Domain Controller, Firewall and SIEM, an attacker requires to have a user with high privilege, or even better a global admin. In this phase, an attacker tries to power up their privileges compromising another suitable account, or using Unix binaries such as GFTOBins to bypass local security restrictions in Unix-Like systems.
- **Defence Evasion - TA0005:** The attacker used his knowledge to avoid being detected by the security system adopted by the company. In this case, an attacker tries to cover his foot by deleting logs, hiding artefacts, disabling the system firewall, or adding the path to the exclusion list of an antivirus such as the cobalt-strike tool.
- **Credential Access - TA0006:** In this phase the attacker is trying to access with the username and password of all internal users. This tactic can be implemented through key-logger, credential dump, ARP poisoning and consecutively MITM. Otherwise, an attacker can also exploit proper resources such as GPU computing to crack NTML hash, especially in Windows Active Directory environment which supports Kerberos Authentication.

- **Discovery - TA0007:** Attacker is trying to figure out the environment to understand better where sensitive data are located, trying to identify vulnerable systems or crackable user passwords. Often native operating system tools are used in this post-compromise information-gathering phase. In this phase techniques similar to the one explained in the TA0043 - Reconnaissance is involved.
- **Lateral Movement - TA0008:** It is used by the attacker to reach and compromise another system in the environment which can be useful to achieve its objectives. To do that, an attacker can exploit Remote Services, internal spear-phishing, infection using removable media and so on.
- **Collection - TA0009:** This is the effective data collection phase, where an attacker collects sensitive information according to its goal. Data collection and preparation are the initial phases of executing data exfiltration to implement double extortion ransomware techniques. Techniques involved in this phase can be Adversary-In-The-Middle due to ARP poisoning, otherwise, an attacker can hijack a browser session or simply get data from local or interconnected systems.
- **Command and Control - TA0011:** The attacker tries to communicate with the compromised system to control them. Attackers use those techniques to communicate to another node within the network trying to mimic a normal traffic scenario to evade detection systems such as Data Obfuscation, Encoding, and Encryption in order to avoid the detection of shell commands in a communication.
- **Exfiltration - TA0010:** This is the phase in which an attacker sends sensitive information out of the internal network evading detection system. The data are collected and preprocessed in terms of encoding and/or encryption in the collection phase discussed before. To avoid detection an attacker can exfiltrate data over different protocols to reach a C2 Server controlled by himself or by third-party web services.
- **Impact - TA0040:** the attacker tries to interrupt or destroy internal systems or data (such as in a ransomware attack) to impact the company stabbing it straight in the heart. Techniques can involve data encryption, wiping, manipulation or services defacement up to firmware corruption etc...

The previous list reports all the tactics adopted by an attacker to succeed in its attack against a company. All the Tactics explained are composed of several sub-categories which are identifiable by specific techniques used by an attacker. By setting up columns as tactics and rows as techniques, we obtain an irregular matrix able to classify and recognize attacks using operations performed by the attackers, evaluating its strategies and, most precisely, the kill chain and the techniques adopted to execute attacks in all their shapes and phases.

In this way is possible to understand the way an attacker thinks according to a well-known procedure, providing model generation and evaluations. Models such as the one just explained cover a central role in pattern recognition and classification, and they can be massively integrated with Machine Learning also in order to detect and classify attacks based on similarities and actions performed. In the next section will be presented a general approach born to follow the military doctrine which can be resumed in the MITRE ATT&CK framework.

TTPs - Tactics, Techniques and Procedures

The basic idea behind the necessity of a common framework was born in anti-terrorism operations in order to prevent and recognize terrorist attacks analyzing some recurrent patterns and behaviours. Those operations require a well-structured modelling language to define all the variables and aspects involved in the scenario. For this reason, in MITRE ATT&CK there are specific situations such as Enterprise, Mobile or ICS tactics. A good kill-chain model can represent the logical phases of an attack defining a hierarchy among various tasks, and then the logical correlation among them can define the modus operandi itself. There are 4 sets of actions recognised in anti-terrorism operations:

- **Attack Preparation:** involving all the material requirements and information to prepare the attack.
- **Execution Timeline:** index of persistence and tendency to change tactics
- **Targeting:** dedicated reconnaissance, eventually on the target, to understand the weaknesses. It can be a way to know and understand an emulation of the real target.
- **Planning Stages:** defining a way to act according to other elements present in the kill-chain, this pattern defines the real essence of the modus operandi according to top the level of the detail.

According to the definitions reported before, it can be related to those methods and the one presented in the MITRE ATT&CK matrix defining The first stages of planning can be viewed as preparatory actions to Recon the target and consecutively weaponize according to the requirements. Those are the PRE-ATT&CK techniques contained in the Reconnaissance and Resource Development operations. After this initial stage, there is the real execution of the attack involving the remaining stages of MITRE ATT&CK Matrix discussed before.

Purple Teaming operations are massively used pattern detection to classify a pattern as malicious. Differently from hash-based recognition TTP classification is not an exact match as in the digest equality, but it is an evaluation of similarity among the environment variables with respect to some well-known and structured malicious models. The main idea behind mixing that operation as in purple team operation is to use the red team to assess a system and at the same time evaluating detection system and incident response capabilities trying to reduce the false positive ratio in case of automatic response. For this reason, as the NIST SP 800-150 states, TTPs describe the behaviour of an actor.

Tactics are high-level descriptions of behaviour, techniques are detailed descriptions of behaviour in the context of a tactic, and procedures are even lower-level, highly detailed descriptions in the context of a technique. TTPs could describe an actor's tendency to use a specific malware variant, order of operations, attack tool, delivery mechanism (e.g., phishing or watering hole attack), or exploit. For this reason, nowadays cyber-warfare prevention systems can associate attack types and specific activism groups. Follows an HTTP payload from the KILLNET attack against Italian Energy companies performed in June 2022:

Figure 3.11. KillNet's HTTP Request

The DDoS IPs sources are connected to zombies which belong to Legion and KillNet botnets (especially IoT devices and proxies).

Now for the scope of this artefact, we are interested in a specific details level, to understand the phases and the threats involved in the system at the level of files, processes and, more important, behaviours. Once identify the threats according to those parameters we assume to remove permanently the threats and mitigate the flaw by implementing strategies to resolve the issues.

Those aspects will be covered later during the implementation part, by the way, there is the need to handle an incident once the detection system has recognized it. The next section will be focused on the NIST Cybersecurity Framework to handle and efficiently manage security and according to the techniques provided in purple teaming operations.

Chapter 4

Conclusions

4.1 Global Benefit and Considerations

As the data collected shows, there is a huge speed-up of processes taken into exam starting from their effective speed-up that means faster incident response as required in many standards stated in the section chapter 1.

Based on the playbooks developed customers with SOC's support can potentially automize every process within their cybersecurity scope. The real challenge in making that is related to the difficulty to design an effective playbook able to cover all the possible scenarios both for incident response and for assessment mitigation or recovery system especially for critical infrastructure. Playbooks must be calibrated inside an iterative process aimed to correct their false positives impact in case the detection's prediction was incorrect. For this reason, issues and low accuracy in detection can have a huge impact.

But as discussed in the playbook designed, the process must evaluate this aspect within manual operations that are needed at before deploying the real automation itself. Once this preliminary tuning phase is completed and false positives are excluded from the playbook is possible to automate the incident response processes gaining all its benefits.

Corrections and policies hardening is a sign that the system is evolving correctly following new scenarios and threats emerging into the wild. This is an important consideration because it means that the system has developed a maturity model so that it can adapt new cases modifying consistently playbook and operative modules with less effort. This can be viewed as a state-of-the-art in cybersecurity posture: a system able to understand how the system works according to their behaviour as discussed in the detection phase in section A.2.2, especially for UEBA and NBAD systems. Anomalies can be triggered and based on their impact is possible to apply an automated playbook based on the risk detected. With the technologies introduced, it is possible to detect internal and external threats and meet regulatory and compliance requirements. For a company, this means gaining confidence in their customer not only from a core services point of view but also in terms of awareness. For a customer, knowing that a company is using leading technologies to protect their data makes the user itself trusts the organization. For the corporate, this proof can be given using compliance satisfaction, and more importantly, thanks to a PDCA model which not only guarantees a proper security posture today but also within a month a year and so on...

All those assumptions are made following the processes explained in this artefact, to demonstrate that customers after a tuning period, which requires the same effort, can improve their system in terms of:

- **Visibility:** If SOC operator ′ wastes ′ his time in handling recurrent and well-known incidents he cannot deal with hardening and improvements.
- **Speed-Up:** Automations work faster than a human response.

- **Frequency:** If the system speeds up, frequency increases, consecutively assessment and mitigation can be automated as well.
- **Security Posture:** If assessments are performed frequently also mitigation can be implemented more frequently. This is equivalent to saying that the system evolves to a better target status many times, again following ideas of the PDCA model. Improvement means increasing the overall security posture.

By the way, this is not the only benefit because all the improvements described are just from a company point of view. SOC will benefit from this system as well, especially in earning and managed services. A model like the one presented requires a decreasing management time by an operator. Obviously, in the first phase, SOC must define the security stack, develop the system in a compliant way, test it on a pilot, deploying it on the entire network. Then the SOC must define the playbook based on scenarios, business scopes and relevant incidents. The real power of the playbook stays here: the playbook defines a logical kill-chain to respond to an attack or implement the mitigation. It does not define low-level code: basically, the SOC operator once designed a high-level playbook can implement them in several ways depending on how the security stack is composed. This fact reduces drastically the designing phase because the operative and governance processes are the same.

4.1.1 Detection Benefit Review

Detection's benefit concerns the one already mentioned in the previous sections. Machine learning nowadays offers a consistent solution to detect anomalies and malware with high accuracy, starting from those considerations is possible to resolve also problems concerning hash evasion. Finally, Machine learning solutions especially for Anomaly Detection can be easily trained from the moment that common activities can be easily collected within a production environment resolving the problems of dataset creation.

Furthermore, for malware detection dataset of benign samples can be composed by the organization's allowed list and well-known software such as Windows executable, Office suites, Adobe Suite, browsers etc... In the same way, malware samples can be obtained from governance entities as CISA, local CERT and CSIRT as well as cybersecurity brand as Bitdefender, Sophos etc... Also, in this case, the dataset can be easily composed to train the machine learning model to recognize as benign a known set of allowed executables. Hash recognition can be used anyway to support detection operations. Thanks to integrations with MITRE ATT&CK and there can be viewed also patter categorization to recognize threats and their techniques.

If source code is available detection can be also performed on open-source software looking for a pattern that can be identified as critical or vulnerable thanks to integration with the CWE pattern provided again by MITRE. Many projects are already available to support also source code analysis as well as malware and anomaly detection widely adopted in innovative detection systems such as SIEM, XDR and NDR. Starting from this point SOC can easily detect threats close-to-real time and then proceed to handle the incident according to NIST's best practices. The benefit provided by machine learning-based solutions is dominant in the detection phase.

4.1.2 Response Benefit Review

Concerning the response automation benefit, they are already exposed in the previous sections. The benefits acquired by those solution permits speeding up the global response system both in terms of incident response and in terms of VAPT mitigations.

Incident response requires to automize classical human processes using API offered by various security stack components. Especially for XDR and EDR, which have a global vision of what is happening on the device both in terms of local processes, activities and user behaviour, it is possible to act directly by blocking devices, and deleting malware entities by hash as described in the appendix A.4. If the system permits, it is also possible to halt the machine, especially for the virtualized environment such as server farms etc... Those solutions can be also extended to

Active Directory architecture to block user activity in case of compromise, this can be also useful especially for email traffic from the moment that phishing represents the preferred attack vector for an attacker to steal credentials. Services like that permit to block completely user activity revoking all the sessions active for the user. Finally, those operations can be also automated in terms of backups and recovery of machines that are provided with a backup agent. Automations permit also keeping track of which actions are completed against the ones that present errors, in that way the IRT has a clear vision of the system during post-incident activities.

Vulnerability Assessment and Penetration Tests are more trivial to patch, but exploiting innovative solutions is possible to highlight possible weak passwords thanks to the tool as Hash-Cat then, starting from that list, it is possible to force a password reset for all the users. In the same way, is possible to prioritize the mitigation to update software and Operative Systems deciding the best moment to schedule such operations.

4.1.3 Long Terms Predictions in SOC's Managed Services

As anticipated in the introduction of this section SOC benefit consists of effort reduction, organization of operative procedures as required by standards and processes to speed up, permitting SOC's operator to focus on post-incident activity and hardening strategies.

By the way, it is important to highlight that starting from those concepts, a SOC can reuse a well-proven playbook in incident response and PT mitigation in a similar scenario. Once the API interaction is implemented and tested the adopted systems can be applied in other cases. Even if different products selected by customers can present different APIs the operative logic flows are the same as described in the playbook, and this fact reduces significantly the effort required. The basic aspects are the following: as described in the Appendix A.4 APIs depend on the manufacturer, for example, VMWare will provide API different from Cynet products. By the way, the two systems will be also adopted to execute basic tasks such as user block or malware removal as described previously. The operative procedure, based on the severity and risk related to the asset or user itself, will always be described by the same playbook and then can be extended with use cases according to the scenario and customer's needs.

The results collected thanks to my experience in the NAIS' Security Operation Centre explain that model like the one exposed can be involved to improve the security posture of an organization thanks to automation and cooperation among SOC's managed services. Improvements are described in terms of frequency and duration between risk identification and its complete mitigation; in addition, many controls provided in major standards are covered. Results can demonstrate that as might guess automation is more efficient than a human operator, especially in well-proven activities. This fact led to reducing significantly the time required between each PDCA iteration which in turn means speeding up security posture improvements respecting what is stated in ISO 27001 Annexes.

Customers will be able to acquire those models to satisfy their security requirements even if they are not operating in cybersecurity field, exploiting SOC's expertise and designing a resilient and robust security stack to support entirely the infrastructure. Furthermore, from the SOC's point of view, a process like the one explained can significantly reduce the effort by a human operator after a tuning period, which can be considered a huge step for productivity and, more important, earnings. This means that human operators can be involved with newer customers to design the automation system following the same approaches. More customer for SOC means more earnings, and by reducing the effort required, the analyst can focus on critical alerts and new security events.

It is a *'Win-Win'* model where customers can improve their security posture, the frequency for assessment speeding up their operations; service providers as SOC instead gain in effort reduction and earnings. All operations are performed following major standards and this aspect gives compliance validation of the system extremely useful in terms of audit and certification. In conclusion, those types of considerations consolidate the reason why a system like that should be adopted to satisfy security aspects nowadays, permitting consistent automation to improve the security posture globally.

Appendix A

Developer's Manual

A.1 Malware Detection and Analysis

In this implementation, the part will be discussed all the strategies that a developer can use to monitor the events that can happen in the system. In the following sections will be presented simple script using Python3 to support analyst's operations, referencing the NIST Cybersecurity framework where it is possible. The decision to use Python3 for the implementation derives from the useful features offered by the Language in terms of libraries and Machine Learning support. Obviously, in complex systems, there can be also minor devices that are not able to support a Python Interpreter. This is the case of IoT and old devices where computational resources are limited; concerning those issues, there can be several solutions able to address them for example Agentless solutions working at the network level. All those solutions will be discussed in the review phase according to the code developed in the next sections. All the code presented in this artefact is provided as a basic building block which can be integrated into more complex technologies such as XDR or SOAR to implement the automatism discussed before.

As introduced in previous sections, two strategies will be analyzed. The first concerns a simple static analyzer implemented through `psutil` and `hashlib` libraries and Virus Total APIs to detect running processes and their relative reputation based on the hash comparison. This is just a simple example to introduce the working context, then more suitable and complex requirements and consideration are taken into account during the development phase.

A.1.1 Hash Based Malware Detection

As introduced in the previous section, the starting point is a simple static malware detection system in order to understand how things work. The code presented refers to the module `/pythonSrc/detection/HashBasedMalwareDetection/`.

The first library involved is `psutil` which is an intermediate API library able to retrieve system information such as running processes, system stats (CPU usage, Memory usage, available disk space etc...), network stats per process and other useful information able to reconstruct the system status. The second one is `Hashlib`, a popular hashing tool library in order to compute the hash of the PE file involved in the running process.

The basic block presented in this section consists of 3 different phases:

- Get all the running processes.
- Iterate over all processes
- Get the hash (SHA256) related to the PE file
- Get info from Virus Total concerning the reputation of the Portable Executable file.

According to this procedure, the python script is able to detect all the running processes in the system.

Once the evaluations are performed, an output consisting of ProcessName, ProcessID, AbsolutePath and VirusTotal Reputation is created. By the way, Virus Total API returns a lot of useful data such as reputation according to several AV Engines, SandBox Behavior, Creation Date and many others that will be used in the next section based on the requirements. Follows the code needed to implement what has just been discussed:

```
import hashlib
import psutil
from VirusTotalREPU import virus_total_reputation

def get_hash_value(file_name: str):
    hash_val = hashlib.sha256()
    try:
        with open(file_name, "rb") as f:
            for byte_block in iter(lambda: f.read(4096), b""):
                hash_val.update(byte_block)
        return hash_val.hexdigest()
    except FileNotFoundError:
        print("No file detected for process: " + file_name)
        return "<NULL>"


for proc in psutil.process_iter():
    try:
        with proc.oneshot():
            with open('./res.txt', 'a') as file:
                file.write(proc.name() + " ::: " + proc.exe() + " ::: " +
                    str(proc.pid) + "\n")
                dgst = get_hash_value(proc.exe())
                reputation = virus_total_reputation(dgst)
                file.write("\t\tSHA256: " + dgst + "\n")
                file.write("\t\tVIRUS-TOTAL INFO:" + "REPUTATION: " +
                    str(reputation) + "\n\n")
                if reputation < 0:
                    print("WARNING: Possible malware detected(%s)" % proc.exe())
            file.close()
    except (psutil.NoSuchProcess, psutil.AccessDenied, psutil.ZombieProcess):
        print("Issue with process: " + str(proc.pid) + " ::: " + proc.name())
        continue
```

Listing A.1. HashDetectionSystem.py

In order to interface the evaluated hash with VirusTotal and get the reputation, the program has to use the Virus Total APIs. As many reputation providers do, to use the APIs interface it requires a subscription and, based on its type, it is able to provide basic or premium services. Follows the script able to provide this kind of interaction, implementing the `VirusTotalRepo()` routine:

```
import const
import requests

def VirusTotalREPU(sha256):
    # GET approach by deafult
    try:
        print('processing: ' + sha256)
        url = "https://www.virustotal.com/api/v3/files/{0}".format(sha256)
        response = requests.get(url, headers={
```

```
        'Accept': 'application/json',
        'x-apikey': const.VIRUS_TOTAL_APIKEY,
        'User-Agent': 'Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36
            (KHTML, like Gecko) Chrome/41.0.2228.0 '
                    'Safari/537.36',
    })
    if response.status_code == 200:
        # print(str(response.json()))
        return str(response.json()['data']['attributes']['reputation'])
except Exception as e:
    print(e)
```

Listing A.2. VirusTotalREPU.py

The system is able to generate records like the one presented above:

```
PROCESS_NAME: C:\Windows\System32\csrss.exe,
PROC_DGST:
    6f1c9b4c187669bc0371260d121caf48d65f829a9104c483befbd8fc0bed24f5,
PROC_REPUTATION: 12
```

Latter presents a Windows system process, and it is confirmed by the fact the location of the process is `C:\Windows\System32` and, most importantly, the executable has a reputation greater than 0. According to VirusTotal engines and the community this hash is considered benign.

Based on the information gathered by this sample, a script like that is able to detect known malware, and consecutively trigger a kill process based on PID information. Once the threats are contained and suppressed can be eradicated by means of hashdeletion in the filesystem or directly knowing the source path of the executable. This subroutine can be easily integrated with the work-flow described before by means of `os` utils provided by python as follows:

```
import os
import signal

AUTOMATIC_DESTORY_FLAG = True

def seek_and_destroy(pid: int, path: str):
    print("Automatic remove malware at %s" % path)
    os.kill(pid, signal.SIGKILL)
    if AUTOMATIC_DESTORY_FLAG:
        os.remove(path)
    else:
        res = input("Remove malicious exe located at: %s ? (press 'Y' to
            confirm, other keys will not remove) " % path)
        if res.upper() == 'Y':
            os.remove(path)

def get_hash_value(file_name: str):

    ...

    if reputation < 0:
        #print("WARNING: Possible malware detected(%s)" % proc.exe())
        seek_and_destroy(proc.pid, proc.exe())

    ...
```

Listing A.3. HashDetectionSystem.py

Eluding Static Hash Analyzer

In this brief introduction has been presented the code needed to implement an automated static detection and response automation. As stated before, cyberattackers can easily elude the hash detection system just discussed by changing its structure but not its behaviour. Suppose that an executable malicious file is composed of the following set of instructions:

```
pushl %ebp
movl %esp, %ebp
subl $0x4, %esp
```

The following code presents a simple situation in which the CPU save the content of `%ebp` register into the stack, then override move content of `%esp` into register `%ebp` and finally performing a subtraction like this: `R[esp] = R[esp] - 0x04`, where convention `R[regName]` indicates the content of the register.

By the way, the same result can be obtained with this other set of instructions:

```
nop
pushl %ebp
nop
movl %esp, %ebp
nop
subl $0x4, %esp
nop
```

Differently from a high-level programming language where comments are trimmed out during the compilation phase, NOP operations are persistent also in the executable altering the final hash produced. This assumption is confirmed by the fact that NOP operation presents OP-CODE as well as other "real" Machine Operations such as SUB, ADD or PUSH. For example, its value in the x86 CPU Family is 0x90.

According to those facts, a system like the one described before is vulnerable to those kinds of attacks and may not recognize threats. By the way, the example presented shows also the major resolution in order to address those issues. The detection system has to no longer evaluate the structure of a program, but otherwise, they examine the behaviour. In order to do this we need a model and the next section will be described some important innovations in the Machine Learning scenarios are able to address those kinds of situations.

A.1.2 Machine Learning Based Malware Detection System

In this section will be presented the technical aspects and strategies to implement a machine learning-based malware detection systems referring the code present in the module `/pythonSrc/detection/MLBasedMalwareDetection/`. Those exploit a technique called locality-sensitive hashing (LSH) often used also to reduce the number of dimensions of an entity maintaining a good level of description and can be considered a good implementation for data clustering and the nearest neighbour search. With respect to a classical hash function, it is able to maximize the number of collisions in order to implement a probabilistic bucket selector. Those kinds of models work to reduce the input set (composed by M entity) in a number N so that M ¿¿ N. Follows some basic instructions able to implement a solution like the one presented above exploiting `ssdepp` library:

```
import ssdeep

str1="nop pushl %ebp nop movl %esp, %ebp"
str2="pushl %ebp movl %esp, %ebp"
hash1 = ssdeep.hash(bytes(str1, 'utf-8'))
hash2 = ssdeep.hash(bytes(str2, 'utf-8'))
print("Compare hash1 and hash1: " +
    str(ssdeep.compare(bytes(hash1, 'utf-8'), bytes(hash1, 'utf-8'))))
```

```
print("Compare hash1 and hash1: " +
    str(ssdeep.compare(bytes(hash1, 'utf-8'), bytes(hash2, 'utf-8'))))
```

Libraries like the one presented are widely used nowadays to implement those systems. They provide a sensitive solution able to reduce and detect recurrent patterns by means of dimensionality reduction. Obviously, this type of solution can be easily integrated to perform similarity analysis on a more complex entity such as a file working directly on its binary representation.

Another important solution can be the one concerning the N-Grams extraction. This is a technique where a sentence is divided into a set of N sequential tokens, and this sequential analysis is able to recognize patterns in a more detailed way by means of corpus analysis. According to the Markov chain model, N-Grams solutions are able to statistically predict and generate text from a model corpus.

Follow a simple example written in pseudocode to understand better the solution proposed:

```
string = "I am a human being"
N-Grams(1, string) = ["I", "am", "a", "human", "being"]
N-Grams(2, string) = [["I", "am"], ["am", "a"], ["a", "human"], ["human",
    "being"]]
N-Grams(3, string) = [["I", "am", "a"], ["am", "a", "human"] ["a",
    "human", "being"]]
```

Python offers several libraries to perform those kinds of computations, among them the most important are the ones contained into the `nltk` module. According to the results obtained in the wild, the most frequent N-Grams set are also the most informative ones regarding the malware classification branch. For this reason, good malware detection can easily analyze the common pattern in well know malware or families to recognize the classic pattern and the modus operandi of the malware. Follows a simple script able to implement the strategies described above:

```
from nltk import ngrams
import collections
import sys

if len(sys.argv) != 3:
    sys.exit(-1)
filePath = sys.argv[1]
N = int(sys.argv[2])
with open(filePath, "rb") as file:
    rawData = file.read()
ngrams = list(ngrams(rawData, N))
collectionInfo = collections.Counter(ngrams)
for item in collectionInfo.most_common(15):
    print("pattern " + str(item[0]) + " found " + str(item[1]) + "
        times\n")
```

Listing A.4. NGramsSelection.py

The presented script accepts two command line parameters which are the file path we want to analyze and the cardinality of the NGrams (components of the set). After the file reading, we collect the N-Grams result into a list provided by `collections` utils.

Then, applying simple map-reduce operations we count the number of patterns which present the same structure (sequence of 32 bits). As the first option we select the most common patterns provided by the cardinality for a specific pattern thanks to Collection's most_common routine which analyzes a tuple composed as pattern key and frequency value. The result produced for the chrome.exe is composed of those patterns:

```
pattern (0, 0, 0, 0) found 110833 times
pattern (255, 255, 255, 255) found 62317 times
...
pattern (0, 0, 72, 137) found 6387 times
pattern (0, 0, 72, 139) found 6036 times
```

As the result shows, the most common pattern is the (0, 0, 0, 0) that is a 32-bit `0x00` value. This is a result which is not so useful since the pattern is the same and basically this sequence inside an executable file does not represent anything. Maybe its recurrent presence can be associated with padding functionalities.

Based on the solution just presented there can be the possibility to develop a complete detection system based on the statistical analysis was performed on data samples. By the way, there is an important aspect to keep into account, and it regards the access to the samples itself. As stated in the previous section this solution follows the similarity distance as a metric for the clustering, thanks to the algorithm as hash-similarity or N-Grams; all of those techniques can be categorized as unsupervised learning, able to extract independently relevant features in different patterns.

The challenging evaluation is in classification which is the nature of the cluster derived from this first step. This is made by means of supervised machine learning where data present labels, basically assigned by a human operator. This is the most delicate phase because noisy data will produce wrong predictions and classifications. Often in cybersecurity operations, the construction of a machine learning prediction model is made as described following the steps presented below:

- Select the dataset to analyze and check that data is consistent
- If possible labelled data in a consistent way as well, by a human operator
- Reduce cardinality by filtering and standardizing collected data
- Extract relevant feature
- Detect which features are considered important to classify the input data according to the labels desired
- used a supervised machine learning process to classify unknown data, avoiding over-fitting

In the next section will be presented a more robust algorithm able to implement what is stated in this section.

Selecting Best N-Grams from samples

In this section is presented the processes used to implement a feature selector based on the executable file binaries. In order to do that a malware executable and a benign executable dataset are created with a balanced approach. The script requires all the arguments by command line and must be provided with the following ones:

- Malware executable samples folder
- Benign executable samples folder
- NGrams cardinality
- Number of the extracted most common pattern (NGrams)
- Index for score function (Frequency, Mutual Information, Chi-Squared)

In order to get the samples and train the model, executable files related to malware were downloaded from the SOREL-20M repo. This repo contains more than 20 million malware samples handled by the well-known cybersecurity company SOPHOS.

Below is reported the Python script able to implement what was described before:

```
from nltk import ngrams
import collections
import sys
from os.path import isdir, isfile, join
from os import listdir
```

```
import numpy as np
from sklearn.feature_selection import SelectKBest, mutual_info_classif,
    chi2

# SECTION 1
if len(sys.argv) != 6:
    sys.exit(-1)
try:
    pattern_cardinality = int(sys.argv[3])
    k1 = int(sys.argv[4])
    k2 = int(sys.argv[5])
    malware_directory_path = sys.argv[1]
    if not isdir(malware_directory_path):
        sys.exit(-2)
    benign_directory_path = sys.argv[2]
    if not isdir(benign_directory_path):
        sys.exit(-3)

    # SECTION 2
    tot_ngrams = collections.Counter([])
    for directory in [malware_directory_path, benign_directory_path]:
        local_sample_set = [file for file in listdir(directory) if
            (isfile(join(directory, file)))]
        for sample in local_sample_set:
            file_abs_path = join(directory, sample)
            raw_file = read_file(file_abs_path)
            tot_ngrams +=
                get_collection_counter(get_ngrams_from_sample(raw_file,
                pattern_cardinality))

    most_common_ngrams = tot_ngrams.most_common(k1)
    most_common_ngrams_list = [item[0] for item in most_common_ngrams]
    # print(str(most_common_ngrams_list))


    # SECTION 3-A
    directory_and_label = [(malware_directory_path, 0),
        (benign_directory_path, 1)]
    X = [] # feature
    y = [] # label
    for (directory, label) in directory_and_label:
        local_sample_set = [file for file in listdir(directory) if
            isfile(join(directory, file))]
        for sample in local_sample_set:
            file_abs_path = join(directory, sample)
            raw_file = read_file(file_abs_path)
            X.append(get_feature_from_sample(raw_file,
                most_common_ngrams_list, pattern_cardinality))
            y.append(label)

    # SECTION 3-B
    X = np.asarray(X)
    max_X_freq = X[:, :k2]
    print("TOP FERQUENCY:")
    print(str(max_X_freq))
    mutual_info_selector = SelectKBest(mutual_info_classif, k=k2)
    max_X_mutual_info = mutual_info_selector.fit_transform(X, y)
```

```
print("\n\n\nTOP MUTUAL INFORMATION:")
print(str(max_X_mutual_info))
chi2_selector = SelectKBest(chi2, k=k2)
max_X_chi2 = chi2_selector.fit_transform(X, y)
print("\n\n\nTOP CHI2 SELECTION:")
print(str(max_X_chi2))
```

Listing A.5. NGramsSelection.py

The reported script can be divided into 3 main sections:

- **SECTION 1:** In this section is just checked the command line parameters in order to ensure that all the configurations are set up properly.
- **SECTION 2:** In this section is generated the N-Grams for all the executables found in the source directories. Pay attention that in this case there is no interest to know the class to which an entity belongs. The outer for-loop iterates on all sources while the inner one read the file, consequently generating Ngrams and counting occurrences. Follows the three routines involved in the section just presented:

```
def get_ngrams_from_sample(sample_input, n):
    ngrams_list = list(ngrams(sample_input, n))
    collection_info = collections.Counter(ngrams_list)
    return collection_info

def get_collection_counter(collection):
     return collections.Counter(collection)

def read_file(path):
    with open(path, "rb") as file:
        raw = file.read()
    return raw
```

Listing A.6. NGramsSelection.py

- **SECTION 3-A:** In this section will be selected the most interesting NGrams among the ones selected as the most frequent in section 2. In this phase, a good NGrams is one able to discriminate perfectly between malware and benign executables. In order to do that there is the necessity to have classified and labelled data. For this reason, at that point, the sources' folder which contains the samples is labelled with 1 (benign) and 0 (malware) defining a binary classifier. Then is performed the same routine as presented in section 2, but now we composed two sets X filled with the relevant features according to the `most_common_ngrams_list` globally extracted in section 2 by means of the `get_feature_from_sample` routine, reported below:

```
def get_feature_from_sample(sample_input,
    most_frequent_ngrams, n):
    list_length = len(most_frequent_ngrams)
    feature_array = list_length * [0]
    list_ngrams = get_ngrams_from_sample(sample_input, n)
    for i in range(list_length):
        feature_array[i] = list_ngrams[most_frequent_ngrams[i]]
    return feature_array
```

Listing A.7. NGramsSelection.py

As the code shows a `feature_array` is generated based on computed ngram for the current sample, then it searches all the ngrams that are also presented in the global `most_common_ngrams_list`. If there is a match, the number of local occurrences is selected as a feature otherwise remains zero, as setting up in the initialization of the `feature_array`. Now we have the following relationship: in `X[i]` we have the `k1` most relevant feature for the sample of index `i`, while in the `y[i]` its related label (malware of benign).

- **SECTION 3-B:** Now we want to discriminate among the features extracted in order to reduce again the cardinality according to reduction factor `k2` and three different algorithm:
  - *Frequency:* Select the most frequent `k2` items among the one presented.
  - *Mutual Information:* In probability theory and information theory, the mutual information (MI) of two random the variable is a measure of the mutual dependence between the two variables. More specifically, it quantifies the "amount of information" obtained about one random variable by observing the other random variable.
  - *Chi-Squared:* Mathematically, a Chi-Square test is done on two distributions two determine the level of similarity of their respective variances. In its null hypothesis, it assumes that the given distributions are independent. This test thus can be used to determine the best features for a given dataset by determining the features on which the output class label is most dependent. Follows the formulas:

    $$\chi^2 = \sum_{i=1}^{m} \sum_{j=1}^{k} \frac{(O\_ij - E\_i,j)^2}{E\_i,j}$$

    Where O stay for Observed Frequency while E stays for Expected Frequency, i and j are the table indexes.

This section ends by demonstrating the strategical utility of these strategies working with the following command lines parameter:

```
C:\Temp\reduced\malware C:\Temp\reduced\benign 2 1000 10
```

In order to have a balanced environment, the two selected folders contain both 200 samples, then there is the computation of 2 Ngrams sets, selecting 1000 global most common patterns (ngrams) and extracting the 10 most relevant information according to the labels. According to Python's displayed output, we have reduced this amount of information. From the 400 total samples analyzed, the system extracts 65536 distinct NGrams, then we selected the most common 1000 globally. Then in the labelled scenario, there is the generation of X which is a 400 x 1000 matrix, then according to `feature_selection` the algorithm in the `scikit-learn` module, the system is able to reduce the initial matrix to a 400 X 10 table maintaining the same level of accuracy and detail.

Machine Learning Based Malware Detection System

In this subsection will be presented a simple solution for a malware classifier which considers various information contained in the executable file. The model presented is specifically created to deal with Portable Executable file format by means of `pefile` module installable in Python. In a similar way can be generalized another model to deal with ELF from the moment that some features are different in the two standards.

The presented model will examine and classify a file according to a supervised strategy and an NGrams feature selection, the analyzed feature will be:

- Number of Sections
- Imported DLLs
- Type of Section

In the next code's sections will be presented the main phases, some parts can be omitted from the moment that they concern the solution already explained in the previous sections:

```
# Create data set
directory_and_label = [(malware_directory_path, 1),
    (benign_directory_path, 0)]
all_samples = []
```

```
labels = []
for (directory, label) in directory_and_label:
    local_sample_set = [file for file in listdir(directory)]
    for sample in local_sample_set:
        file_abs_path = join(directory, sample)
        all_samples.append(file_abs_path)
        labels.append(label)

# Splitting dataset into test and train sets
samples_train, samples_test, labels_train, labels_test = train_test_split(
    all_samples, labels, test_size=0.33, stratify=labels, random_state=11
)

# Collect NGrams of train samples
ngram_counts_global = collections.Counter([])
for item in samples_train:
    raw_file = read_file(item)
    ngram_counts_global +=
        get_collection_counter(get_ngrams_from_sample(raw_file,
        pattern_cardinality))

# Get k1-common pattern
most_common_ngrams = ngram_counts_global.most_common(k1)
most_common_ngrams_list = [item[0] for item in most_common_ngrams]
```

Listing A.8. MalwareDetectionSystem.py

In this section is presented the pre-processing of the executables creating a global data set containing all the executables the absolute path needed and its relative binary label. The malware and benign directories are indicated by the command line parameter.

Then the global dataset is split into test and train partitions and is extracted the common patterns (2-Grams) from the training partition, searching for the `k1` most common patterns, as explained in the previous subsections.

```
pe_imports_train_set = []
pe_sections_train_set = []
pe_number_of_section_train_set = []
ngrams_feature_train_set = []
labels_train_set = []

for i in range(len(samples_train)):
    sample = samples_train[i]
    try:
        raw_file = read_file(sample)
        NGram_features = get_feature_from_sample(raw_file,
            most_common_ngrams_list, pattern_cardinality)
        pe = pefile.PE(sample)
        imports = get_imports(pe)
        n_sections = len(pe.sections)
        sec_names = get_section_names(pe)
        pe_imports_train_set.append(imports)
        pe_number_of_section_train_set.append(n_sections)
        pe_sections_train_set.append(sec_names)
        ngrams_feature_train_set.append(NGram_features)
        labels_train_set.append(labels_train[i])
    except Exception as e:
        print("Error: " + str(e))
```

Listing A.9. MalwareDetectionSystem.py

In this phase is analyzed features stated before, are consecutively indexed all the in an apposite arrays in order to keep into account information related to the same index extracted in the train-sample dataset. Then thanks to integrations offered by the pefile module is possible to extract common information and discriminating among them with features extractor method `get_feature_from_sample` explained in the previous section. Here are reported the involved routine implemented with pefile module tools.

```
def preprocess_imports(dll_list):
    temp = [dll.decode().split(".")[0].lower() for dll in dll_list]
    return " ".join(temp)

def get_imports(pe_file):
    list_of_imports = []
    for entry in pe_file.DIRECTORY_ENTRY_IMPORT:
        list_of_imports.append(entry.dll)
    return preprocess_imports(list_of_imports)

def get_section_names(pe_file):
    list_of_section_names = []
    for sec in pe_file.sections:
        normalized_name = sec.Name.decode().replace("\x00", "").lower()
        list_of_section_names.append(normalized_name)
    return "".join(list_of_section_names)
```

Listing A.10. MalwareDetectionSystem.py

Then are exploited the tools offered by `scikit-learn` in order to pre-process textual data and standardize them according to numerical data required by the analysis. Finally, a Random Forest Classifier model with 100 estimators is trained on the gathered data.

```
    pe_imports_get_feature = Pipeline(
        [("vect", HashingVectorizer(input="content", ngram_range=(1, 2))),
         ("tfidf", TfidfTransformer(use_idf=True))]
    )
    pe_section_names_get_feature = Pipeline(
        [("vect", HashingVectorizer(input="content", ngram_range=(1, 2))),
         ("tfidf", TfidfTransformer(use_idf=True))]
    )

    # Transform training data-set
    pe_imports_train_transformed = pe_imports_get_feature.fit_transform(
        pe_imports_train_set
    )
    pe_section_names_train_transformed =
        pe_section_names_get_feature.fit_transform(
        pe_sections_train_set
    )

    # Get the composed train set X, using all the information gathered in the
        previous phase
    X_train = hstack([ngrams_feature_train_set, pe_imports_train_transformed,
                    pe_section_names_train_transformed,
                    csr_matrix(pe_number_of_section_train_set).transpose()])

    # Get RandomForest and train the model, set 100 estimators
    clf = RandomForestClassifier(n_estimators=100)
    clf = clf.fit(X_train, labels_train_set)
```

Listing A.11. MalwareDetectionSystem.py

This concludes the training section part.

At that time in order to evaluate the performance of the model, there is the necessity to test the model with the portion of initial data reserved for the test scope in a similar way to the one reported above. In this case, there is no training section since there is no fit operation to train the model.

```
pe_imports_test_set = []
pe_sections_test_set = []
pe_number_of_section_test_set = []
ngrams_feature_test_set = []
labels_test_set = []

for i in range(len(samples_test)):
sample = samples_test[i]
try:
    raw_file = read_file(sample)
    NGram_features = get_feature_from_sample(raw_file,
        most_common_ngrams_list, pattern_cardinality)
    pe = pefile.PE(sample)
    imports = get_imports(pe)
    n_sections = len(pe.sections)
    sec_names = get_section_names(pe)
    pe_imports_test_set.append(imports)
    pe_number_of_section_test_set.append(n_sections)
    pe_sections_test_set.append(sec_names)
    ngrams_feature_test_set.append(NGram_features)
    labels_test_set.append(labels_test[i])
except Exception as e:
    print("Error: " + str(e))

# Transform data-set
pe_imports_test_transformed = pe_imports_get_feature.fit_transform(
pe_imports_test_set
)
pe_section_names_test_transformed =
    pe_section_names_get_feature.fit_transform(
pe_sections_test_set
)

# Get the composed test set X, using all the information gathered in the
    previous phase
X_test = hstack([ngrams_feature_test_set, pe_imports_test_transformed,
            pe_section_names_test_transformed,
                csr_matrix(pe_number_of_section_test_set).transpose()]
           )
print("End testing\n")

print("Accuracy in testing: " + str(clf.score(X_test, labels_test_set)))
```

Listing A.12. MalwareDetectionSystem.py

A model like the one selected is able to provide an accuracy between 96% and 98% based on the random extracted samples.

Final review and consideration

As stated in this section, Machine Learning based malware detection classifier can be an excellent solution to deal with limitations and problems presented in the classical hash-based malware

detection techniques. Another possible improvement in this field is to consider also dynamic malware detection systems able to extract and classify entity information at run-time.

Among other information is possible to detect anomalous behaviour or well-known high-level execution pattern without considering the assembly structure of the executable. This can be particularly efficient in case of file-less threats which live in RAM without any executable files related to them. This kind of analysis requires a sandbox environment where to execute the malware consecutively collect and tracks all the operation performed by the malware. In this way is possible to perform two distinct operations: the first is to discriminate between malware dynamic operations and legit ones, and, more important discriminate among malware families such as ransomware, trojan, worm, Command-And-Control etc...

A.2 Machine Learning Based Network Analysis Detection

In the previous section, the artefact poses its attention on the device and user point of view in order to monitor and detect malicious activity concerning malware detection. In this section, the code exposed in the module `/pythonSrc/detection/NetworkDetectionSystem/` will present the basic of NBAD and UEBA.

Now, in order to be compliant with the main requirement expressed in the major standards, there is also the necessity to monitor and detect anomalies at the network level. In order to do that we can monitor two different vectors: web traffic and mail traffic. To monitor mail traffic in order to detect malicious mail we can use detect malware system used before in order to detect malicious executables, URLs and attachments. According to the principles illustrated in the previous sections, will be implemented a system to recognize spam and phishing mails.

A.2.1 Email Spam and Phishing Detection

Spam filtering is one of the most simple solutions that can be implemented by means of a decision-tree in a similar way to the one presented before for malware analysis. For sake of simplicity will be adopted a well-known repository called `spamassassin` which contains several examples of spam messages and ham (legal) messages. In the following code, snippets will be presented the solution adopted to implement a spam filter.

```
from os.path import isdir, isfile, join
from sklearn.model_selection import train_test_split
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import HashingVectorizer,
    TfidfTransformer
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn import tree
from os import listdir
import sys

# CHECK PARAMETERS
if len(sys.argv) != 3:
    sys.exit(-1)
if not isdir(sys.argv[1]):
    sys.exit(-2)
if not isdir(sys.argv[2]):
    sys.exit(-3)

# GET MAIL AND LOAD CONTENTS AND LABELS
ham_mail = sys.argv[1]
spam_mail = sys.argv[2]
emails_path = [(ham_mail, 1), (spam_mail, 0)]
all_mails_content = []
```

```
all_labels = []
for path, label in emails_path:
    samples = [sample for sample in listdir(path) if isfile(join(path,
        sample))]
    for email in samples:
        # print("Read file " + join(path, email))
        try:
            with open(join(path, email), "r") as file:
                buffer = file.read().replace("\n", "")
                buffer = str(buffer)
                all_mails_content.append(buffer)
                all_labels.append(label)
        except Exception as e:
            print("Error: " + str(e))
            continue

# SPLIT DATASET INT TRAIN AND TEST
X_train, X_test, y_train, y_test = train_test_split(all_mails_content,
    all_labels, test_size=0.2, random_state=11)

# PRE-PROCESS TEXTUAL DATA
natural_language_processing_then_dtree = Pipeline(
    [
        ("vect", HashingVectorizer(input="content", ngram_range=(1, 3))),
        ("tfidf", TfidfTransformer(use_idf=True)),
        ("dt", tree.DecisionTreeClassifier(class_weight="balanced"))
    ])

# TRAIN MODEL AND PRINT STATS
natural_language_processing_then_dtree.fit(X_train, y_train)
y_test_pred = natural_language_processing_then_dtree.predict(X_test)
print("STATS:\n")
print("Accuracy: " + str(accuracy_score(y_test, y_test_pred)))
print("Confusion-Matrix:\n" + str(confusion_matrix(y_test, y_test_pred)))
```

Listing A.13. SpamDetectionSystem.py

As already presented in the previous section, the first phase is to check the directory path where the program will fetch the samples for the training. Then, it is created the global message content and labels for both ham and spam content.

Then a splitting method is used to generate test and train subsets, processing the training section with a pipeline. The pipeline is composed of the following elements: an `HashingVectorizer` in order to select the common 3 NGrams (subject, predicates and complementary object), a `tfidf` (term frequency-inverse document frequency) engine in order to perform text and occurrence analytics, then the numerical results are analyzed by a decision tree. Finally, the program displays the accuracy and the confusion matrix:

```
Accuracy: 0.98953451356
Confusion Matrix :
    [[269, 4]
     [5, 561]]
```

Listing A.14. Output Result

Phishing URL's Detection

Now the artefact will present a more powerful machine learning program in order to detect malicious URL. Basically, by experience, a lot of phishing pages have some common features such as

the presence of hardcoded mail in the URL or in the HTML source, the use of shortening services, the presence of redirects to reload the official login page etc...

All of those attributes can be analyzed by means of a Random Forest in order to generate a model which is based on various estimators that are able to predict the nature of URLs. The following example omits the data collection and preprocessing which goes beyond the artefact and can be adapted specifically based on the environment and the attributes used inside the system. All the CSV involved in this section are not based on the URL itself, but they are in numerical form. Each CSV's entries can be expressed by a set of binary od ternary fields (ternary in case there is needed a guard threshold). For example columns, has_at (presence of symbol '@') has value 1 or 0 (present or not present). While having_long_url, based on two value T_INF and T_MAJ, can produce value as 1 (URL's length lower than a T_INF value), 0 (URL's length is between T_INF and T_MAJ) or 1 (URL's length is greater than T_MAJ).

All of those attributes are also related to a target value which is the label evaluated for this entry. In this case, will be used pandas in order to handle big data in an efficient way. The following snippet presents a simple way to implement a system like the one just presented:

```
from os.path import isfile
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix
import pandas as pd
import sys

# CHECK PARAMETERS
if len(sys.argv) != 3:
    sys.exit(-1)
if not isfile(sys.argv[1]):
    sys.exit(-2)
if not isfile(sys.argv[2]):
    sys.exit(-3)

# READ THE CSV GENERATING A DATAFRAME
dataset_train = pd.read_csv(sys.argv[1]) # TRAIN
dataset_test = pd.read_csv(sys.argv[2]) # TEST

# COMPOSE THE LABELS STARTING FROM THE "TARGET" COLUMN
target_train = dataset_train.pop("target").values
target_test = dataset_test.pop("target").values

# COMPOSE THE ATTRIBUTES FROM OTHER ELEMENT
attributes_train = dataset_train.values
attributes_test = dataset_test.values

# CREATE AND EVALUATE MODEL
random_forest_classifier = RandomForestClassifier(n_estimators=100)
random_forest_classifier.fit(attributes_train, target_train)
target_predicted = random_forest_classifier.predict(attributes_test)
print("STATS:\n")
print("Accuracy: " + str(accuracy_score(target_test, target_predicted)))
print("Confusion-Matrix:\n" + str(confusion_matrix(target_test,
    target_predicted)))
```

Listing A.15. PhishingUrlDetectionSystem.py

A system like the one proposed is able to produce a good level of accuracy according to the data quality related to the two CSV files. Again the model implemented in the `scikitlearn` module can be very powerful if and only if the data are consistent.

Email Protection is a good entry point in order to detect security issues involved within the mail scope that is responsible for a good portion of the company's employee operations. By the

way, this is not an oracle, and it is able to prevent other threats provided in the network scope for example. In order to do that there is the necessity to monitor also the network in order to detect possible malicious or anomalous behaviour performed by the user. This field is known as UEBA (User and Entity Behavioral Analytics) which is able to collect information to design a normal model of the user behaviour, then in case of strange operations can detect anomalies. It will be presented in the next section.

A.2.2 Network's Anomaly Detection

In this section will be presented the main strategies to detect anomalies events in the system, those kinds of solutions are especially used in network and user behaviour monitoring. There can be two main solutions: the first is to track user common activity in a protected scope in order to create a model which is able to highlight points that have a distance greater than a certain threshold from the centroid, otherwise is possible to detect common malicious patterns in terms of HTTP requests, traffic, user actions and many more in order to detect anomalies based on similarity with those malicious well-known patterns.

These two approaches can reach the same result but the first is not tolerant of false-positive, this means that an anomalous activity can be not related to malicious patterns or activities. The second discussed instead is able to detect malicious patterns minimizing the false positive rate according to the dta quality, at the same, is more difficult to obtain since malware and malicious pattern are difficult to be found in the wild. A bracket must be open in order to deal with 0-day attacks: an anomaly detection system is able to detect un-seen malicious and anomalous behaviour related to a 0-day attack, instead a pattern recognizer is not able to detect a 0-day attack from the moment that its behaviour and patterns are not known. In conclusion, those two techniques must be merged in order to create a layered model which keeps into account all the benefit discussed above, specifically, an Anomaly detection system can be implemented with Isolation Forest to identify anomalies while pattern recognition can be implemented with Random Forest. In the next subsections will be presented the main technical and implementation aspects concerning these two approaches.

Network Behavioral Analytics, Dataset and Data Challenges

Concerning the aspects related to the system complexity explained in the First Chapter, must be considered this impact in the detecting. Nowadays complex networks can generate a huge traffic amount which should be analyzed in order to match with possible malicious patterns. For this reason solution as SIEM and IDS help analysts monitor the internal perimeter and at the same time are able to provide useful hints in order to respond and mitigate attacks as fast as possible.

In order to do that, machine learning can offer all its solutions to detect malicious patterns and the similarity of patterns. As already stated in previous sections, those kinds of solutions should be trained with dataset and prediction evaluations are performed based on the data accuracy involved in the model construction. To capture the network traffic should be assured also the physical resources to support an appropriate stream's bandwidth reception as well as topological in order to capture important and strategic traffic among internal nodes, VLAN, Proxies and other network aspects. In that way, a virtual sensor is able to collect all the IP traffic, based on the upper layer of the logic which governs a SIEM are able to correlate packets in order to reconstruct a structured network transmission among hosts. In solutions like the one presented here are pre-built patterns that are assumed as malicious or strange based on the scope and good practices in terms of network, time, space, and configuration aspects. In a network, there can be several categories of traffic such as vital protocols DNS, HTTPS, SMTP or specific ones like Kerberos, smb etc... Possible rules or commonly known policies can be Impossible Travel Logon, Abnormal Login, and Tor Browsing. By the way, is not possible to perfectly map all the malicious behaviours into a deterministic choice and for this reason, machine learning tries to match attack patterns with network traffic for anomaly detection based on normal network flow.

In the next section will be presented how to treat the training aspect of a machine learning model to network anomaly detection. For this purpose will be used the NLS-kdd dataset. It is

a widely used dataset used to train and construct IDS, and it contains intrusions simulation in a military network. In the following snippet will be explained how is possible to use an Isolation Forest to detect anomalies, start with reading and preprocessing the dataset.

```
# CHECK PARAMETERS
if len(sys.argv) != 2:
    sys.exit(-1)
if not isfile(sys.argv[1]):
    sys.exit(-2)


# READ THE CSV GENERATING A DATAFRAME
kdd_nls = pd.read_csv(sys.argv[1])
y = kdd_nls["label"].values
kdd_nls["label"] = kdd_nls["label"].apply(binary_classifier)
# Ratio of anomalies respect to normal items
y = kdd_nls["label"].values
most_common_labels = Counter(y).most_common()
print(most_common_labels)
res = most_common_labels[1][1] / (most_common_labels[0][1] +
    most_common_labels[1][1])
print(f"Ratio of #abnormal/#total: {res}")
```

Listing A.16. NetworkBehavioralDetection.py

The script read the content of the CSV input file thanks to pandas utilities, consequently, the dataset is processed by means of a binary classifier. The data provided by the KDD dataset collect a set of session information such as TCP port, sent bytes, duration as well a label describing the activity. It discriminates against normal operation and other types of traffic such as Nmap traffic, smurf dossing and others. For sake of simplicity, the script will collect two types of categories: normal or abnormal sessions.

Then the data are ready to be divided into test and train partition

```
label_encoder_dict = dict()
for i in kdd_nls.columns:
    if kdd_nls[i].dtype == 'object':
        label_encoder_dict[i] = LabelEncoder()
        kdd_nls[i] = label_encoder_dict[i].fit_transform(kdd_nls[i])
kdd_nls_normal = kdd_nls[kdd_nls["label"] == 0]
kdd_nls_abnormal = kdd_nls[kdd_nls["label"] == 1]

y_normal = kdd_nls_normal.pop("label").values
y_abnormal = kdd_nls_abnormal.pop("label").values
X_normal = kdd_nls_normal.values
X_abnormal = kdd_nls_abnormal.values

X_normal_train, X_normal_test, y_normal_train, y_normal_test =
       train_test_split(X_normal, y_normal, test_size=0.3,
           random_state=11)
X_abnormal_train, X_abnormal_test, y_abnormal_train, y_abnormal_test =
       train_test_split(X_abnormal, y_abnormal, test_size=0.3,
           random_state=11)

X_train = np.concatenate((X_normal_train, X_abnormal_train))
X_test = np.concatenate((X_normal_test, X_abnormal_test))
y_train = np.concatenate((y_normal_train, y_abnormal_train))
y_test = np.concatenate((y_normal_test, y_abnormal_test))
```

Listing A.17. NetworkBehavioralDetection.py

In the first loop, all the categorical features are transformed into numerical representation, and then results are separated into normal and abnormal behaviour. This algorithm works well also in highly unbalanced situations, for the results presented above is used a dataset where the ratio of abnormal situations is 4,8%.

```
isolation_forest = IsolationForest(contamination=res)
isolation_forest.fit(X_train)

decision_score_normal_train =
    isolation_forest.decision_function(X_normal_train)
decision_score_abnormal_train =
    isolation_forest.decision_function(X_abnormal_train)

# Normal
plot.figure(figsize=(20, 10))
plot.hist(decision_score_normal_train, bins=50)
plot.show()
# Abnormal
plot.figure(figsize=(20, 10))
plot.hist(decision_score_abnormal_train, bins=50)
plot.show()

cutoff = float(input("set cuf-off: "))
print(Counter(y_test))
print(Counter(y_test[cutoff >
    isolation_forest.decision_function(X_test)]))
```

Listing A.18. NetworkBehavioralDetection.py

In the code is instantiated the isolation forest in order to detect anomalous points:

The figures A.1 and A.2 are able to suggest a good horizontal threshold, in this case a good value can be 0.015 which is able to train the model in an accurate way. The output of the program are the following:

```
tot: 41237
[('normal', 39247), ('back', 1098), ('apache2', 794), ('neptune', 93),
    ('phf', 2), ('portsweep', 2), ('saint', 1)]
Ratio of #abnormal/(#normal + #abnormal)): 0.04825763270848995
Counter({0: 11775, 1: 597})
Counter({1: 590, 0: 55})
```

As the snippet shows the model is able to provide an accuracy greater than 90% in recognition of anomalous patterns according to dataset information.

Improvement and Other Considerations

The example just presented can be a good base in order to detect isolated points according to a feature set classification. By the way, each company can evaluate its peculiar models based on more accurate data: the internally generated ones.

That kind of model can be easily trained on well-known threats but can be also involved in featuring and analyzing user behaviour in order to isolate some critical cases. In order to do that an enterprise has to create and engineer some features extractors able to collect user information. In cybersecurity scopes can be an email sent, the file created, copied, or exported on USB devices, as well as average CPU's percentage and memory usage. Then a similar model is able to detect anomalies in all of these tasks. At the same time, Isolation Forests are efficient, scalable and expressive algorithms, specifically, the time complexity of an Isolation Forest is linear since it is a recursive problem based on binary trees as Random Forest as well.

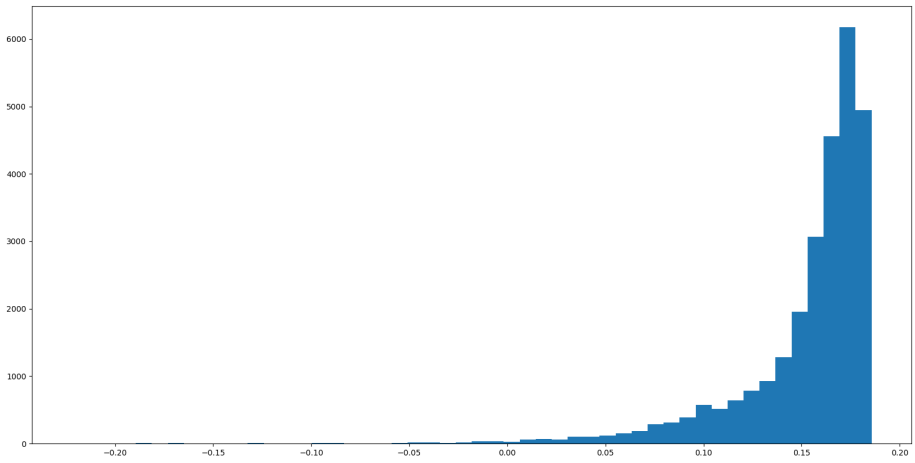

Figure A.1. normal values

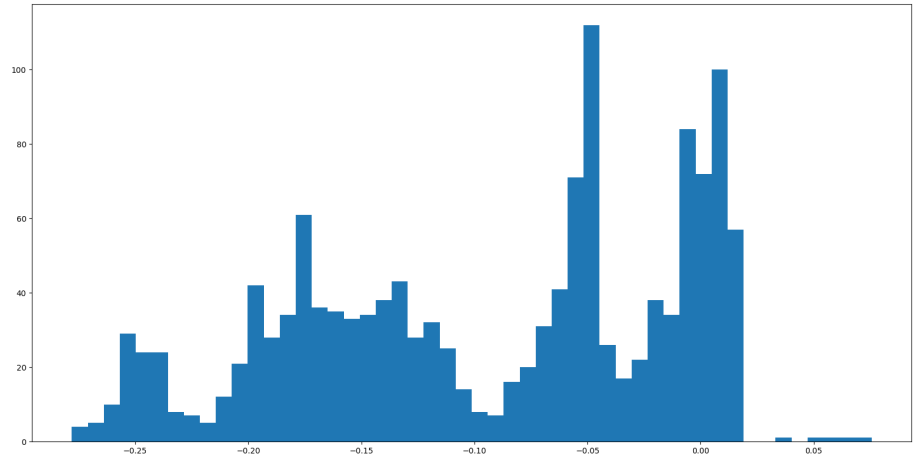

Figure A.2. abnormal values

The previous example works on a semi-supervised approach since the dataset was provided by labels, but at the same time Isolation Forest can be used also in the Unsupervised Approach. Latter is quite useful when no labels are available to detect anomalous points based on the most representative characteristics of the objects analyzed.

Solutions like that can be easily integrated in the detection system in order to detect anomalous situations and are easily trained on log data provided by perimeter device such as firewalls, WAF, routers switch as well as logging systems. This data availability is the main scope of the detection phase which can be easily integrated with such kind of checks so that the system can correctly implement and match the requirements provided by the global standards. Those kinds of considerations will be described in the section.

A.3 Automations, Hardening Strategies and Technological Solutions

In this section will be presented a simple scenario where data collection and correlation are used in order to detect some possible improvements in the system. Those improvements are strictly related to the definition of IoC or possible malicious scenarios based on data correlation. During my experience in Nais, I have had access to a huge amount of data. All those data are generated by logs from non-security devices. In the next section will be presented data correlation performed on Azure interactive sign in a big organization. I have developed a simple data correlation engine with Python3 in order to detect possible IoC and highlight frequently targeted users.

A.3.1 Correlation and Detection

The security of a digital perimeter must be protected to avoid illegal access from outside and, more important, from the inside. For this reason, it exists some specific devices such as firewalls and Web Application Firewalls (WAF) able to intercept malicious request from an external client before they reach the internal. In the design of that architecture becomes highly relevant the use of a whitelist approach where the system accepts only specific traffic according to some specific policies. Those policies can be of several types:

- IP ranges: grant access to those IPs since they are in a VPN.
- Accept Headers: accept only a specific header in an HTTP request (such as Authorization with JWT).

A similar approach can be also useful to handle Cross Site Forgery Requests through specific headers to prove the authenticity of data submitted by the client browser. Inspection of packets payload, regarding cryptographic operations which can obfuscate the real payload also for the monitoring system. If the firewall is not able to perform a security check due to invalid, or encrypted data, the latter must be discarded. For those reasons useful architecture based on the concept of Secure Gateway Architecture where TLS inspection can be performed to access decrypted data. Another simple solution can be blocking writing HTTP methods such as DELETE or PUT to avoid data modification.

Log, Events and IoC

The one just presented is a general idea to protect a company network from outside malicious operations based on popular exposed service such as HTTP, SMTP, IMAP, SSH etc... By the way, this is not the only way to protect a company's digital domain. Another way is based not only on a reactive approach such as monitoring. Monitoring is the art of looking for suspicious activity inside the company network to try to find out possible issues. As it was discussed in the previous chapter, the log is a piece of information relevant in order to keep a trace of operations performed by devices or applications. Operations can be derived from the Machine Learning model: an example can be the log recorded to keep track of user sign-in on an application. The collection of all relevant logs in the system is an excellent starting point to correlate those logs and consecutively investigate them looking for suspicious activities. From the moment that logs are generated also in the intranet, it is a good approach to monitor the situation inside the network and not just from outside. To highlight an attack pattern, logs are correlated according to some policies and if a policy is triggered it is considered a possible event related to an attack. The analyst has the task to investigate them in order to apply possible mitigations and take actions regarding the security event or, to declare the event as a False Positive and understand why the system detects a false positive, maybe due to some side conditions.

In case of a real attack, a pattern should be used in order to detect possible IoC (Index of Compromise) and to contain problems. IoCs are entities which are associated with some malicious activity such as a mail sender, an IP, a process and son on...

To understand better the scenario, I want to make a practical example: imagine a brute force attack where an external malicious client tries to guess a password related to an application user to gain privileged access to the system. Consider an alphanumeric password of 10 characters. To brute force the password an attacker has to attempt 36 to the power of 10 possible permutations of all numbers and characters. A possible policy for brute force in this situation can be the following: If login logs highlight an external IP which is performing more than 1 attempt/sec in a specific interval time according to computational resources the event correlation can be matched as a possible brute force. It must guarantee that the possible attempts performed by the hostile device do not threaten the password's strength.

As explained in the previous chapter, once a security-relevant event is triggered a consecutive phase of mitigation is required according to its severity impact. In this case, the analyst has to investigate, then he should know the reason behind this security event: the first is to investigate the IP behind the request and its reputation. Some possible scenarios are explained in the following section:

- The IP is an IoC: threat data can prove that entity is associated with a malicious operation. A possible mitigation is to block the IP in case of persistent activity according to password resistance.
- The IP is internal: and the analyst has to investigate the reason for that behaviour. Maybe the involved user may have performed a password change and on some other internal device, the old password is cached on the application which continues to perform login with a disaligned password. In the worst scenario, this can be an IoC of possible intranet-compromised nodes. In order to implement a resilient solution companies has to consider also intranet networking where connections are generally seen as less important and taken as granted.
- The IPs are external: this is one of the more complex situations. In that case, a set of IPs is responsible for brute force. Typically, in this kind of scenario, a finer level of correlation is required in order to understand the relationship among the set of IPs, perhaps related to the same botnet or the same criminal organizations. In that case, alerts are generated but until the analyst performs the blocks the Bruteforce persists in threatening the robustness of the password, in this situation due to the specificity of the attack some automated actions can be taken. In those situations can be imposed a Quarantine for such IPs or in a simpler way, can be instantiated a Global Application Lock for a reasonable amount of time.

A.3.2 Technological Solutions

Nowadays major manufacturers such as Microsoft, PaloAlto, Cisco, and VMware provide several solutions to support clients to satisfy audit and local compliance aspects by integrating monitoring systems inside their products. In the next sections will be presented a basic scenario in which is required to monitor the Active Directory sign-in logs to detect malicious operations, and more important prioritize the MFA activation of critical users, according to their number of failed logon monitored by the system.

Microsoft Azure

Azure is developed by Microsoft, and it implements cloud computing solutions such as IaaS, PaaS, SaaS and mBaaS. It can be easily used within the Active Directory to implement Identity Management and Group Management System of all users present in a specific domain. From the moment that it can be used in such a way, the platform can produce logs related to user access and sign-in attempts as well as audit operations. All of this data is available to the analyst who can correlate information according to big data analytics to figure out possible improvement based on the result.

Among the services provided by Azure, there is the opportunity to export data related to users signing in. It can be possible through two methods: the first is to use the APIs provided by Microsoft which requires the opposite licence, otherwise is possible to export a CSV. The licence

used by the client did not satisfy the APIs integration so the only way to export data is using a CSV file. The CSV header is composed in the following way:

```
Date (UTC), Request ID, Correlation ID, User ID, User, Username, User
    type, Cross tenant access type,
Incoming token type, Authentication Protocol, Unique token identifier,
    Client credential type, Application,
Application ID, Resource, Resource ID, Resource tenant ID, Home tenant
    ID, Home tenant name, IP address, Location,
Status, Sign-in error code, Failure reason, Client app, Device ID,
    Browser, Operating System, Compliant, Managed,
Authentication requirement, Sign-in identifier, IP address, Autonomous
    system number, Flagged for review, Token issuer type,
Incoming token type, Token issuer name, Latency, Conditional Access
```

As the snippet shows, each entry presents a lot of information. During this experience, I have been assigned to implement a script able to detect possible frequently targeted users and to detect the frequent foreign IPs which attempts to sign in as one of the domain users guessing the password. The final client expects their users to access from a location in Italy, by the way, possible foreign accesses are permitted due to business travel. In this scenario, we expect to find a nonmalicious IP, maybe related to some foreign Internet Service Provider. To perform this check, we use Microsoft intelligence (already integrated into Azure ACL) and AbuseIPDB APIs. The next section will present the crucial routine in Python Script.

A.3.3 MFA Prioritization Python Script

In this section, all the script refers `/pythonSrc/hardening/MFAHardeningAndPrioritizationModule`. The scripts are used to prioritize the MFA activation of an organization also with several thousands of users. The prioritization is based on the more exposed users to brute force. In order to highlight users with this proclivity, the strategy implements a counter of failed access for a user weekly.

This script is also able to correlate the involved user with his MFA status according to a dump which contains all the users provided by MFA strategy authentication enforced. Follow the Python code able to implement such requirements:

```
import xlsxwriter
from datetime import datetime
import sys, os
from operator import itemgetter
from tkinter import messagebox

from script import signInAnalyzer
from script import failedAccessCounter_withFalsePosEvalution


def evaluateEvent(data: tuple):
    severity = 'LOW'
    user = data[0]
    totFailAccess = data[1]
    activeSyncFailAccess = data[2]
    mfaStatus = data[3]
    if (totFailAccess - activeSyncFailAccess) > 14:
        if mfaStatus == 'YES':
            severity = 'MEDIUM'
        else:
            severity = 'HIGH'
    return severity
```

```
def removeFiles():
    os.remove('./Failure.csv')
    os.remove('./report.csv')
    os.remove('./parsedFile.csv')
    os.remove('./Success.csv')


headerSignIn = [{'header': 'IP'}, {'header': 'DOMAIN'}, {'header':
    'REPUTATION'},
               {'header': 'CATEGORIES'}, {'header': 'INVOLVED_USER_COUNT'},
                   {'header': 'TOTAL_COUNT'}]
headerFailedAttempts = [{'header': 'USERNAME'}, {'header': 'TOTAL FAILED
    ACCESS'},
                        {'header': 'ACTIVE SYNC FAILED ACCESS'}, {'header':
                            'MFA ENABLE'}]

flaggedForReview = []

if len(sys.argv) != 2:
    sys.exit('ERROR: invlaid number of parameters')

try:
    # Set up workbook
    print('\nStarting up...\n')
    if not (os.path.exists('./report/')):
        os.mkdir('./report/')
    fileName = "report.xlsx"
    workbook = xlsxwriter.Workbook('./report/' + fileName)
    redFormat = workbook.add_format({'bg_color': '#FFC7CE', 'font_color':
        '#9C0006'})
    style = workbook.add_format()
    style.set_text_wrap()
    style.set_align('vcenter')
    success = workbook.add_worksheet('Success')
    failure = workbook.add_worksheet('Failures')
    userAttempts = workbook.add_worksheet('UserAttempts')

    # SUCCESS
    print('\nAnalyzing success sign in\n')
    success.set_column(0, len(headerSignIn) - 1, 40)
    for i, dataS in enumerate(sorted(signInAnalyzer.signInReporter('S',
        sys.argv[1]), key=itemgetter(5), reverse=True)):
        success.write_row(i + 1, 0, dataS, style)
    success.add_table(0, 0, i + 1, len(headerSignIn) - 1, {'columns':
        headerSignIn})

    # FAILURES
    print('\nAnanalyzing failed sign in\n')
    failure.set_column(0, len(headerSignIn) - 1, 40)
    for i, dataF in enumerate(sorted(signInAnalyzer.signInReporter('F',
        sys.argv[1]), key=itemgetter(5), reverse=True)):
        failure.write_row(i + 1, 0, dataF, style)
    failure.add_table(0, 0, i + 1, len(headerSignIn) - 1, {'columns':
        headerSignIn})
```

```
    # USER-ATTEMPTS
    print('\nAnalyzing MFA status\n')
    userAttempts.set_column(0, len(headerFailedAttempts) - 1, 45)
    for i, dataUA in enumerate(
            sorted(failedAccessCounter_withFalsePosEvalution.failedAccessReporter(),
                key=lambda x: (x[3], x[1]),
                   reverse=True)):
        tmp = evaluateEvent(dataUA)
        if tmp != 'LOW':
            flaggedForReview.append({'username': dataUA[0], 'severity': tmp})
        userAttempts.write_row(i + 1, 0, dataUA, style)
    userAttempts.add_table(0, 0, i + 1, len(headerFailedAttempts) - 1,
        {'columns': headerFailedAttempts})
    userAttempts.conditional_format(0, len(headerFailedAttempts) - 1, i + 1,
        len(headerFailedAttempts) - 1,
                                {'type': 'text', 'criteria': 'begins with',
                                    'value': 'NO', 'format': redFormat})

    # Close workbook and remove files
    workbook.close()
    print('\nRemoving temporary files...\n')
    removeFiles()
    with open('./report/flaggedForReview.txt', 'a') as file:
        file.write('GENERATED ON: ' + datetime.today().strftime("%m-%d-%Y") +
            '\n')
        if len(flaggedForReview) > 0:
            for item in flaggedForReview:
                file.write(item['username'] + ' --- ' + item['severity'] +
                    '\n')
            file.write('\n\n')
        file.close()
    messagebox.showinfo(title='Severity Review',
                        message='Ci sono %d eventi flaggati per la Review.' %
                            len(flaggedForReview))
    print('\nEnding...')
except Exception as e:
    sys.exit(e)
```

Listing A.19. getMfaStatus.py

The script is the main entry point, and it analyses the success and failures of sign-in by means of `signInAnalyzer.signInReporter` that provides evidence about Locations, source IP, methodologies, protocol etc.... Using the `xlsxwriter` module previously explained the script produces a report with three worksheets:

- Success: it highlights global success access in terms of source IP, reputation provided by AbuseIPDB, and global count of attempts.
- Failure: it highlights global failed access in terms of source IP, reputation provided by AbuseIPDB, and global count of attempts.
- MFA Review: For each user is displayed the number of failed attempts, reporting false positives due to ActiveSync (will be deprecated by MS), and MAF status. All data presented on this worksheet will be sorted in descending order in order to highlight users that have a lot of failed sign-in attempts and no MFA enforced.

The script which governs those processes is reported below:

```
import sys, csv, os
```

```
from abuseIPDB import analyzeIp

ipList = {}
outRes = []
status = 'Failure'


def printResFormatted(collection):
    for item in collection:
        print(item)
        print('\t' + str(collection[item]))

def signInReporter(type: str, inputFile: str):
    if(type == 'F'):
        status = 'Failure'
    elif(type == 'S'):
            status = 'Success'
    else:
        print('[INFO]: using default status = Failures')

    try:
        with open(inputFile, 'r', encoding = 'utf-8') as tmpFile:
            buff = tmpFile.read()
            buff = buff.replace(u'\ufeff', '')
            buff = buff.replace('\t', ',')
            out = open('parsedfile.csv', 'w')
            out.write(buff)
            tmpFile.close()
            out.close
        with open('parsedfile.csv') as csvFile:
            csvReader = csv.DictReader(csvFile, delimiter = ',')
            for row in csvReader:
                if(row['Status'] == status):
                    ip = row['IP address']
                    affectedUser = row['Username']
                    if ip in ipList:
                         if affectedUser in ipList[ip]['userInvolved'].keys():
                             ipList[ip]['userInvolved'][affectedUser] =
                                 ipList[ip]['userInvolved'][affectedUser] + 1
                         else:
                             ipList[ip]['userInvolved'][affectedUser] = int(1)
                    else:
                        if (status == 'Failure' and ('was blocked' in
                            row['Failure reason'] or 'locked' in row['Failure
                            reason'])):
                            [reputation, domain, categories] = ['Malicious IP:
                                blocked by Microsoft-AZURE or Locked Account due
                                to Many Failure Attempts', '-', row['Failure
                                reason']]
                        else:
                            [reputation, domain, categories] = analyzeIp(ip)
                            #[reputation, domain, categories] = ['rep',
                                'domain', 'cat']
                        ipList[ip] = {'reputation' : reputation, 'domain':
                            domain, 'tag' : categories, 'userInvolved' :
                            {affectedUser : int(1)}}
    except Exception as e:
```

```
        sys.exit(e)
    tmpFile.close()
    #printResFormated(ipList)
    try:
        with open(status + '.csv', 'w', newline='', encoding='utf-8') as
            report:
            writer = csv.writer(report)
            header = ['IP', 'DOMAIN', 'REPUTATION', 'CATEGORIES',
                'INVOLVED_USER_COUNT', 'TOTAL_COUNT']
            writer.writerow(header)
            for item in ipList:
                buff = ''
                tmp = 0
                for idx,user in enumerate(ipList[item]['userInvolved']):
                    buff += str(user) + '<' +
                        str(ipList[item]['userInvolved'][user]) + '>\n'
                    tmp += ipList[item]['userInvolved'][user]
                    buffCat = ''
                if(isinstance(ipList[item]['tag'], str)):
                    buffCat = ipList[item]['tag']
                else:
                    for idx,cat in enumerate(ipList[item]['tag']):
                        buffCat += cat + ', '
                    buffCat = buffCat[0: len(buffCat) - 2]
                writer.writerow([item ,ipList[item]['domain'],
                    ipList[item]['reputation'], buffCat , buff[0: len(buff) -
                    1], tmp])
                outRes.append([item ,ipList[item]['domain'],
                    ipList[item]['reputation'], buffCat , buff[0: len(buff) -
                    1], tmp])
            return outRes
    except Exception as e:
      sys.exit('2, ' + str(e))
    report.close()
```

Listing A.20. failedAccessCounterWithFalsePosEvalution.py

The support module process all the entry in the signing logs CSV file, collecting information by means of source IP. The source IP is correlated to the user on which it attempts a log-on and which quantity, then using IP as the key is related also to a global sum of the global attempts.

The IP is also related to its reputation provided by AbuseIPDB thanks to external module `abuseIPDB`, which will be explained in the next section.

IoC Python Script Detection

From the moment that Azure provides also the IP which tries the interactive sign-in, we can correlate data again in order to detect possible IoCs. The solution that will be presented is supported by AbuseIpDB in order to collect the reputation of an IP.

Follows the code able to interact with AbuseIpDB API in order to retrieve the information about an IP.

```
import requests, constants

"""
    Get request from abuseIPDB
"""
```

```
categories = {1: 'DNS Compromise', 2: 'DNS Poisoning', 3: 'Fraud Orders', 4:
    'DDoS Attack', 5: 'FTP Brute-Force', 6:'Ping of Death', 7: 'Phishing',
             8: 'Fraud VOIP', 9 : 'Open Proxy', 10 : 'WebSpam', 11 : 'Email
                 Spam', 12: 'Blog Spam', 13: 'VPN IP', 14 : 'Port Scan', 15:
                 'Hacking',
             16 : 'SQL injection', 17: 'Spoofing', 18 : 'BruteForce', 19:
                 'Bad Web Bot', 20 : 'Exploited Host', 21 : 'WebApp Attack',
                 22 : 'SSH',
             23 : 'IoT Targeted'}

def analyzeIp(IP):
    #GET approach
    try:
        print('processing: ' + IP)
        url = "https://www.abuseipdb.com/api/v2/check"
        response = requests.get(url, headers={
             'Accept':'application/json',
             'Key' : constants.IP_ABUSEIPDB_API_KEY,
             'User-Agent': 'Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36
                 (KHTML, like Gecko) Chrome/41.0.2228.0 Safari/537.36',
             }, params = {
                 'maxAgeInDays' : constants.IP_ABUSEIPDB_API_DAY,
                 'ipAddress' : IP,
                 'verbose' : 'true'
                 })
        if response.status_code == 200:
            tmp = []
            for item in response.json()['data']['reports']:
                for cat in item['categories']:
                    if int(cat) not in range(1, 23):
                        continue
                    if categories[cat] not in tmp:
                        tmp.append(categories[cat])
            #print([str(response.json()['data']['abuseConfidenceScore']) +
                '%', response.json()['data']['domain'], tmp])
            return [str(response.json()['data']['abuseConfidenceScore']) +
                '%', response.json()['data']['domain'], tmp]
        print('Unable to process ip: ' + IP + ' - status code: ' +
            str(response.status_code) + '\n')
        return('', '', '')
    except Exception as e:
        print(e)
```

Listing A.21. abuseIPDB.py

The snippet present a tool able to gather reputation according to AbuseIPDB provider and `request` module. The information gathered concerns the abusescore, the categories of offence and the base domain.

A.4 Incident Response Automations

In this section will be presented the code needed to configure effective automation for an incident response using solutions provided by VMWare CarbonBlack and VSphere and Microsoft Graph API. The first is an EDR Cloud Platform able to monitor and respond to security events, while the second is the APIs set provided by Microsoft in order to interact directly with Azure cloud services. The main action as reported in the theoretical part consists of incident response

operations such as blocking users, isolating devices, deleting by hash etc... Please refer to folder `/response/IncidentResponseFlow/`.

A.4.1 Flows and Architecture

The core system will be developed by means APIs interaction as stated in the section 3.3. The system will use a central node which can be viewed as a SOAR integrable with the core system. In order to provide a widely usable support to incident response the principal code concerns a library which implements all the operations required in the playbook discussed. The main flow is the one described below:

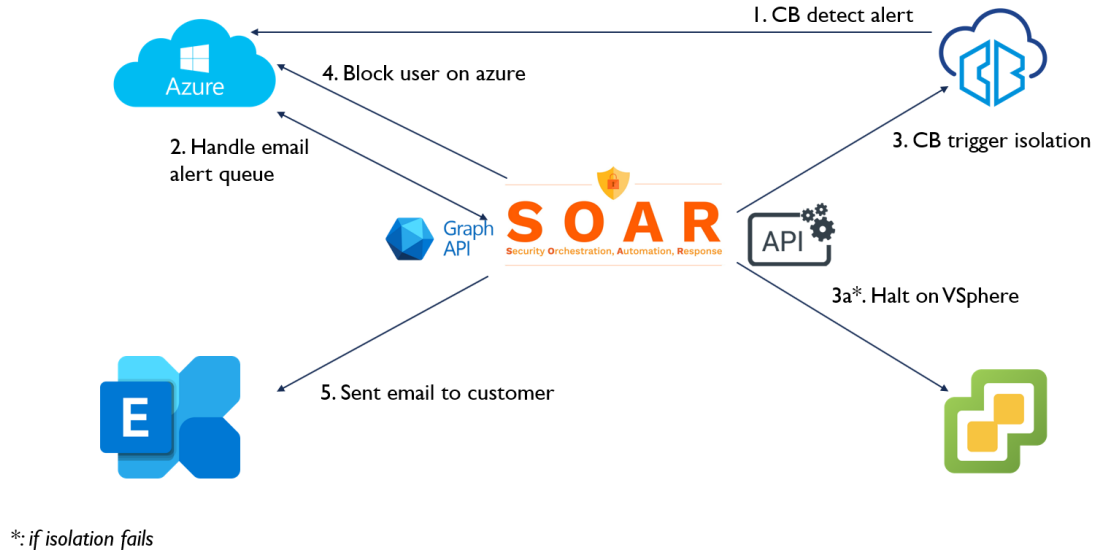

Figure A.3. IR Flow

The code presented in the next section will be related to the infrastructure presented above, and in order to maintain a modular the system will be implemented a support library in order to map every operative flows (arrow in the image) with a specific API call between the systems involved. As it is reported in the image will be involved three different systems that will be argued in the next sections.

A.4.2 Azure and MS Graph Interaction

Azure is a Cloud based Active Directory environment widely used nowadays in small and big organizations, and it can be integrated with On-Prem Active Directory as well. In order to interface with this architecture, Microsoft provides a set of API thank its Graph Services where the authentication process is performed by the AD user in order to inherit its privileges and emails.

Please refer to the B chapter to configure the application on Azure. The first thing to do after application creation is to export the information to bind the application and Azure. This is reported in the *[azure]* section in the `constants.cfg` inside the `IncidentResponseFlow` module:

```
[azure]
APP_ID = APP-ID
DIRECTORY_ID = TENANT-ID
SECRET_ID = SECRET-ID
AUTH_TENANT = common
SCOPES = Mail.ReadWrite User.ReadWrite.All Mail.Send
```

```
EMAIL_SUBJECT_OK = Automation Response Review - OK
EMAIL_SUBJECT_ERROR = Automation Response Review - ERROR
EMAIL_RECIPIENT = abc.cba@abc.com
EMAIL_TEXT_OK = Hi,<br>there are no uncompleted tasks in response automation,
    please refer attachment for overview.
EMAIL_TEXT_ERROR = Hi,<br>there are uncompleted tasks in response automation,
    follow incident id:<br>
EMAIL_FOOTER = <br><br><i>This email is sent from an unmonitored mailbox,
    please does not reply.</i>
```

Listing A.22. constants.cfg

Once provided those config will be possible to initiate a connection with Azure services creating a GraphServices Class in Python as reported in the `GraphService.py`:

```
class GraphService:
    settings: SectionProxy
    dev_code_credential: DeviceCodeCredential
    graph_client: GraphClient
    client_credential: ClientSecretCredential
    access_token: AccessToken

    def __init__(self, config: SectionProxy):
        self.settings = config
        app_id = self.settings['app_id']
        directory_id = self.settings['directory_id']
        scopes = self.settings['scopes'].split(' ')
        self.dev_code_credential = DeviceCodeCredential(app_id,
            tenant_id=directory_id)
        self.graph_client = GraphClient(credential=self.dev_code_credential,
            scopes=scopes)

    def get_access_token(self):
        scopes = self.settings['scopes']
        token = self.dev_code_credential.get_token(scopes)
        self.access_token = token
        return self.access_token.token

    def compose_email(self, alert_review):
        body = ''
        if len(alert_review) == 0:
            body += self.settings['email_text_ok']
            subject = self.settings['email_subject_ok']
        else:
            body += self.settings['email_text_error']
            body += '<ul>'
            for item in alert_review:
                body += f'<li><b>{item}</b></li>'
            body += '</ul>'
            subject = self.settings['email_subject_error']
        body += self.settings['email_footer']
        return subject, body

    def send_mail_review(self, file_path, alert_review):
        try:
            file_base64 = base64.b64encode(open(file_path, 'rb').read())
            mime_type = mimetypes.guess_type(file_path)[0]
            attachment = {'@odata.type': '#microsoft.graph.fileAttachment',
                          'contentBytes': file_base64.decode('utf-8'),
```

```
                    'contentType': mime_type,
                    'name': os.path.basename(file_path)}
        sub, body = self.compose_email(alert_review)
        email_msg = {'message': {'subject': sub,
                                 'body': {'contentType': 'HTML', 'Content':
                                     body},
                                 'toRecipients': [
                                     {
                                         'emailAddress': {
                                             'address':
                                                 self.settings['email_recipient']
                                         }
                                     }
                                 ],
                                 'attachments': [attachment],
                                 'importance': 'low' if len(alert_review) ==
                                     0 else 'high'
                                 },
                     'SaveToSentItems': 'true'}
        resp = self.graph_client.post('/me/sendmail',
                                      headers={'Content-Type':
                                          'application/json'},
                                      data=json.dumps(email_msg)
                                      )
        print(resp)
        return True
    except Exception as e:
        print(f"Error sending mail {str(e)}")
        return False

def get_target_mailbox(self):
    endpoint_get_mail_folders = '/me/mailFolders/inbox/childFolders'
    processed_id = ""
    processed_with_error_id = ""
    for mail_folder in self.graph_client.get(endpoint_get_mail_folders,
                                             headers={'Content-Type':
                                                 'application/json'}).json()['value']:
        if mail_folder['displayName'] == "Processed":
            processed_id = mail_folder['id']
        if mail_folder['displayName'] == "Error":
            processed_with_error_id = mail_folder['id']
    if processed_id == "":
        resp = self.graph_client.post(endpoint_get_mail_folders,
                                      headers={'Content-Type':
                                          'application/json'},
                                      data=json.dumps({
                                          'displayName': 'Processed',
                                          "isHidden": False
                                      })).json()
        processed_id = resp['id']
    if processed_with_error_id == "":
        resp = self.graph_client.post(endpoint_get_mail_folders,
                                      headers={'Content-Type':
                                          'application/json'},
                                      data=json.dumps({
                                          'displayName': 'Error',
                                          "isHidden": False
```

```
                                    })).json()
            processed_with_error_id = resp['id']
        return processed_id, processed_with_error_id

    def get_new_alert_from_mail(self):
        endpoint_inbox = '/me/mailFolders/inbox/messages'
        select = 'from,receivedDateTime,subject'
        order_by = 'receivedDateTime DESC'
        alerts_id = []
        request_url =
            f'{endpoint_inbox}?$select={select}&$top={100}&$orderBy={order_by}'

        for message in self.graph_client.get(request_url).json()['value']:
            if "Carbon Black Cloud Alert" in message['subject']:
                alerts_id.append({'alert_id': message['subject'].split(" -
                    ")[1], 'message_id': message['id']})
        return alerts_id

    def move_processed(self, message_id, folder_id):
        endpoint_message_move = "/me/mailFolders/inbox/messages/{id}/move"
        self.graph_client.post(endpoint_message_move.format(id=message_id),
                               headers={'Content-Type': 'application/json'},
                               data=json.dumps({'destinationId': folder_id}))
```

Listing A.23. GraphService.py

The constructor requires a setting file which will be parsed directly from the configuration file discussed before. Then the app will set all the parameters needed, and it performs the authentication directly on web-browser. The strategy used is the *DeviceCodeCredential* used by microsoft to bind Azure user and devices, other possibility are available on Microsoft Doc [23]. Then the authentication is performed by means of `GraphClient` method which provides an AuthZ code to submit in the browser for the authentication. Finally, the scopes described which are the privilege granted to the endpoint. They must be a subset of the one reported in the configuration phase in the Appendix B. There are many utilities to interfacing also with email services. In that scenario EDR and detection system many times permits to sent email notification to a SOC mailbox in order to notify the alert detected. In this way is possible to implement a system able to sent mail notification to a specific inbox, the core process will fetch periodically the mailbox looking for new mail alerts. At that point if the mail is considered related to an alert it will be processed and then automatically moved into a child folder of the inbox so that they will not handle again. This process implements a basic queue where alert are sent to a mailbox, then are executed actions, then message is moved on Processed or Error subfolder based on results. This is a logic schema:

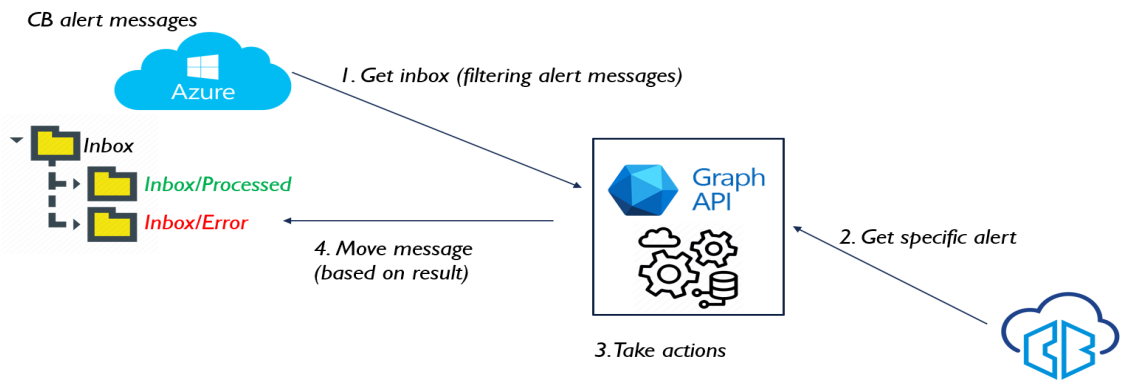

Figure A.4. Mail IR Flow

The operations described are reported in the routines presented in the class:

- get_target_mailbox: defined the inbox subfolder in order to store the messages. If are not present the function creates the two subfolder, in any case it returns the two folder id required to move messages. As shows in the `return` statement the procedure return a binding between alert_id and message_id.
- get_new_alert_from_mail: get all messages in inbox (thai is the pending queue) filtering mails which satisfy the standard subject for Carbon Black Alerts.
- move_processed: move a specific message (identified by message_id) into the subfolder identified by folder_id.

For sake of completeness is also possible to require the access token thanks to the method `get_access_token`. By the way the client permits to include token automatically by means of request built method of GraphClient. At the end of the operations is possible to sent emails with reports and failed tasks as described in `send_mail_review` method which creates the email attaching and Excel workbook. Workbook creation will be explained later in `utils.py` and basically it reports general overview of the incident handled in that iterations, then for each incident will be report which tasks have been performed against the ones failed.

In the file `responseAutomations.py` routines, that act as wrapper, are reported how includes this support to access Azure Services:

```
def set_up_graph():
    conf = configparser.ConfigParser()
    conf.read(['./constants.cfg'])
    azure_cfg = conf['azure']
    graph_service = GraphService(azure_cfg)
    return graph_service


def display_access_token(graph: GraphService):
    token = graph.get_access_token()
    print(f"Access Token is: {token}\n")
    return token
```

Listing A.24. responseAutomations.py

At that time the system is correctly configured to interact with Azure Services based on the scopes and permissions granted. Now is possible to extend the Graph Services to perform action on the user such as the one reported in the `automationsResponse.py`:

```
# User.Write
def block_user(user_email, token):
    url = f'https://graph.microsoft.com/v1.0/users/{user_email}'
    response = requests.patch(url,
                              headers={
                                  'Accept': 'application/json',
                                  'Content': 'application/json',
                                  'Authorization': f'Bearer {token}'
                              },
                              json={'accountEnabled': False})
    print("\tBLOCK USER: " + str(response.status_code))
    if response.status_code == 204:
        print(f'User: {user_email} is disabled\n')
        return True
    else:
        print(f"Error in disabling user - resp:{response.status_code}\n")
```

```
    return False


# User.Write
def reset_password(user_email, token):
    tmp_password = ''
    for _ in range(18):
        tmp_password += ''.join(secrets.choice(string.ascii_letters +
            string.punctuation + string.digits))
    url = f'https://graph.microsoft.com/v1.0/users/{user_email}'
    response = requests.patch(url,
                              headers={
                                  'Accept': 'application/json',
                                  'Content': 'application/json',
                                  'Authorization': f'Bearer {token}'
                              },
                              json={
                                  'passwordProfile': {
                                      'forceChangePasswordNextSignIn': True,
                                      'password': tmp_password
                                  }
                              })
    print("\tRESET PASSWORD: " + str(response.status_code))
    if response.status_code == 204:
        print(f'Triggered reset-password for user: {user_email} (TMP PASSWORD
            IS: \'{tmp_password}\')\n')
    else:
        print(f"Error in reset-password triggering -
            resp:{response.status_code}\n")
    return response.status_code, response.json()

# User.Write
def clean_up_sessions(user_email, token):
    url = f'https://graph.microsoft.com/v1.0/users/{user_email}/presence'
    response = requests.get(url,
                            headers={
                                'Accept': 'application/json',
                                'Content': 'application/json',
                                'Authorization': f'Bearer {token}'
                            })
    print("\tGET PRESENCE: " + str(response.status_code))
    if response.status_code == 200:
        print(f'Fetching user\'s presence completed for {user_email}\n')
        sessions_id = [item['id'] for item in
            json.loads(response.json()['value'])]
    else:
        print(f"Error in fetching user\'s presence -
            resp:{response.status_code}\n")
        return False
    tmp = 0
    for session in sessions_id:
        url =
            f'https://graph.microsoft.com/v1.0/users/{user_email}/presence/clearPresence'
        response = requests.post(url,
                                 headers={
                                     'Accept': 'application/json',
                                     'Content': 'application/json',
```

```
                'Authorization': f'Bearer {token}'
            },
            json={
                'sessionId': f'{session}'
            })
    print("\tCLEAR PRESENCE: " + str(response.status_code))
    if response.status_code == 200:
        print(f"Clean session {session} for user {user_email} completed\n")
        tmp += 1
    else:
        print(f"ERROR: Clean session {session} for user {user_email}
            failed - resp:{response.status_code}\n")
    return response.status_code, response.json(), len(sessions_id) - tmp
        # Return 0 on complete success otherwise number of failed session
        clean-up
```

Listing A.25. responseAutomations.py

The actions reported above concerns user block, reset password and session clean-up. The last two are quite important in case of phishing and email security where employee credentials are compromised. The main operative flows presented in all the routines starts invoking the API call thanks to the requests' module. The HTTPS request contains several headers such as Accept, Content-Type (for block user) and the Authorization which is composed by *Bearer* word and the token required thanks to the explained processes.

Then the response is evaluated starting from its status-code which can highlight that issues occurs, the overall return is based on the response status and on response value that will be used to understand if the automated operation is successful or not in the report phase. Pay attention that there is no API call to revoke session by bulk, the process is that require all the session for a user, collect their session-id, then clean up single session one by one.

A.4.3 VMWare CarbonBlack

Now that Azure is correctly configured is possible to adopt EDR functionalities to act directly on the involved end point. To do that Carbon Black provides a huge set of Integrable APIs to perform all the action discussed previously. At first there is the creation of API key which is done by means of processes described in the Appendix B. Likely the process presented for Azure, the configuration parameters are provided always in `constants.cfg`. Here the relevant information to exploit CarbonBlack functionalities:

```
[carbon-black]
ORG_KEY = ORG-KEY
API_ID = API-ID
API_SECRET = API-SECRET
```

Listing A.26. constants.cfg

Then to read those information from the configuration file there can be exploited a similar routine to the one presented in Azure, then can be implemented all the opportunities offered by CarbonBlack as stated in the `responseAutomations.py`:

```
def set_up_carbon_black():
    conf = configparser.ConfigParser()
    conf.read(['./constants.cfg'])
    cb_cfg = conf['carbon-black']
    return cb_cfg


def block_device(device_id, conf):
    url = f"https://defense-eu.conferdeploy.net/appservices/v6/orgs/
```

```
        {conf['ORG_KEY']}/device_actions"
    response = requests.patch(url,
                              headers={
                                  'Accept': 'application/json',
                                  'Content': 'application/json',
                                  'x-auth-token':
                                      f"{conf['API_ID']}/{conf['API_KEY']}"
                              },
                              json={"action_type": "QUARANTINE", "device_id":
                                  f"{device_id}"})
    print("\tBLOCK DEVICE: " + str(response.status_code))
    if response.status_code == 204:
        print(f'Device: {device_id} is blocked\n')
    else:
        print(f"Error in disabling user - resp:{response.status_code}\n")
    return response.status_code, response.json()


def extract_field_incident(item):
    tmp = [item['id'], item['create_time'], item['reason'], item['severity'],
        item['device_name'],
           item['device_username']]
    return tmp


def get_alert(conf, alert_id):
    url = f"https://defense-eu.conferdeploy.net/appservices/v6/orgs/
        {conf['ORG_KEY']}/alerts/{alert_id}"
    response = requests.get(url,
                            headers={
                                'Accept': 'application/json',
                                'Content': 'application/json',
                                'x-auth-token':
                                    f"{conf['API_KEY']}/{conf['API_ID']}"
                            })
    print("\tGET ALERT: " + str(response.status_code))
    if response.status_code == 200:
        print(f'Alerts fetched correctly\n')
        return extract_field_incident(response.json())
    else:
        print(f"Error fetching alert {alert_id} -
            resp:{response.status_code}\n")
    return False


def run_av_scan(device_id, conf):
    url =
        f"https://defense-eu.conferdeploy.net/appservices/v6/orgs/{conf['ORG_KEY']}/device_a
    response = requests.post(url,
                             headers={
                                 'Accept': 'application/json',
                                 'Content': 'application/json',
                                 'x-auth-token':
                                     f"{conf['API_ID']}/{conf['API_KEY']}"
                             },
                             json={
                                 "action_type": "BACKGROUND_SCAN",
```

```
                                "device_id": {device_id},
                                "options": {
                                    "toggle": "ON"
                                }
                            })
    print("\tRUN AV SCAN: " + str(response.status_code))
    if response.status_code == 204:
        print(f'AV Scan triggered on device {device_id} correctly\n')
    else:
        print(f"Error triggering AV-SCAN - resp:{response.status_code}\n")
    return response.status_code, response.json()
```

Listing A.27. responseAutomations.py

As the code snippet shows the CarbonBlack APIs supports several operations, but they require the three parameters explained before to work correctly. The org-key in inserted in the URL to identify the cloud console, then the API-ID and API-KEY are used in the HTPPS headers `X-AUTH-TOKEN`, similar to the authorization one provided for Azure systems.

The routine implements quarantine isolation for a device, alert list which is supported by the parser routine extracted_field_incident, and finally run AV scan on the machine. From the moment that alerts contains all the info concerning user and device involved, this will be the entrypoint to process incidents.

A.4.4 VMWare VSphere

VSphere is a virtualized environment used to handle dynamically services and server generated in farm. It a product widely used nowadays from the moment that it permits to generate, destroy, back-up and restore operations of virtual hosts. in addition, it is used also to up redundant machine acting as load balancer.

Refers to Appendix B to generate the API key and to insert them into `constants.cfg` file as reported previously. Then the operative code is similar to the one presented for the EDR counterpart, and it is reported in the file `responseAutomations.py`.

```
"""VSphere"""

def set_up_vsphere():
    conf = configparser.ConfigParser()
    conf.read(['./constants.cfg'])
    vsphere_cfg = conf['vsphere']
    return vsphere_cfg


def halt_machine(device_id, conf):
    url = f"https://{device_id}/api/appliance/shutdown?action=poweroff"
    response = requests.post(url,
                             headers={
                                 'Accept': 'application/json',
                                 'Content': 'application/json',
                                 'vmware-api-session-id': f"{conf['API_KEY']}"
                             },
                             json={
                                 "delay": 0,
                                 "reason": "Incident Automated Response"
                             })
    print("\tHALT MACHINE: " + str(response.status_code))
    if response.status_code == 200:
        print(f'Device {device_id} halted correctly\n')
```

```
    else:
        print(f"Error in halting system - resp:{response.status_code}\n")
    return response.status_code, response.json()


def restore_machine(device_id, conf):
    url = f"https://{device_id}/api/appliance/recovery/restore/job"
    response = requests.post(url,
                             headers={
                                 'Accept': 'application/json',
                                 'Content': 'application/json',
                                 'vmware-api-session-id': f"{conf['API_KEY']}"
                             },
                             json={
                                 "location": f"{conf['location']}",
                                 "location_type": "FTP"
                             })
    print("\tRESTORE MACHINE: " + str(response.status_code))
    if response.status_code == 201:
        print(f'Device {device_id} perform restore correctly\n')

    else:
        print(f"Error in restore job - resp:{response.status_code}\n")
    return response.status_code, response.json()



def get_back_up_parts(device_id, conf):
    url = f"https://{device_id}/api/appliance/recovery/backup/parts"
    response = requests.get(url,
                            headers={
                                'Accept': 'application/json',
                                'Content': 'application/json',
                                'vmware-api-session-id': f"{conf['API_KEY']}"
                            })
    print("\tBACK UP PARTS: " + str(response.status_code))
    if response.status_code == 200:
        print(f'Fetch {device_id} backup-parts correctly\n')
    else:
        print(f"Error in fetch backup parts - resp:{response.status_code}\n")
    return response.status_code, response.json()



def create_backup(device_id, conf):
    url = f"https://{device_id}/api/appliance/recovery/backup/job"
    response = requests.post(url,
                             headers={
                                 'Accept': 'application/json',
                                 'Content': 'application/json',
                                 'vmware-api-session-id': f"{conf['API_KEY']}"
                             },
                             json={
                                 "location": f"{conf['location']}",
                                 "location_type": "FTP",
                                 "parts": [
                                     "string"
```

```
                        ]
                    })
    print("\tCREATE BACK UP: " + str(response.status_code))
    if response.status_code == 201:
        print(f'Device {device_id} perform backup correctly\n')
    else:
        print(f"Error in backup job - resp:{response.status_code}\n")
        return response.status_code, response.json()
```

Listing A.28. responseAutomations.py

In the snippet starting from the device-id is possible to force the machine shut-down if required, again the result of the operation is mapped on the response status code and body information. For sake of completeness also back-up and restore basics are reported.

A.4.5 Orchestration With Python3

Now the system has all the ingredients to perform Incident Response mitigations autonomously. Now there is the necessity to implement a system that periodically fetch inbox emails looking for new incidents. To do that will be used the built-in library `sched` offered by Python3.

Will be initially presented the main logic to run correctly the App, producing also the report; then will be implemented the playbook itself.

Orchestrator and Scheduler Core

The main logics presented in the orchestrator is the following:

```
import responseAutomations
import utils
import sched
import time
from sys import exit

# .... #


def check_alerts_from_mail(sc, graph, conf_carbon_black, conf_vsphere):
    print(f"here {time.localtime()}\n")
    alerts = graph.get_new_alert_from_mail()
    global_data = []
    action_taken = []
    for i, alert in enumerate(alerts):
        try:
            incident_data = responseAutomations.get_alert(cb_conf,
                alert['alert_id'])
            global_data.append(incident_data)
            action = playbook_wrapper_1(incident_data)
            action_taken.append({'alert_id': alert['alert_id'], 'action':
                action})
            graph_services.move_processed(alert['message_id'], processed_id)
        except Exception as exc:
            graph_services.move_processed(alert['message_id'], error_id)
            print(f"[ERROR] --- {str(exc)} -> {alert['message_id']} is moved
                in /inbox/error\n")
            continue
    if len(global_data) != 0:
        res, filename = utils.create_workbook(global_data, action_taken)
```

```
        graph_services.send_mail_review(filename, res)
    sc.enter(60, 1, check_alerts_from_mail, (sc, graph, conf_carbon_black,
        conf_vsphere))


try:
    print("start...")
    graph_services = responseAutomations.set_up_graph()
    token = responseAutomations.display_access_token(graph_services)
    processed_id, error_id = graph_services.get_target_mailbox()
    cb_conf = responseAutomations.set_up_carbon_black()
    vsphere_conf = responseAutomations.set_up_vsphere()
except Exception as e:
    print(f"[ERROR] --- {str(e)}")
    exit(-1)

scheduler = sched.scheduler(time.time, time.sleep)
scheduler.enter(1, 1, check_alerts_from_mail, (scheduler, graph_services,
    cb_conf, vsphere_conf))
scheduler.run()
```

Listing A.29. main.py

In the snippet reported before is reported the following phases:

- Phase-1: in try/except corpus is set up the system reading all the configurations for Graph, Carbon-Black and VSphere as reported in the previous sections. Then mailbox are created if they do not exist.
- Phase-2: once systems are correctly configured, scheduler is created thanks to `sched` module. The scheduler is set up params requires the following input: delay, priority, routine, routine params
- Phase-3: the routine check_alerts_from_mail is invoked with `scheduler.run()`, now the program execute with one-second delay, the routine get all the alert-message bindings then process the alert thanks to the playbook. The routine has all the information about device and user gathered from alert description as described before. If no errors occur during processing, the emails is moved on processed otherwise it is moved on error subfolder. Finally, report are generated by means of `utils` support routine. Finally, the scheduler is reloaded in a similar way to a recursive function, with 60 seconds delay and the same parameters.

Reporting Support

Report are needed to give evidence of what action has been taken and possible problems. Understand which action are executed is of primarily importance in the Post Incident Activity phase.

As stated in the `check_alerts_from_mail` if alert are handled will be created an Excel report, in the next snippet is reported operative code, implemented exploiting `xlsxwriter` module.

```
import os
import xlsxwriter as xlsxwriter
import datetime

header_incident = [{'header': 'ID'}, {'header': 'DATE'}, {'header':
    'REASON'}, {'header': 'SEVERITY'},
                {'header': 'DEVICE'}, {'header': 'USERNAME'}]
header_action = [{'header': 'ACTION'}, {'header': 'TARGET'}, {'header':
    'API-RESP-STATUS'}, {'header': 'ERROR'}]
```

```
def create_workbook(data_incident, data_action):
    incident_tot = 0
    uncompleted_incident_response = []
    if not (os.path.exists('./report/')):
        os.mkdir('./report/')
    if not (os.path.exists("./report/excel")):
        os.mkdir('./report/excel')
    file_path = './report/excel/' +
        f"report_CB_{str(datetime.datetime.now().strftime('%m_%d_%Y&%H_%M'))}.xlsx"
    workbook = xlsxwriter.Workbook(file_path)
    style = workbook.add_format()
    style.set_text_wrap()
    style.set_align('vcenter')
    style.set_align('center')
    incident = workbook.add_worksheet('INCIDENTS-CB')
    red_format = workbook.add_format({'bg_color': '#FFC7CE', 'font_color':
        '#9C0006'})
    orange_format = workbook.add_format({'bg_color': '#F0FA66', 'font_color':
        '#FF8000'})
    green_format = workbook.add_format({'bg_color': '#48C427', 'font_color':
        '#000000'})
    incident.set_column(0, len(header_incident) - 1, 40)
    for i, row in enumerate(data_incident):
        incident.write_row(i + 1, 0, row, style)
        incident_tot += 1
    incident.add_table(0, 0, incident_tot, len(header_incident) - 1,
        {'columns': header_incident})
    incident.conditional_format(f'D2:D{incident_tot + 1}',
                            {'type': 'cell', 'criteria': '>=', 'value': 4,
                                'format': red_format})
    incident.conditional_format(f'D2:D{incident_tot + 1}',
                            {'type': 'cell', 'criteria': '=', 'value': 3,
                                'format': orange_format})
    for item in data_action:
        action_tot = 0
        completed_action = 0
        item_worksheet = workbook.add_worksheet(item['alert_id'].replace('-',
            '')[:31])
        item_worksheet.set_column(0, len(header_action) - 1, 25)
        for i, row in enumerate(item['action']):
            item_worksheet.write_row(i + 1, 0, row, style)
            action_tot += 1
            if row[3] == "---":
                completed_action += 1
        item_worksheet.add_table(0, 0, action_tot, len(header_action) - 1,
            {'columns': header_action})
        item_worksheet.conditional_format(f'D2:D{action_tot + 1}',
                                    {'type': 'cell', 'criteria': 'not equal
                                        to', 'value': '"---"',
                                     'format': red_format})
        success_ratio = completed_action / action_tot * 100
        if success_ratio != 100:
            uncompleted_incident_response.append(item['alert_id'])
        item_worksheet.write_row(action_tot + 2, 0, ['EFFECTIVENESS (%)',
            int(success_ratio)], style)
```

```
        item_worksheet.conditional_format(action_tot + 2, 1, action_tot + 2,
            1,
                                          {'type': 'cell', 'criteria': 'between',
                                           'minimum': 85,
                                           'maximum': 100,
                                           'format': green_format})
        item_worksheet.conditional_format(action_tot + 2, 1, action_tot + 2,
            1,
                                          {'type': 'cell', 'criteria': 'between',
                                           'minimum': 70,
                                           'maximum': 85,
                                           'format': orange_format})
        item_worksheet.conditional_format(action_tot + 2, 1, action_tot + 2,
            1,
                                          {'type': 'cell', 'criteria': 'between',
                                           'minimum': 0,
                                           'maximum': 70,
                                           'format': red_format})
    workbook.close()
    return uncompleted_incident_response, file_path
```

Listing A.30. utils.py

The code provide utils to create Excel workbook, for each iteration performed by the scheduler there is a global worksheet, then will be created a specific review of each incident handled based on percentage of completed task. Procedure return filepath referring the workbook created and `uncompleted_incident_response` array containing all the incident which incurs in failures. Input are respectively incident data and action take for ache incident data. The two typology of table created will present the following information:

- `INCIDENT-CB`: the table refers the `header_incident` schema. The table created on Excel is the one reported on table A.1.
- `incident-id`: report the incident_id and the schema fill the `header_action` schema. The table created on Excel is reported in A.2.

Id	Date	Reason	Severity	Device	Username
53bb7c2c-b5686db3098e	2022-11-02, 09:02	The application treesizefree.exe...	8	l***7	i**a@***d
35346b43-d3bbda78b097	2022-08-03, 14:37	The application dllhost.exe...	3	l***2	m****i@n****d
a2f958a9-262aff7d0bb5	2022-09-14, 13:06	The application ramdiskui.exe...	5	l***2	m***i@n***d
a7534cb4-69348d16956a	2022-10-22, 15:21	The application cmd.exe...	5	l***2	m****i@n***d

Table A.1. Automated Incident Response Global Stats

Action	Target	Response Status	Error
Block User	USER1	200	—
Isolate Device	DEVICE1	200	—
Halt Machine	DEVICE1	400	BAD REQUEST
Reset Password	USER1	401	UNAUTHORIZED

Table A.2. Automated Incident Response Detailed Overview

That Tabe will present also a line containing the following information `EFFECTIVENESS (%) 50`, highlighted in red from th moment that value match the last `item_worksheet.conditional_format` where bound are between 0 and 80. Concerning the `xlsxwriter` module the snippet presents

the creation of an Excel workbook in the folder `/report/<name>.xlsx` there are many formatting options which are used to highlights failed tasks and successful ratio. For further information about the utils used in the snippet presented, refers the official documentation available at https://xlsxwriter.readthedocs.io/.

Playbook Integration

Now is possible to implement the playbook according what is required by the system ot the designer. Playbook will work on incident_data and will provide all the information required by the reporting system that in turns are produced by the single task exposed in the `responseAutomations.py` code.

Basically a playbook can be derived from all of this operation implementing a specific conditional flows based on the information gathered in the alert provided by Carbon Black. Follows some examples included a support function to create the data format needed in the report system:

```
def evaluate_action_result(res_data, target, action_name):
    if 200 <= res_data[0] < 300:
        return [action_name, target, res_data[0], '---']
    else:
        return [action_name, target, res_data[0], res_data[1]]


"""BLOCK AND ISOLATE"""
def playbook_wrapper_1(incident_data):
    res = []
    block_user =
        responseAutomations.block_user(incident_data['device_username'],
        token)
    res.append(evaluate_action_result(block_user,
        incident_data['device_username'], 'Block User'))
    block_dev =
        responseAutomations.block_device(incident_data['device_name'],
        cb_conf)
    res.append(evaluate_action_result(block_dev,
        incident_data['device_name'], 'Block Device'))
    return res


"""BLOCK, ISOLATE, HALT IF SEVERITY GT 5"""
def playbook_wrapper_2(incident_data):
    res = []
    block_user =
        responseAutomations.block_user(incident_data['device_username'],
        token)
    res.append(evaluate_action_result(block_user,
        incident_data['device_username'], 'Block User'))
    block_dev =
        responseAutomations.block_device(incident_data['device_name'],
        cb_conf)
    res.append(evaluate_action_result(block_dev,
        incident_data['device_name'], 'Block Device'))
    if incident_data['severity'] > 5:
        halt_dev =
            responseAutomations.halt_machine(incident_data['device_name'],
            vsphere_conf)
        res.append(evaluate_action_result(halt_dev,
            incident_data['device_name'], 'Halt Device'))
```

```
return res
```

Listing A.31. main.py

As it is reported in the snippet, the two playbook performs basic operations required many times by the incident response. Unifying conditional flows provided by Python a programmer is able to implement all the scenarios that he wants, extending with few lines of code also complex scenarios. The considerations done here will be the baseline also for the VAPT's remediation.

A.5 Assessment Mitigation

Automations can be also involved to address evidence of risks highlighted by Red Teaming assessments, especially for Vulnerability assessment and penetration testing. It has been discussed support provided by automated framework in section 1.4.2 concerning Pentera framework. It provides a platform able to assess periodically a system considering new threats, and specific scenario (provided by Ransomware-ready or web-app module) and classical VAPT processes. Code that will be presented refers module `/pythonSrc/AssessmentMitigation/`

The system exploits machine learning and external resources to guarantee the best performance, giving the possibility to make an important assessment every week or month. Starting from those considerations will be exposed reports and data gathered from Pantera in order to automize mitigation processes.

A.5.1 Pentera's Assessment

During this experience, Pentera was configured to exploit computational power provided by other nodes to crack hashed passwords. In this phase with the customer was decided to involve external nodes with GPU based clusters in order to provide computationally resources to HashCat password Cracking tools. The attack simulation was designed to run at most 4 hours using a core node involved in the vulnerability scanning and exploit interactions, and a cluster composed of 4 RTX-3070 GPUs to support password cracking in parallel with HashCat. The normal password cracker is also supported with a predefined dictionary based on the well-known rock-you dictionary as well as other basic words related to the organization.

The metrics designed are the following:

- Password Cracked: this is one of the most important aspects. Many password cracks are synonyms of a weak password policy that an attacker may exploit, this aspect should be also related to the available computational resources, most important, when it can be adopted by an attacker.
- Number of Total Exploit: high exploit available can open up new attack chains and operations.
- Number of Administrative User Compromised: This is another important aspect because with high-level privileges attacks can become powerful.
- Vulnerabilities and related Score: A system with lots of high vulnerability does not survive to an attack. Furthermore, can be evidence of bad patch management.

The basic idea is to automize the password reset to create continuous monitoring of that possible users which became a significant risk for the infrastructure. Password cracking, many times, is the principal attack vector used by the attacker to get a foothold inside the internal network.

A.5.2 Weak Password Reset

After the Pentera Assessment, many specific reports are created and among them, there is the password reports generated within the operative domain. It consists of a list of users whose passwords can be considered weak according to the computational cracking resources employed by the assessment.

Furthermore, the reports explain if the password was cracked thanks to an exploit or thanks to classical cracking using hash-cat, reporting also the time consumed for complete cracking. Follows an example table:

User	Type	Obtained	Time Consumed	Domain/Host
u******1@*******.com	Domain User	Exploit, Cracking	Easy (00:05:37)	*******.com
r********o@*******.com	Domain User	Cracking	Easy (00:07:19)	*******.com
n**********e@*******.com	Domain User	Cracking	Strong (01:25:04)	*******.com
f*****r@*******.com	Domain User	Exploit, Cracking	Medium (00:35:21)	*******.com

Table A.3. Pentera - Password Cracking (HashCat) Stats

Starting from those data and exploiting the concept presented in the section A.4.2 is possible to force a password reset for such users.

```
import sys
from os.path import isfile, join
from os import getcwd
from GraphService import GraphService
import configparser
import pandas as pd

if len(sys.argv) != 2:
    sys.exit(-1)
if not isfile(getcwd().join(sys.argv[1])):
    sys.exit(-2)

try:
    print("start...")
    conf = configparser.ConfigParser()
    conf.read(['./constants.cfg'])
    azure_cfg = conf['azure']
    graph_services = GraphService(azure_cfg)
    users = pd.read_csv(getcwd().join(sys.argv[1]), usecols=[0])
    errors = 'ERROR_USER:\n'
    for user in users:
        if graph_services.reset_password(user):
            errors += f'{user}\n'
    with open('./report.txt', 'w') as report:
        report.write(errors)
    print("end...")
except Exception as e:
    print(f"[ERROR] --- {str(e)}")
    exit(-1)
```

Listing A.32. AssessmentMitigation/main.py

As described before, the script will analyze the produced report, passed as a CLI parameter, and then will iterate over all risky user to trigger a reset password. In case of errors, user will be reported in the report.txt.

While the reset-password routine is the following:

```
def reset_password(self, user_email):
    url = f'https://graph.microsoft.com/v1.0/users/{user_email}'
    resp = self.graph_client.patch(url,
                          headers={'Content-Type': 'application/json'},
                          data=json.dumps({'passwordProfile':
                              {'forceChangePasswordNextSignIn': True}}))
    if resp.status_code != 204:
        return False
    return True
```

Listing A.33. AssessmentMitigation/GraphService.py

Also in that case the system can easily be extended by means of support tools and operations, such as mail notification reporting or session clean-up as described in the incident response operations in section A.4.

Appendix B

User's Manual

B.1 Python and System Set Up

B.1.1 Install Python3

In this Appendix will be explained how to configure and set up the entire system to run correctly all the python script provided within this opera on Windows Systems. Brief details will be given to support the same operation also in the Unix-Like System.

Windows

The first step is to download the Python3 from the official website from this link https://www.python.org/downloads/ then download and follow the set-up wizard for a correct installation of the interpreter.

Linux and macOS

To run the code on Linux devices download the interpreter

```
sudo apt update
sudo apt upgrade
```

Python3 should be installed as baseline components, if not run the following command:

```
sudo apt install python3
```

On the MacOSX system python can be downloaded from the same link referenced for Windows or by the homebrew package manager (https://brew.sh/index_it). Follow the installation procedure reported on the official site, then run the following command:

```
sudo brew install python3
```

Invoke Python Interpreter

Once Python is installed locate its binary path, for Windows system the interpreter will use the *py.exe* syntax to be invoked. By the way, the syntax will be the following:

```
<path/to/python> <path/to/script> <...arguments>
examples:
py.exe ./NetworkDetectionSystem/SpamDetectionSystem.py
    ./datasource/emails/spamhassassin/ham
    ./datasource/emails/spamhassassin/spam
```

If everything goes well in the installations, using a Powershell or a simple terminal Python can be invoked and added to PATH Env variable as the user prefers. This operation will be used Powershell within a Windows System. *NOTE: if the user has already an available Python Interpreter downloaded by the Microsoft Store, the syntax to call the interpreter will be python.exe instead of py.exe, by the way, calls to the absolute path to the interpreter always work especially for Linux and MacOSX User. Pay attention that different python interpreters can manage different package locations, for this reason, be sure to run the dependencies satisfaction and the code examples with the same interpreter.*

Now that Python is configured is possible to proceed with the setup script. Then the working directory which is open our PowerShell will be the pythonSrc folder unzipped:

```
C:\path\to\folder\pythonSrc\
```

B.1.2 Dependencies Satisfaction

To run the code offered in this artefact the system requires some well-known dependencies to perform correctly. Follows the list of external modules required by the system:

- wheel: used as dependencies for other modules.
- requests: used to invoke easily APIs.
- psutil: used to get information about systems (processes, CPU load etc. . .)
- hashlib: hashing libraries.
- nltk: used to exploit NGrams algo.
- pefile: used to feature Portable Executable files in malware detection.
- sklearn: used to implement machine learning model.
- xgboost: Gradient Boosting Classifier.
- numpy: used to work with array (required by sklearn).
- pandas: used to read CSV and dataset.
- matplotlib: used to plot graphs and stats.

All those dependencies will be downloaded by the *setup.py* script in the rootcode folder. Follows the command:

```
py.exe setup.py -i (install)
py.exe setup.py -u (uninstall)
```

pay attention: Uninstalling the modules with '-u' option will remove permanently all the previously stated modules. All the command requires the Powershell admin privileges (sudo grant in Unix) in order to install correctly the dependencies:

B.1.3 Datasource and CLI parameters

All the code is provided with essential data to work correctly operations. starting from the rootcode folder is possible to identify the *datasource* folder. Inside the folder, there is dataset and samples to pass to Command Line Interface to run the script. Follows the data source tree:

```
C:\PATH\TO\MY\FOLDER\pythonSrc\datasource\
|---emails
| |---spamhassassin
| | |---ham
| | |___spam
| |___url_phishing
|---exe_samples
| |---benign
| |___malware
|___NBAD
```

The following dataset and folder will be used in the following sections to run correctly the code. **The Working Directory assumed will be C:\PATH\TO\MY\FOLDER\pythonSrc**, consider that if you run the system outside. **Pay Attention that exe_samples contains 300 well-known malware and download/access can trigger the antivirus action stopping the execution, is suggested to create an exclusion to run the scripts properly.**

B.2 Run The Code

B.2.1 Detection

In the detection, code will be presented the main aspects concerning machine learning models. The tree is formed as follows:

```
C:\PATH\TO\MY\UFOLDER\pythonSrc\detection
|---HashBasedMalwareDetection
|---MLBasedMalwareDetection
|___NetworkDetectionSystem
```

HashBasedMalwareDetection Module

In this module is offered a simple script to monitor and remove local processes based on their reputation provided by Virus-Total API lookup. The command is the following:

```
#Hash Detection System:
py.exe ./detection/HashBasedMalwareDetection/HashDetectionSystem.py
```

MLBasedMalwareDetection Module

In this subsection will be executed many modules starting from Ngrams Concept to Feature extraction, Then malware detection system based on Random Forest will be analyzed and finally, there is a global comparison between Random-Forest and XGBoost (this can take many hours to be completed, set NGrams max properly). The commands are the following:

```
# NGrams concept:
py.exe ./detection/MLBasedMalwareDetection/NGrams.py
    ./datasource/exe_samples/benign/00eea85752664955047caad7d6280b
    c7bf1ab91c61eb9a2542c26b747a12e963.exe 5

# NGrams featurization:
py.exe ./detection/MLBasedMalwareDetection/NGramsSelection.py
    ./datasource/exe_samples/benign ./datasource/exe_samples/malware 2
    1000 10

# Ranndom Forest Malware Detection Classifier:
```

```
py.exe ./detection/MLBasedMalwareDetection/MalwareDetectionSystem.py
    ./datasource/exe_samples/benign ./datasource/exe_samples/malware 2 10

# Comparator (Random Forest and XGBoost, most common pattern are [10, 20,
    50, 100, 200]):
py.exe ./detection/MLBasedMalwareDetection/MalwareDetectionComparator.py
    ./datasource/exe_samples/benign ./datasource/exe_samples/malware 5 10
```

Few samples can present errors due to platform incompatibility with binary, especially in malware detection system pefile module will raise the error without stopping the execution

Follows a report of what is viewed on PowerShell:

```
py.exe ./detection/MLBasedMalwareDetection/MalwareDetectionSystem.py
    ./datasource/exe_samples/benign ./datasource/exe_samples/malware 2 10

Start training
Error: 'Invalid e_lfanew value, probably not a PE file
Error: 'PE' object has no attribute 'DIRECTORY_ENTRY_IMPORT'
Error: 'PE' object has no attribute 'DIRECTORY_ENTRY_IMPORT'
Error: 'PE' object has no attribute 'DIRECTORY_ENTRY_IMPORT'
Error: 'Invalid NT Headers signature. Probably a NE file'
Error: 'PE' object has no attribute 'DIRECTORY_ENTRY_IMPORT'
Error: 'PE' object has no attribute 'DIRECTORY_ENTRY_IMPORT'
Error: 'PE' object has no attribute 'DIRECTORY_ENTRY_IMPORT'
Error: 'DOS Header magic not found.'
Error: 'DOS Header magic not found.'
Error: 'Invalid e_lfanew value, probably not a PE file
End training

Start testing
Error: 'DOS Header magic not found.'
Elapsed time: 150.93721079826355
End testing

Accuracy in testing: 98.88268156424581%

CONFUSION MATRIX
[[90 0]
 [ 2 87]]
```

Listing B.1. MalwareDetectionSystem.py - Output result

NBAD and Email Detection

In this folder are presented many modules starting from NBAD models thanks to Isolation Forest to Phishing URLs and Spam Detection. The commands are the following:

```
# NBAD (set cutoff to reasonable value, 0.00 for the example):
py.exe ./detection/NetworkDetectionSystem/NetworkBehavioralDetection.py
    ./datasource/NBAD/kddcup_dataset.csv 0 0.01

# Phishing URL Detection:
py.exe ./detection/NetworkDetectionSystem/PhishingUrlDetectionSystem.py
    ./datasource/emails/url_phishing/url_phishing_train.csv
    ./datasource/emails/url_phishing/url_phishing_test.csv

#Spam Detection:
```

```
py.exe ./detection/NetworkDetectionSystem/SpamDetectionSystem.py
    ./datasource/emails/spamhassassin/ham
    ./datasource/emails/spamhassassin/spam
```

Few samples can present errors due to platform incompatibility with binary, especially in malware detection system pefile module will raise the error without stopping the execution

Follows a report of what is viewed on PowerShell:

```
py.exe ./detection/NetworkDetectionSystem/SpamDetectionSystem.py
    ./datasource/emails/spamhassassin/ham
    ./datasource/emails/spamhassassin/spam
Error: 'charmap' codec can't decode byte 0x8d in position 3062: character
    maps to <undefined>
Error: 'charmap' codec can't decode byte 0x90 in position 2832: character
    maps to <undefined>
Error: 'charmap' codec can't decode byte 0x9d in position 4099: character
    maps to <undefined>

STATS:
Accuracy: 0.9892729439809297

Confusion-Matrix:
[[269 4]
 [ 5 561]]
```

Listing B.2. SpamDetectionSystem.py - Output result

B.2.2 Hardening and Prioritization

The code presented in this section refers to the module `/pythonScr/hardening/MFAHardeningAndPrioritization`. To run the script Azure services are needed, pay attention that headers MUST BE in English format this is possible by modifying browser language settings or acting directly on Azure console [24].

In that sense is possible to access the signin logs from Azure's portal: https://login.microsoftonline.com/, then from the section users sign in is possible to export the needed CSV, rename the file `sign_in.csv` and move it into the root folder `/pythonScr/hardening/MFAHardeningAndPrioritization`.

Get MFA status and run program

It is also required the MFA status of users in the organizations that can be easily exported through `./utils/getmfa.ps1` script (admin privileges are needed). Run the following command in Powershell to export the file in CSV format:

```
./getmfa.ps1 | Export-Csv ./mfa.csv
```

From the exported file remove the first line that is not needed.

Once the two files are ready to script can be invoked using the following command:

```
py.exe ./getMfaStatus.py ./sign_in.csv
```

The script will produce the results in `/pythonScr/hardening/MFAHardeningAndPrioritization/report` folder.

B.2.3 Response

To run the code concerning the response automation is required to have an active Azure Tenant and a CarbonBlack Cloud licence on Vsphere's nodes. Starting from the Azure point of view is needed to set up the binding with an application in order to access Azure Services through Graph APIs as reported by Microsoft best practice [25], pay attention that this procedure requires again an administrative user.

Once App Registration is completed, import the required information in the configuration file provided in the code section for the file `constants.cfg` within `[azure]` section. Are required:

- APP ID = Generated during the App Registration process
- DIRECTORY ID = Directory (Tenant) ID is available from the same page of app registration otherwise from the Azure properties are accessible always by Azure Portal. Scroll down to the *Tenant ID field*.
- SECRET ID = Secret id is generated during app registration

The permission needed is the one already explained in the previous section, and they are:

- **Mail.ReadWrite**: to move and read emails.
- **User.ReadWrite.All**: to block and reset password to a user (required delegated permission for admin).
- **Mail.Send**: to send out emails.

Now access the CarbonBlack Cloud console in order to require an API key following the official documentation giving the right access level and to the API User[26]. In that case, are required following information:

- API KEY: generated during key generation.
- API ID: generated during key generation.
- ORG KEY: organization key, Carbon Black is a multi-tenant console, choose the correct one.

Similarly is possible to obtain also the API key for the Vsphere following the official documentation[27]. In that case, is also required the API key itself.

Once the constants file is created run the following command in the folder `/pythonSrc/response/IncidentResponseFlow`.

```
py.exe ./main.py
```

After a few seconds will be displayed a string containing an authentication code to be inserted into the online form. The authentication required the admin user to guarantee a grant to perform the operation on the Azure tenant. Once Login successfully follows the guided procedure by Microsoft consenting to all the requirements then the program will initiate alerts' fetching and response if needed.

Note that configurations and commands can be used also to work on the AssessmentMitigation module present at the path `/pythonSrc/response/AssessmentMitigation/`.

Bibliography

[1] Mark Campbell, "What are the consequences of data loss?", 2021, https://www.unitrends.com/blog/what-are-the-consequences-of-data-loss

[2] Wikipedia, "ISO/IEC27001", 2022, https://it.wikipedia.org/wiki/ISO/IEC_27001

[3] Wikipedia, "PDCA Model," https://it.wikipedia.org/wiki/Ciclo_di_Deming

[4] International Organization for Standardization, "ISO/IEC27001 - Annex A Controls", 2019, https://www.isms.online/iso-27001/annex-a-controls/

[5] GDPR, "General Data Protection Regulation", 2016, https://gdpr-info.eu/

[6] Payment Card Industry Data Security Standard, "Requisiti e procedure di valutazione della sicurezza", 2021, https://it.pcisecuritystandards.org/_onelink_/pcisecurity/en2it/minisite/en/docs/PCI_DSS_v3-2-1_IT.pdf

[7] NIST, "NIST CSRC Glossary", 2022, https://csrc.nist.gov/glossary

[8] NIST, "NIST Risk Management Framework", 2020, https://csrc.nist.gov/projects/risk-management/about-rmf

[9] MITRE, "MITRE CVE", 2022, https://cve.mitre.org/

[10] MITRE, "MITRE CWE", 2022, https://cwe.mitre.org/

[11] Pentera, "Pentera", 2022, https://pentera.io/

[12] K. Dempsey, P. Eavy, and G. Moore, "Automation Support for Security Control Assessments", 2017, https://nvlpubs.nist.gov/nistpubs/ir/2017/NIST.IR.8011-1.pdf

[13] NIST, "Computer Security Incident Handling Guide", 2018, https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-37r2.pdf

[14] NIST, "Security and Privacy Controls for Information Systems and Organizations", 2020, https://doi.org/10.6028/NIST.SP.800-53r5

[15] Kaspersky, "Kaspersky ML Resources", 2021, https://www.kaspersky.com/enterprise-security/wiki-section/products/machine-learning-in-cybersecurity

[16] E. Tsukerman, "Machine Learning for Cybersecurity Cookbook", 2019, https://www.packtpub.com/product/machine-learning-for-cybersecurity-cookbook/9781789614671

[17] NIST, "Framework for Improving Critical Infrastructure Cybersecurity", 2018, https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.04162018.pdf

[18] P. Cichonski, T. Millar, T. Grance, and K. Scarfone, "Computer Security Incident Handling Guide", 2018, https://nvlpubs.nist.gov/nistpubs/specialpublications/nist.sp.800-61r2.pdf

[19] ENISA, "ENISA's CERT List", 2022, https://www.enisa.europa.eu/topics/csirts-in-europe/csirt-inventory/certs-by-country-interactive-map

[20] K. Dempsey, N. S. Chawla, A. Johnson, R. Johnston, A. C. Jones, A. Orebaugh, M. Scholl, and K. Stine, "Information Security Continuous Monitoring (ISCM) for Federal Information Systems and Organizations", 2011, https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-137.pdf

[21] M. Souppaya and K. Scarfone, "Guide to Enterprise Patch Management Planning: Preventive Maintenance for Technology", 2022, https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-40r4.pdf

[22] MITRE, "MITRE ATTACK", 2022, https://attack.mitre.org/

[23] Microsoft, "Microsoft Graph API", 2022, https://learn.microsoft.com/en-us/graph/sdks/choose-authentication-providers?tabs=CS

[24] Microsoft, "Microsoft Azure Settings", 2022, https://learn.microsoft.com/en-us/azure/azure-portal/set-preferences

[25] Microsoft, "Microsoft Azure Register App", 2022, https://learn.microsoft.com/en-us/azure/healthcare-apis/register-application

[26] VMWare, "Carbon Black API Key Generation", 2022, https://developer.carbonblack.com/reference/carbon-black-cloud/authentication/

[27] VMWare, "VSphere API Key Generation", 2022, https://docs.vmware.com/en/VMware-vRealize-Log-Insight-Cloud/services/User-Guide/GUID-A46EAF77-22DE-48B4-93ED-FD02A407D41C.html