



**Politecnico
di Torino**

POLITECNICO DI TORINO

Master's of Engineering in Cybersecurity

MASTER'S DEGREE THESIS

Anomaly detection for passive monitoring of internet traffic

Supervisor:

Prof. Marco Mellia
Prof. Danilo Giordano

Author:

Edoardo Oldani

Supporter:

Prof. Martino Trevisan

Academic Year 2025/2026

Abstract

This thesis focuses on anomaly detection for passive monitoring of satellite Internet traffic. It starts by outlining the satellite communication context and the characteristics of EUTELSAT customer traffic, supported by a review of related and competitor studies. The work then builds a scalable processing pipeline to transform raw monitoring logs into structured data, enabling consistent analysis on large volumes of traffic.

Using the aggregated dataset, the thesis develops a multi-step analysis of user behavior: initial exploration of traffic patterns and outliers, profiling of service usage, and a shift from purely volume-based metrics to behavioral indicators. These macro-level steps provide a clearer view of how users differ and where anomalous behaviors may emerge.

Building on this foundation, the thesis applies unsupervised techniques to prioritize suspicious users by combining clustering evidence and autoencoder reconstruction signals into an internal anomaly score. The ranked candidates are then organized through anomaly-driven behavioral classes and further examined through a raw-traffic validation workflow, with specific focus on residential-proxy-plausible profiles. Rather than claiming direct forensic attribution, the proposed framework provides a reproducible and operationally actionable methodology that links large-scale passive measurements to interpretable, high-priority investigation candidates.

Contents

1	Introduction	4
1.1	Problem statement and motivation	4
1.2	Contributions	5
1.3	Thesis organization	5
2	Background and related work	7
2.1	Satellite communication systems	7
2.2	Cybersecurity in satellite communications and residential proxies	8
2.3	Traffic metrics	9
2.4	Related work	10
2.4.1	Satellite-network measurements	10
2.4.2	Erroneous-traffic signals for passive security monitoring	10
2.4.3	Anomaly detection using passive traffic measurements	10
2.4.4	Residential-proxy and intermediary-ecosystem studies	12
3	Aggregation pipeline and exploratory analysis	13
3.1	Motivation and objectives	13
3.1.1	Why aggregation is required	13
3.2	Data sources and script inputs	13
3.2.1	Tstat logs and completeness	13
3.2.2	Inputs and configuration	14
3.3	Aggregation script architecture	15
3.3.1	Pipeline overview	15
3.4	Spark execution model and scalability considerations	16
3.4.1	Execution setup and parameterization	16
3.4.2	How Spark is used in the pipeline	17
3.4.3	Output materialization strategy	17
4	Exploratory analysis and feature engineering	19
4.1	Data validation and visualization	19
4.1.1	Why traffic is split by IP entity type	19
4.1.2	Generic user analysis	20
4.1.3	Service-level analysis	20
4.1.4	Traffic distribution and user variability	22
4.2	Feature engineering	23
4.2.1	Volume feature set	23
4.2.2	Behavioural feature set	24
4.2.3	Normalization and user signature representation	25

5	Anomaly detection and evaluation	26
5.1	Methodological roadmap	26
5.2	Clustering-based anomaly exploration: methodology and results	27
5.2.1	DBSCAN	27
5.2.2	PCA-supported DBSCAN	31
5.2.3	Clustering results	33
5.3	Deep-learning anomaly modeling	33
5.3.1	Deep-learning theoretical background	34
5.3.2	Autoencoder setup	36
5.3.3	Autoencoder results: reconstruction-error evidence	37
5.3.4	Latent-space clustering	39
5.4	Internal anomaly score integration	42
5.4.1	Evaluation perspective	44
6	Attack pattern recognition	45
6.1	Anomaly-Driven Behavioral Classification: methodology and results	45
6.2	Residential Proxy Identification: methodology	47
6.2.1	Candidate generation	49
6.3	Engineering Analysis on Raw Traffic Evidence: results and interpretation	50
6.3.1	Data scope and reconstruction protocol	51
6.3.2	Residual candidate profiling	52
7	Conclusion and future work	55
7.1	Future works	56

Chapter 1

Introduction

The growth of satellite broadband is bringing Internet connectivity to areas where terrestrial networks are scarce or unreliable, and it is increasingly used as a complementary access technology even in well-served regions. This shift creates new operational challenges: satellite links face higher latency, variable capacity, and more pronounced congestion effects than fiber or cable networks, and they must support traffic mixes that range from interactive web browsing to large software updates and streaming. The result is a highly dynamic environment where service quality can fluctuate due to weather, capacity contention, or infrastructure issues that are not always visible through active probes alone.

Maintaining acceptable service quality therefore depends heavily on passive observation of traffic. Passive monitoring offers a granular and comparatively unfiltered account of what subscribers actually generate and receive, rather than what test traffic predicts. By reflecting genuine application behavior, protocol dynamics, and the consequences of traffic management choices, these measurements support a broad, population-level view of performance. Yet the same strength creates a practical obstacle: data arrive in large volumes and with substantial variability, which complicates timely recognition of anomalies, degradations, and early-stage incidents. Collection alone is insufficient, the central difficulty lies in extracting actionable meaning from what is recorded.

This thesis addresses anomaly detection in the setting of passive monitoring for satellite Internet traffic. It starts from operational logs and prior measurement studies assembled before the project formally began, then proceeds to the construction of a scalable processing pipeline suitable for high data rates. The intended outcome is twofold: an analytics-ready dataset that remains dependable under operational constraints, and an assessment of anomaly detection approaches that can assist operators in identifying atypical traffic behavior. Coverage extends across the data lifecycle, from interpreting raw logging artifacts, through aggregation choices aimed at retaining signals of operational interest, to evaluating detection methods with an emphasis on applicability in real deployments.

1.1 Problem statement and motivation

Satellite networks support users spread across wide geographic areas, and their needs vary sharply, ranging from everyday household browsing to applications that are operationally critical for businesses. As a result, traffic demand is uneven in time and place: daily cycles, location-specific surges, and idiosyncratic user habits all accumulate on top of a shared platform. In this setting, even minor faults or configuration errors can surface as user-visible degradations, although the evidence is often faint and dispersed across millions of individual flows. Passive measurement yields detailed observations, but the

logs themselves do not readily translate into routine operational analysis.

The thesis therefore addresses two connected tasks. One concerns representation: how raw traffic records can be converted into a compact and organized form that still retains what is necessary to describe customer behavior. Monitoring output is generated at very large scale, with fine-resolution entries and flow counts in the millions, which makes direct inspection infeasible. Some level of aggregation is unavoidable if the dataset is to become workable, yet the aggregation must preserve indicators that track user activity and perceived service quality. The target representation must also separate meaningful protocol and application differences, while remaining small enough to sustain analytics at scale. The other task concerns detection: how that representation can be used to identify anomalies that are operationally and customer-relevant. Not all anomalies are simple service degradations. Detection is also central for spotting malicious activity and misuse of satellite capacity, including patterns consistent with abuse, rule violations, or coordinated hostile behavior. When effective, anomaly detection shortens diagnosis time, supports service quality management, and brings attention to unanticipated events such as outages, routing mistakes, or atypical demand spikes. Over longer horizons, it can also expose persistent shifts in usage or recurring changes in infrastructure behavior.

From an operational standpoint, anomaly detection in satellite traffic poses several additional constraints. Data is collected at high volume and must be processed efficiently. Signals of interest can be rare and short-lived, yet they may have disproportionate impact on user experience. Moreover, labeling anomalies is difficult, so practical methods should work in a weakly supervised or unsupervised setting. These constraints motivate a workflow that prioritizes scalable data engineering, carefully designed features, and robust detection strategies that do not depend on extensive ground truth.

1.2 Contributions

This thesis delivers a cohesive workflow that moves from preliminary studies to a deployable data pipeline and, finally, to anomaly detection experiments on the resulting dataset. The contributions combine engineering work (to make large-scale traffic data usable), analytical exploration (to understand behavior and design features), and modeling efforts (to detect abnormal patterns with operational relevance). In particular, the contributions can be summarized as:

- an exportable, scalable PySpark aggregation pipeline with configuration files and documentation to transform raw Tstat logs into hourly, per-customer summaries;
- exploratory data analysis and feature engineering in notebooks to characterize traffic patterns and build modeling-ready datasets;
- an anomaly detection workflow and experiments on the aggregated data to identify abnormal behavior and potential misuse.

1.3 Thesis organization

The remainder of the thesis is organized as follows. Chapter 2 ([Background and related work](#)) introduces the satellite-network context, the traffic-observation perspective, and the related literature on passive measurements and anomaly detection. Chapter 3 ([Aggregation pipeline and exploratory analysis](#)) presents the dataset, the execution environment, and the scalable aggregation pipeline that converts raw Tstat logs into analysis-ready

user-level summaries. Chapter 4 ([Exploratory analysis and feature engineering](#)) reports exploratory analysis and feature-engineering choices that define the behavioral representation used by downstream models. Chapter 5 ([Anomaly detection and evaluation](#)) details the anomaly-detection workflow, including clustering exploration, autoencoder-based reconstruction analysis, and multi-signal score integration. Chapter 6 ([Attack pattern recognition](#)) develops anomaly-driven behavioral classification and residential-proxy-oriented candidate refinement, with subsequent raw-data validation and engineering interpretation. Finally, Chapter 7 ([Conclusion and future work](#)) summarizes the main findings, discusses methodological limits, and outlines future research directions.

Chapter 2

Background and related work

This chapter provides the context needed to interpret satellite Internet traffic and to motivate the anomaly detection approach adopted in this thesis. We first summarize the characteristics of satellite communication systems and the implications for passive monitoring. We then introduce the cybersecurity aspect in passive traffic analysis, including the residential-proxy phenomenon as a role-oriented threat model. After that, we discuss the traffic metrics that are most informative for understanding customer behavior and operational health. Finally, we review related work on satellite performance measurement and on anomaly detection from passive data, with emphasis on erroneous traffic analysis.

2.1 Satellite communication systems

Satellite Internet access differs from terrestrial broadband mainly because of the physical path that traffic must traverse and the operational constraints of space-based infrastructure. The signal must be transmitted over long radio links, pass through satellite payloads, and be relayed to ground gateways before entering the Internet backbone. These elements introduce higher propagation delays and stronger variability than fiber or cable, and they expose the system to weather-related attenuation, interference, and limited spectrum. In addition, the radio access segment is shared among many users, so traffic dynamics and scheduling decisions at the satellite layer can strongly influence end-to-end performance.

Two main orbital regimes are relevant. Geostationary (GEO) satellites orbit at a fixed position relative to the Earth, enabling wide coverage with a small number of spacecraft. The trade-off is a large round-trip delay, which makes interactive applications more sensitive to congestion and loss. Low Earth Orbit (LEO) constellations reduce latency by operating closer to the Earth, but they require a large fleet of satellites and frequent handovers as satellites move across the sky. The Starlink system is a prominent example of this approach and has been shown to achieve low latency and competitive throughput under light load, while still exhibiting higher delay and loss under heavy usage [25]. These dynamics highlight why continuous monitoring is necessary to understand the actual user experience.

From a network perspective, satellite access introduces additional architectural components that are not present in traditional fixed broadband. User terminals connect to the satellite access network, which maps traffic into beams and frequency resources, then forwards it through gateway stations toward the public Internet. Routing decisions may change as satellites and gateways shift, and performance enhancing proxies are often deployed to mitigate the impact of long delays on TCP, shaping the way transport protocols behave end-to-end [25]. This architecture creates a layered system in which performance

issues can originate from multiple points: radio link conditions, gateway congestion, back-haul constraints, or proxy behavior.

At the transport layer, long or variable RTTs interact with congestion control, loss recovery, and handshake procedures. This is particularly relevant for TCP, where slow start and retransmission timeouts can be amplified by high delays, reducing perceived throughput. Satellite providers often deploy acceleration techniques or proxies to improve performance, which can make measurements protocol-dependent and can obscure the true behavior of end-to-end paths. The Starlink study explicitly uses QUIC to bypass such proxies and to provide a clearer view of performance under load [25]. This highlights the need to interpret measurements carefully and to combine passive and active perspectives when diagnosing performance issues.

Because the access portion is shared, multiple users contend for the same capacity, and schedulers can either favor particular services or reallocate resources as demand fluctuates. In practice, the outcome is pronounced day, night structure, marked differences across regions, and sudden discontinuities once congestion emerges. From an operational standpoint, this undermines any attempt to reduce quality of experience to a single indicator: throughput, delay, packet loss, and the service, usage composition must be read jointly if one aims to characterize the network’s condition.

Passive monitoring can provide that complete perspective, yet only when high-rate observations are reduced to compact summaries that still retain the salient signals. Those properties are precisely what make passive approaches especially useful. When real traffic is observed at aggregation points, passive measurements expose application dynamics, cross-protocol effects, and the signatures of congestion without adding extra probe traffic. They also register the coupling between transport behavior and satellite-specific performance tweaks, interactions that active testing on its own can overlook. At the same time, traffic volumes and fast-changing conditions leave raw traces hard to interpret unless they are aggregated.

Accordingly, satellite operators require monitoring systems able to condense large traffic streams while keeping the information most relevant to user experience and overall network health. Variation in delay, loss, and throughput across time, geography, and service classes supports the adoption of scalable aggregation pipelines alongside strong anomaly detection methods. With these in place, operators can separate expected seasonal or diurnal structure from true incidents, then act quickly when observed behavior departs from what is normally anticipated.

2.2 Cybersecurity in satellite communications and residential proxies

Cybersecurity for satellite communications operates under constraints that are not the same as those of terrestrial fixed networks. Rather than presenting a single perimeter, the exposed surface is distributed across several layers, including terminal equipment, the satellite access link, gateway systems, control and management functions, and the points where these networks connect to the public Internet, each element contributing to the overall risk profile. At the same time, the radio portion faces hazards that are less prominent in wired settings, notably interference, deliberate jamming, and spoofing efforts. Yet above this, the service layer remains open to familiar misuse patterns associated with conventional IP-based infrastructures.

Operational constraints amplify the security challenge. Satellite ecosystems often involve long-lifecycle equipment and geographically distributed deployments, which can in-

crease patching latency and reduce uniformity of security controls. Under these conditions, purely signature-based defenses are frequently insufficient, especially against previously unseen attack strategies or adaptive automated abuse.

For this reason, anomaly detection has become a relevant complementary paradigm in satellite-network security. Rather than relying exclusively on known signatures, anomaly-based approaches model baseline traffic regularities and identify statistically or behaviorally inconsistent patterns. This is particularly useful in passive-monitoring scenarios, where payload inspection may be unavailable and where security inference must be extracted from flow-level, temporal, and directional signals [7, 33]. In practice, such methods support early warning, prioritization of suspicious entities, and reduction of analyst search space in large-scale operational datasets.

Within this broader cybersecurity context, residential proxies represent a significant phenomenon because they weaken source-identity assumptions commonly used in defensive workflows. A proxy is an intermediary entity that forwards requests toward upstream services [11]; in residential-proxy infrastructures, forwarding is performed through IP addresses assigned to consumer access networks. As documented in recent studies, these ecosystems can support both legitimate and abuse-oriented use cases, and they complicate attribution by decoupling apparent source identity from effective traffic control [8, 23, 24]. Consequently, their detection in passive data requires behavioral evidence beyond simple origin-based filtering.

Additional measurements of proxy ecosystems reinforce this point. Large-scale analyses of open and free proxies report frequent content manipulation, unstable trust properties, and heterogeneous operational behaviors [22, 28, 38, 41]. Related evidence on HTTPS interception further shows that intermediary infrastructures can alter end-to-end security assumptions in ways that are difficult to identify through origin-based heuristics alone [9]. Taken together, these studies motivate proxy analysis as a behavioral-inference problem rather than a static address-classification task.

2.3 Traffic metrics

The raw dataset is composed of three log types: TCP complete, TCP not-complete, and UDP complete. TCP complete logs include conversations that start and finish correctly, whereas TCP not-complete logs capture connections that do not conclude (for instance, failed handshakes or interrupted sessions). UDP logs are recorded as complete flows given the connectionless nature of the protocol. Together, these sources provide a comprehensive view of traffic behavior before any aggregation.

Each raw record includes identifiers for client and server endpoints, timing information (start, end, duration), and directionality. It captures volume and activity indicators (bytes, packets, retransmissions), transport-layer dynamics (RTT statistics, window sizes, handshake counters), and higher-layer hints when available (HTTP activity, TLS server names or negotiated protocols, DNS-related metadata). Taken together, these signals provide a detailed representation of network behavior but come at the cost of high dimensionality, noise, and sparsity.

Because of this richness, direct analysis is impractical at scale. The aggregation pipeline described in Chapter 3 reduces the raw logs to a compact representation that preserves the most informative signals for behavior characterization and anomaly detection, while remaining feasible to process on large time windows.

2.4 Related work

2.4.1 Satellite-network measurements

Anomaly detection has a long history in data mining and network monitoring. A classical survey by Chandola *et al.* provides a structured overview of techniques and highlights how assumptions about normality, supervision, and computational cost influence detection effectiveness across domains [7]. These insights are particularly important for satellite traffic, where variability is high and labeled data are scarce, making unsupervised or weakly supervised methods more realistic and robust.

For satellite performance, Michel *et al.* present one of the first systematic measurement campaigns of Starlink, comparing LEO performance to traditional GEO-based access and analyzing latency, throughput, and web browsing behavior [25]. Their results confirm the potential of LEO systems to achieve competitive latency and throughput under light load, while also showing that delay and loss increase under heavier traffic, especially when avoiding performance-enhancing proxies by using QUIC. This work provides a baseline for understanding end-user experience in modern satellite systems and motivates the need for continuous traffic monitoring and anomaly detection.

This line of research is part of a broader evolution of non-terrestrial networking studies. Surveys and architectural analyses describe how the New Space ecosystem introduces new design opportunities and new operational constraints, from dynamic routing over moving constellations to latency-sensitive backbone design in orbit [12, 15, 16]. More recent measurement and security-oriented studies on Starlink further emphasize that performance and cyber risk must be analyzed jointly, especially in scenarios with mobility, variable load, and distributed edge conditions [26, 37]. These contributions strengthen the rationale for passive, longitudinal analysis workflows in satellite security monitoring.

2.4.2 Erroneous-traffic signals for passive security monitoring

Within network security and monitoring, erroneous traffic has emerged as a powerful signal for detecting anomalies. CHAMALEONET is a programmable passive probe that focuses on erroneous packets (e.g., unanswered requests or error messages) to reveal misconfigurations, infected hosts, and external scanning activity, while reducing collection overhead through in-network filtering [39]. A recent IMC poster complements this view by arguing that outbound erroneous traffic can unveil silent internal anomalies that would be missed by approaches centered on inbound traffic [34]. The poster identifies erroneous outbound traffic as a low-volume, high-signal subset that can expose misconfigurations, faulty clients, and compromised hosts, reinforcing the idea that selective monitoring can improve anomaly detection effectiveness.

2.4.3 Anomaly detection using passive traffic measurements

A large body of work has investigated anomaly detection using passive measurements and flow-level statistics. One of the earliest contributions by Barford *et al.* proposes signal processing techniques to detect network anomalies by analyzing deviations in aggregated traffic time series collected from backbone links [2]. Their work demonstrates that both large-scale failures and subtle anomalies can be identified without packet inspection, relying solely on passive observations.

Lakhina *et al.* further formalize this idea by modeling normal traffic behavior through statistical features extracted from IP flows and applying principal component analysis (PCA) to separate normal subspaces from anomalous components [19]. This approach

shows that anomalies such as flash crowds, routing changes, and attacks manifest as structural changes in traffic feature distributions. Extensions of this work explore robust PCA variants and multi-timescale analysis to improve detection accuracy and diagnosis capabilities [18].

Several studies emphasize the importance of multi-metric and adaptive approaches. Thottan and Ji propose anomaly detection methods based on statistical traffic models that capture correlations between multiple flow features, improving sensitivity to subtle changes [36]. Tellenbach *et al.* introduce robust estimators and adaptive baselines to reduce false positives in operational networks, demonstrating effectiveness on large ISP traces [35]. Similarly, Ringberg *et al.* analyze the impact of routing changes on traffic features, highlighting the difficulty of distinguishing benign anomalies from malicious events using passive data alone [30].

Classical outlier-detection literature has provided a rich set of unsupervised baselines that remain relevant in network telemetry. Representative methods include local-density scoring (LOF), one-class support estimation, isolation-based partitioning, angle-based outlier detection, and histogram-based scoring [4, 13, 17, 20, 32]. These approaches differ in geometric assumptions and computational profiles, but collectively establish the methodological foundation for anomaly screening when labels are scarce.

Clustering-based approaches are a natural fit for unsupervised anomaly detection because they can expose rare or structurally distinct users without requiring labeled ground truth. Foundational clustering methods include partitioning and graph-based techniques such as k-means [21] and spectral clustering [27], which provide baselines for grouping and separating traffic behaviors. Density-based methods are especially attractive for anomaly detection because they can label sparse regions as outliers and handle clusters with irregular shapes, as shown by DBSCAN [10], its multi-density extension OPTICS [1], and the hierarchical stability-driven HDBSCAN framework [5]. These techniques motivate the clustering workflows adopted later in this thesis for isolating anomalous users and traffic patterns from passive measurements.

In parallel, deep anomaly-detection methods have expanded the modeling space beyond fixed geometric assumptions. One-class deep representation learning and reconstruction-probabilistic hybrids, including Deep SVDD and DAGMM, demonstrate improved sensitivity to high-order patterns in complex feature spaces [31, 42]. Generative approaches based on adversarial learning have also been proposed for unsupervised anomaly scoring [40]. At a broader level, survey and evaluation studies stress that algorithmic gains must be interpreted together with protocol design, dataset characteristics, and metric selection [6, 29].

The limitations of supervised machine learning for network anomaly detection are discussed extensively by Sommer and Paxson, who argue that many learning-based approaches fail when deployed in real environments due to concept drift, lack of labeled data, and evolving traffic patterns [33]. This observation has led to a preference for unsupervised or weakly supervised techniques in large-scale monitoring systems.

Clustering-based approaches have also been proposed to identify anomalous hosts or behaviors without protocol-specific assumptions. Gu *et al.* introduce BotMiner, which detects botnet activity by correlating communication patterns and malicious behaviors inferred from passive traffic, without relying on predefined signatures [14]. While primarily focused on botnets, this work demonstrates how host-level aggregation of passive traffic features can reveal coordinated or abnormal behavior.

An alternative perspective is provided by Benson *et al.*, who analyze traffic in large data centers and show that rare or unexpected events often correlate strongly with failures,

misconfigurations, and performance problems [3]. Their findings suggest that low-volume traffic subsets may carry disproportionate diagnostic value, an idea that aligns closely with erroneous traffic-based anomaly detection.

More recently, programmable and selective monitoring approaches have been proposed to scale passive anomaly detection. CHAMALEONET exemplifies this trend by enabling in-network filtering of erroneous traffic, significantly reducing monitoring overhead while preserving anomaly detection capabilities [39]. This line of work is further supported by evidence that outbound erroneous traffic can reveal silent internal anomalies that are otherwise invisible in inbound-focused monitoring setups [34].

2.4.4 Residential-proxy and intermediary-ecosystem studies

Complementary literature on proxy ecosystems is also relevant for passive security analysis. Studies on residential proxies show that attribution is complicated by the decoupling between exposed source identity and operational traffic control [8, 23, 24]. Large-scale analyses of open proxies report heterogeneous behavior, integrity risks, and unstable trust properties across intermediary infrastructures [22, 28, 38, 41]. Evidence on HTTPS interception further confirms that intermediary roles can alter end-to-end security assumptions in ways that are not captured by origin-only heuristics [9].

Together, these studies demonstrate that passive traffic analysis, particularly at the flow and host aggregation level, provides a strong foundation for anomaly detection in complex and heterogeneous networks. However, most existing work focuses on terrestrial ISP or data center environments. Applying these principles to satellite Internet traffic, which exhibits unique characteristics such as higher latency, performance-enhancing proxies, and constrained access links, remains an open research challenge that this thesis aims to address.

Chapter 3

Aggregation pipeline and exploratory analysis

In this chapter the aggregation script is introduced as the central component that connects raw Tstat logs to analytics-ready datasets, and the resulting exploratory analyses are framed to support anomaly detection. The chapter follows a linear path from motivation and inputs, to pipeline design and outputs, and then to analysis, interpretation, and closing remarks.

3.1 Motivation and objectives

3.1.1 Why aggregation is required

Raw Tstat logs are too large and too granular to be analyzed directly: each record represents a single flow, while the research questions require views by user, service, and time interval. Without aggregation, the data remain fragmented, hard to compare across days, and poorly suited for detecting anomalous behavior.

Aggregation addresses three concrete issues. First, it reduces dimensionality while preserving useful signals (up/down bytes, flow counts, protocols) and makes metrics comparable through fixed time bins. Second, it harmonizes heterogeneous sources (TCP complete, TCP not-complete, UDP, and DNS), enabling visibility into both "normal" traffic and phenomena such as scans or incomplete handshakes. Third, it produces a stable format (aggregated CSVs) that separates parsing complexity from the rest of the analysis, making multi-day or multi-week studies reproducible.

In short, the aggregation script is the central node of the pipeline: it turns raw logs into analytic datasets, provides the foundation for exploratory analysis, and defines the context needed for anomaly detection.

3.2 Data sources and script inputs

3.2.1 Tstat logs and completeness

Tstat is an open-source passive traffic analysis tool developed for high-speed networks. It processes packet traces or live traffic to build flow-level summaries and statistics, providing a standardized view of network behavior.

Tstat inspects packets at a vantage point and reconstructs per-flow summaries, exposing both transport-level signals (e.g., TCP dynamics, timing, volumes) and application

hints (e.g., HTTP/TLS/DNS metadata) without requiring access to the endpoints. Its design makes it suitable for high-throughput environments where full payload inspection is impractical and where privacy constraints favor metadata-based analysis.

Within this study, Tstat serves as the measurement engine responsible for translating raw packet captures into structured log outputs. Selecting it keeps the aggregation pipeline anchored to uniform, clearly specified flow records. It also lets us merge complementary viewpoints, spanning connection-oriented alongside connectionless traffic, and both completed and incomplete handshakes, while maintaining consistency between the tooling choices and the underlying data model.

3.2.2 Inputs and configuration

The aggregation script is parameterized to ensure repeatability. The primary input is a reference day (or start day), which can be used to target a single 24-hour batch of `.out` files for quick runs and validation. For larger analyses, the script can recursively scan a root directory and its subfolders to discover all available logs, enabling a full month (or more) to be processed in a single execution without manual path listing. This design keeps the configuration explicit at the command line while allowing flexible batching.

Service classification relies on a `services.json` configuration file. Within this file, services are specified as a list, each entry providing a canonical name alongside one or more matching rules, for example, domains, SNI patterns, or HTTP hostnames, and an optional fallback category intended for flows that do not match any rule. Operationally, the file functions as a small, editable dictionary: it translates heterogeneous application-level hints into a consistent service label, while remaining straightforward to extend without changes to the aggregation code. With the mapping kept outside the codebase, services can be introduced or withdrawn whenever needed, and this separation also supports basic filtering, such as removing rules that would otherwise assign traffic to a generic `other` category. Listing 3.1 provides a brief extract.

Listing 3.1: Excerpt of `services.json` used for service classification.

```
{
  "Video": {
    "youtube": [
      ".googlevideo.com$",
      ".ytimg.com$",
      ".youtube.com$"
    ],
    "netflix": [
      ".netflix.",
      ".nflxvideo."
    ]
  },
  "Search engine": {
    "google": [
      "^www.google.",
      "^google."
    ]
  }
}
```

3.3 Aggregation script architecture

3.3.1 Pipeline overview

Aggregation is carried out through a single, deterministic pipeline in which raw traffic records pass through an invariant, ordered set of transformations. Within the chosen time horizon, the workflow begins by locating all admissible input traces, checking temporal coherence, then partitioning the data by transport family, namely complete TCP, complete UDP, and incomplete TCP. Although processed as separate families at this stage, all traces pass through the same normalization layer; only afterward does protocol-oriented preparation extract timestamps, set the protocol label, and keep only those fields needed for subsequent analytics.

Traffic records are then enriched through rule-based service attribution. The classification stage combines multiple application-level evidence sources, including encrypted-session host indicators, web host metadata, domain-level hints, and protocol-specific naming fields when available. When no reliable match is found, records are mapped to a residual unlabeled category. Once enrichment is complete, the protocol families are recombined into a single tabular dataset with a consistent schema and time-aligned semantics, and each flow is assigned to an hourly observation bin.

The final step aggregates the unified table by day, client, hourly bin, service class, and protocol. For every resulting group, directional byte volume and flow counts are calculated, yielding compact per-day summaries suitable for large-scale longitudinal work. Results are emitted as one file per day, which supports multi-day or multi-week analyses by straightforward concatenation of daily aggregates, without requiring renewed processing of the underlying raw traces. Figure 3.1 provides an overview of the processing chain.

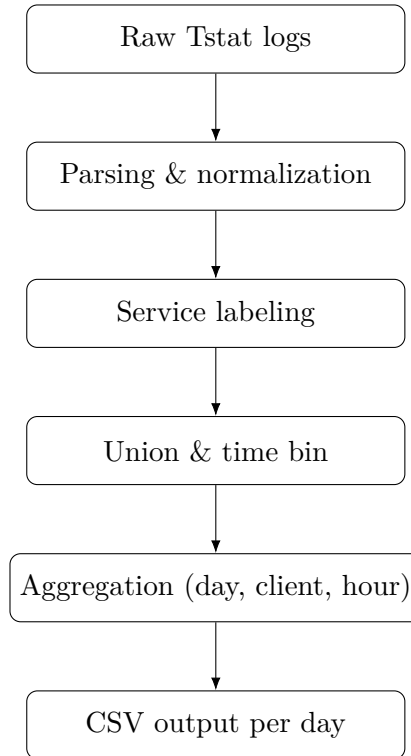


Figure 3.1: Block-level view of the aggregation pipeline.

The block diagram highlights the linear flow from raw logs to daily outputs. In practice,

each stage is implemented as a small, isolated transformation so that parsing, labeling, time binning, and aggregation can be validated independently and adjusted without affecting the rest of the pipeline. This modular design is useful when services are updated or when new log families must be integrated.

Listing 3.2: Core service-labeling logic in the aggregation script.

```
def build_service_udf(spark, services_json_path):
    with open(services_json_path, "r", encoding="utf-8") as fh:
        raw = json.load(fh)
    import re as _re
    rules = []
    for cat, services in raw.items():
        for svc, pats in services.items():
            regs = [_re.compile(p, _re.IGNORECASE) for p in pats]
            rules.append((cat, svc, regs))
    bc = spark.sparkContext.broadcast(rules)
    @f.udf(t.StringType())
    def udf(c_tls_SNI, http_hostname, fqdn, quic_SNI):
        for host in (c_tls_SNI, http_hostname, fqdn, quic_SNI):
            h = (host or "").strip()
            if not h:
                continue
            for cat, svc, regs in bc.value:
                for rgx in regs:
                    if rgx.search(h):
                        return f"{cat}:{svc}"
        return "Unknown"
    return udf
```

Listing 3.2 shows the operational core of service attribution: matching rules are compiled once, broadcast to workers, and applied in order over the available hostname/SNI evidence fields. This design keeps classification consistent across partitions and allows deterministic fallback to `Unknown` when no reliable pattern is observed, avoiding forced or noisy labels in downstream aggregation.

3.4 Spark execution model and scalability considerations

3.4.1 Execution setup and parameterization

Execution is performed in PySpark local mode and configured for high-volume batch processing on a single host. All available local cores are enabled (`local[*]`), Arrow acceleration is activated for efficient columnar interchange, shuffle parallelism is fixed to 128 partitions to avoid under-parallelization during grouping stages, and both driver and executor memory are set to 12 GB to reduce spill pressure in multi-day unions and aggregations. A dedicated Spark local directory is also configured to control temporary I/O placement.

The parameter choices were set to satisfy three practical objectives at once: adequate throughput for wide aggregations, stable memory behavior across repeated runs, and predictable, deterministic results from one execution to the next. By holding shuffle parallelism and memory limits constant, runtime and resource measurements remain comparable when the same setup is carried across distinct observation horizons.

At a broad level, moving to Spark reorients the analysis away from file-based, sequential routines and toward distributed relational execution on partitioned tables. Several

implications follow. Memory load no longer accumulates in a single process because the workload is split into partition-scoped tasks under scheduler control. Costly operations, including joins, wide aggregations, and repartition steps, are mediated by explicit execution plans, so performance constraints become visible and can be adjusted. The same analytical logic also runs over longer time windows without modifying code paths or altering the meaning of the analysis.

3.4.2 How Spark is used in the pipeline

Spark is not used only as a runtime substrate, but as the core execution model of the aggregation logic. The full workflow is expressed as a sequence of distributed DataFrame transformations: schema-normalized ingestion, protocol-family preparation, service enrichment, union on compatible columns, hourly bin assignment, and final grouped aggregation at (day, client, hour, service, protocol) level.

Two implementation choices are particularly relevant. First, service-matching rules are compiled once and broadcast to workers, so classification is evaluated locally on each partition without repeated driver-side lookups. Second, transformations are kept modular and column-selective, which reduces unnecessary data movement and makes each stage independently verifiable. This design simplifies maintenance when mapping rules evolve or additional log families are incorporated.

From a methodological perspective, distributed execution also supports reproducibility. The same transformation graph is applied uniformly to all daily partitions, reducing analyst-dependent variability and making the link between raw records and final aggregate features explicit and auditable.

Viewed from an implementation angle, the pipeline is built around a small, tightly bounded set of distributed primitives, and these remain consistent from one stage to the next. At ingestion, schema-aware readers are paired with column projection so early I/O stays constrained. Temporal normalization and protocol harmonization are handled as deterministic column-level transforms. For service attribution, a user-defined function is run over the hostname and SNI evidence fields, while the rule dictionaries are propagated through broadcast variables. Family merging proceeds via schema-compatible unions. To obtain the final indicators, the system applies grouped aggregations keyed by analytical identifiers, then performs controlled repartitioning or coalescing steps prior to export. The result is an execution graph that stays compact and predictable, yet still leaves room to add further enrichment rules when needed.

Relative to a single-node, non-distributed approach, the practical gains are substantial. Runtime tracks input growth more effectively, failures within intermediate steps are easier to pinpoint at the stage boundary, and operational reruns can be limited to the partitions that were impacted instead of reprocessing the entire dataset. Also, Spark's lazy evaluation allows the physical plan to be refined before any computation is triggered, which cuts down on redundant scans and helps avoid repeatedly materializing intermediates that are functionally the same. Taken together, these characteristics raise both engineering throughput and the methodological dependability of the resulting dataset.

3.4.3 Output materialization strategy

Output generation follows a per-day materialization strategy. For each day, the aggregated partition is first written to a temporary directory as a single CSV part, then moved to `aggregation.csv` under `output/<day>/`. This two-step export avoids ambiguous multi-

part outputs and provides a stable file name for downstream notebooks and batch concatenation.

The daily layout has two practical advantages. It isolates failures and reruns to a limited temporal block, and it enables long-horizon analyses through deterministic concatenation of already materialized daily files, without reprocessing the full raw corpus. As a result, exploratory analysis and anomaly-detection stages can iterate on fixed aggregated snapshots while preserving provenance to the original raw window.

An additional advantage of this strategy is analytical traceability. Since each daily output is produced by the same transformation plan and naming convention, downstream notebooks can reference immutable input slices, compare results across reruns, and audit differences without ambiguity on data lineage. This property is particularly important in anomaly studies, where reproducibility of candidate selection depends on strict consistency of upstream preprocessing.

Field	Type	Meaning
day	string	Reference day for the aggregation (YYYY_MM_DD).
c_ip	string	Client identifier (anonymized IP).
hour_bin	string	One-hour bin in HH-HH format.
service	string	Service label derived from <code>services.json</code> .
protocol	string	Traffic family (TCP, UDP, TCP_NC).
bytes_down	long	Bytes from server to client in the bin.
bytes_up	long	Bytes from client to server in the bin.
flows	long	Number of flows aggregated in the bin.

Table 3.1: Schema of the daily aggregation output CSV.

Chapter 4

Exploratory analysis and feature engineering

The opening section of the anomaly detection notebook relies on exploratory analysis to surface the dataset’s core behavioral signatures. Attention is placed on locating persistent regularities in user activity, patterns of service use, and fluctuations in traffic intensity, and this is done before any unsupervised detection procedure is applied. The work proceeds in a strictly data-driven manner, with an explicit effort to disentangle structural, systematic effects from activity that is genuinely atypical. Establishing this separation supplies the operational frame needed to specify what counts as normal behavior in the network under observation.

By the end of that section, the environment is characterized in concrete terms, derived directly from the data rather than imposed assumptions. Such a baseline is central to anomaly detection, since it fixes the expected patterns and makes the underlying variability visible, which detection methods must accommodate to avoid confusing routine variation with anomalous events.

4.1 Data validation and visualization

The first phase, before any clustering step, is designed to extract behavioral information from aggregate traffic and to define a reliable baseline for anomaly detection. In this perspective, visual analysis is used as supporting evidence to validate assumptions, isolate structural effects, and identify the dominant regularities of user and service behavior.

4.1.1 Why traffic is split by IP entity type

Before computing user or service statistics, traffic is partitioned into three entity types:

- IPv4 users (single `c_ip` addresses),
- IPv6 users (single `c_ip` addresses),
- IPv6 /64 prefixes (first 64 bits of IPv6 addresses).

This separation is methodologically necessary because IPv4 and IPv6 provide different identity stability at measurement time. In operational networks, IPv4 addresses observed at aggregation points are often comparatively stable over short and medium time windows. By contrast, a single endpoint or household can appear under multiple IPv6 interface identifiers over time (e.g., due to privacy extensions, address rotation, or multi-device

behavior within the same delegated prefix). Consequently, the direct use of full IPv6 addresses can artificially fragment the activity of one logical user into multiple entities.

Introducing an IPv6 /64 representation mitigates that problem by shifting to a more stable unit of aggregation that better matches user- or household-level behavior in IPv6 settings. The empirical results support this choice. While the overall shapes of the distributions stay broadly consistent from day to day, both central tendency and spread vary by entity definition, and IPv6 /64 prefixes show systematically larger volumes and wider variability because they combine traffic from multiple hosts. Treating these groups as one population would distort the baseline and raise the chance that structural properties of addressing are mistaken for anomalies.

4.1.2 Generic user analysis

After the IP-based partition, user-level analysis highlights two core outcomes: the stability of the observed population and the consistency of effective daily activity.

Unique users per day. The number of distinct client ips entities per day is used to characterize the baseline exposure of the monitored population. The observed pattern is regular, with limited day-to-day variability and a recurrent weekend reduction, indicating that the dataset captures stable operational dynamics rather than abrupt structural changes.

Active users per day. Active users are identified through a minimum daily traffic threshold of `total bytes transmitted > 1MB` to isolate behavior with substantive data exchange. The pattern that emerges aligns with the same timing regularity already visible in the total-user series, and it suggests that most of the day-to-day movement is driven by structural effects rather than isolated anomalies. What remains between the total and the active counts largely reflects idle endpoints, background polling processes, and other low-intensity forms of traffic. Over time, the dynamics of active entities appear steady, marked by recurring drops during weekends followed by a rebound at the start of the next week.

In terms of protocol mix, IPv6 entities dominate in absolute number, while IPv4 entities persist at a lower level that nonetheless remains consistent. At the prefix level, IPv6 /64 aggregates trace a smaller-amplitude trajectory and, compared with host-level counts, exhibit a smoother evolution. The active IPv6 address count, however, should not be read as a direct measure of distinct human users. Given common IPv6 addressing behavior, including temporary or privacy addresses and routine rotation, a single logical user can be represented by multiple IPv6 addresses across time. To address this mismatch, /64 aggregation is presented together with host-level IPv6 totals, offering a more defensible proxy for user-level activity.

4.1.3 Service-level analysis

At service level, the first relevant result is the predominance of the **Unknown** class. This reflects the scope of the adopted mapping strategy, which targets major public platforms and selected private services. A large share of ordinary traffic generated by enterprise software, long-tail websites, and heterogeneous minor applications therefore remains outside explicit labels.

From a methodological perspective, this prevalence defines a coverage boundary of the labeling process rather than an anomaly indicator. In this context, **Unknown** should

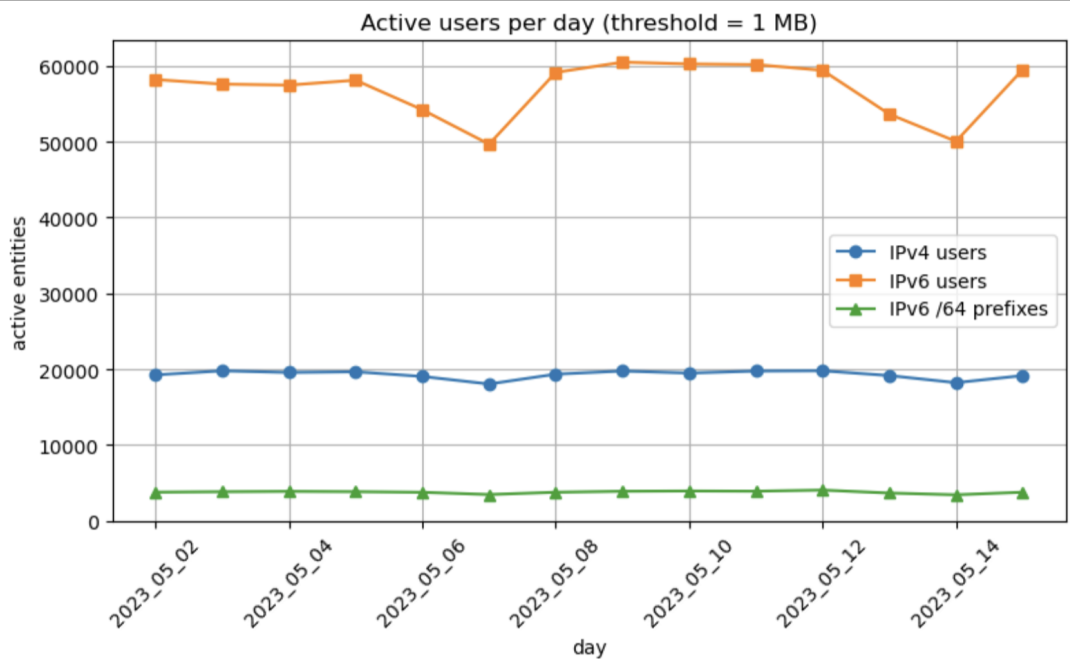


Figure 4.1: Active users per day (threshold = 1 MB), reported separately for IPv4 hosts, IPv6 hosts, and IPv6 /64 prefixes.

be interpreted as “not mapped by the current rule set” and not as inherently suspicious traffic. Its high incidence is expected in passive traffic measurements, where service attribution depends on partial metadata (e.g., SNI/hostname evidence) and where encrypted or weakly classifiable flows are common.

For this reason, **Unknown** is retained in the global traffic accounting, but excluded from comparative rankings (*top* and *least* services), so that relative differences are assessed only across explicitly identified services.

Service traffic is strongly concentrated in a restricted set of categories. Video platforms dominate the ranking, with a marked gap between the first positions and the remaining services, confirming a heavy-tailed service-consumption profile.

Top services. The dominant behavioral result is a strong concentration of network load on a small set of services. In particular, high-bandwidth categories (mainly video) account for a disproportionate fraction of total bytes, while widely used services do not necessarily produce comparable traffic volumes. This separation between adoption and traffic impact is central for feature design.

Tail services and cross-IP consistency. The same dual perspective (traffic volume and unique users) is then applied to low-incidence services to characterize the long tail. These services are interpreted as niche applications, legacy endpoints, geographically limited platforms, or residual mapping noise. Their inspection is useful for validating label quality and for contextualizing sparse events without over-interpreting them as anomalous by default.

When this tail-level evidence is compared with top-service rankings, the apparent IP-version differences become substantially weaker. Mainstream service preferences remain largely consistent across IPv4, IPv6, and IPv6 /64 entities, while residual mismatches are

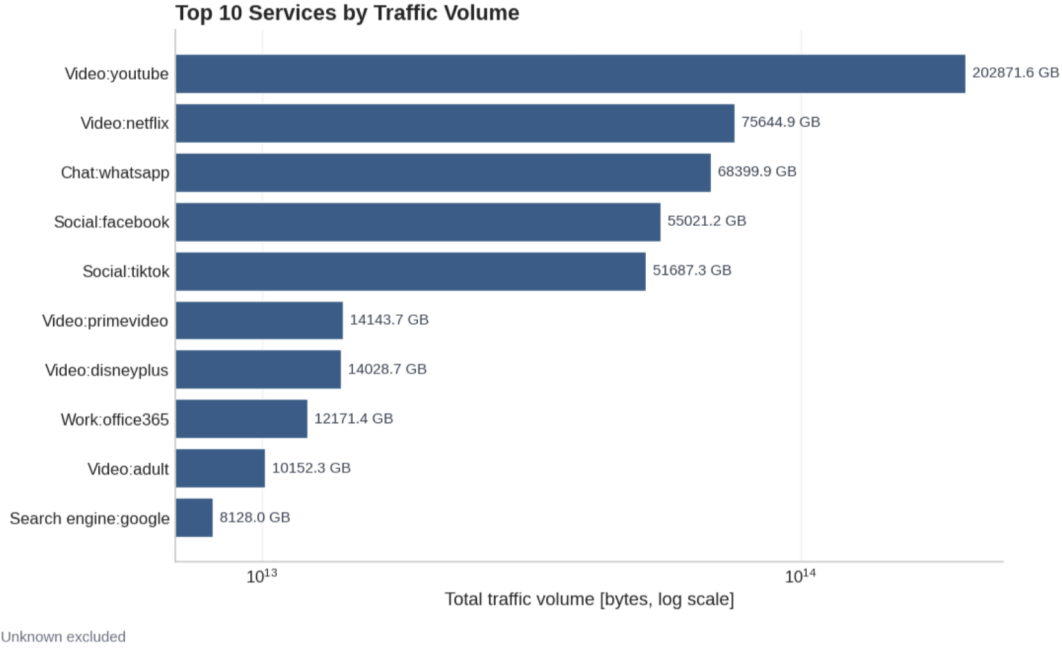


Figure 4.2: Top 10 identified services by total traffic volume (log scale), excluding **Unknown**.

better explained by infrastructure constraints (e.g., services with limited IPv6 support) than by genuine behavioral divergence.

Overall, the pre-clustering service analysis provides two validation outcomes: traffic composition is highly concentrated in a small number of major services, and most cross-IP differences are structural (protocol support and aggregation effects) rather than intrinsically anomalous.

4.1.4 Traffic distribution and user variability

To complete behavioral profiling, traffic is evaluated at user-day granularity and across-user variability over time. The main result is the coexistence of a stable behavioral majority and a sparse set of burst-prone entities.

Baseline intensity across entities. The distributional evidence indicates that baseline behavior is stable across days: medians and interquartile ranges vary only marginally, which confirms temporal consistency of ordinary user activity. At the same time, entity types remain structurally distinct. IPv6 /64 prefixes show systematically higher central traffic and broader variability, coherently with multi-device aggregation, while IPv4 users tend to exhibit slightly higher central intensity than single IPv6 hosts. A persistent upper tail is present in all groups, but it involves only a small fraction of entities and therefore represents deviation from, rather than definition of, the baseline.

From an anomaly-analysis perspective, these results imply that behavior must be modeled within homogeneous entity types and on log-scale deviations, not through a single global threshold. For user-level interpretation, IPv4 remains the most reliable reference because addresses are comparatively more stable in the observation window.

Intra-user temporal variability. User-level variability confirms the same structure: most entities remain concentrated in low-volume regimes, while a smaller subset reaches

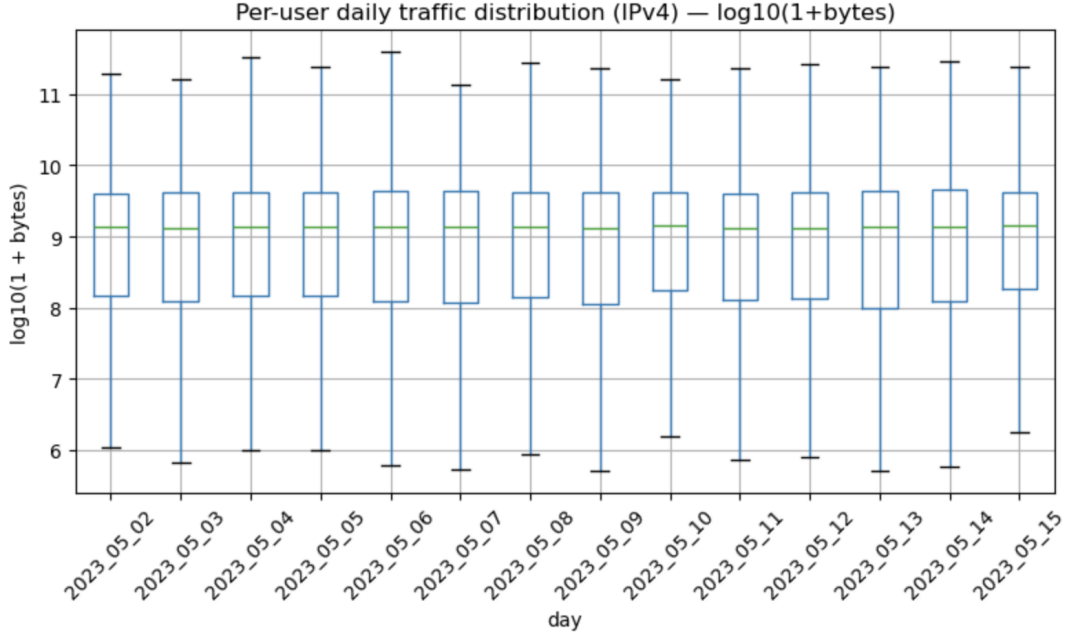


Figure 4.3: Daily traffic distribution of IPv4 users. This view is prioritized for user-level interpretation because IPv4 addresses are generally more stable proxies of individual entities in the observation window.

markedly higher maxima, revealing burst-oriented behavior.

The min–max analysis separates three behavioral regimes: a dense majority with low and stable daily traffic, an intermediate group with low minima but occasional high spikes, and a very small set of extreme profiles with maximum daily volumes up to 4×10^{11} bytes. This last group is far from the central mass and constitutes a priority target for anomaly investigation.

In summary, the data show a stable central regime and a sparse heavy tail. This supports anomaly modeling based on relative deviations within homogeneous entity types, rather than on global absolute thresholds.

4.2 Feature engineering

The feature-engineering stage transforms aggregated logs into user-level descriptors that can represent both traffic intensity and behavioural structure. The design follows a progressive logic: first, compact volume statistics are built; then temporal and directional indicators are added; finally, higher-order hourly and service-profile descriptors are included to capture fine-grained regularity.

4.2.1 Volume feature set

The initial feature block is computed from per-user daily totals (`total_bytes` aggregated over each `(c_ip, day)` pair). For each user, the notebook derives:

- `mean_bytes`: average daily traffic volume,
- `std_bytes`: day-to-day traffic variability,
- `max_bytes`: maximum observed daily volume,

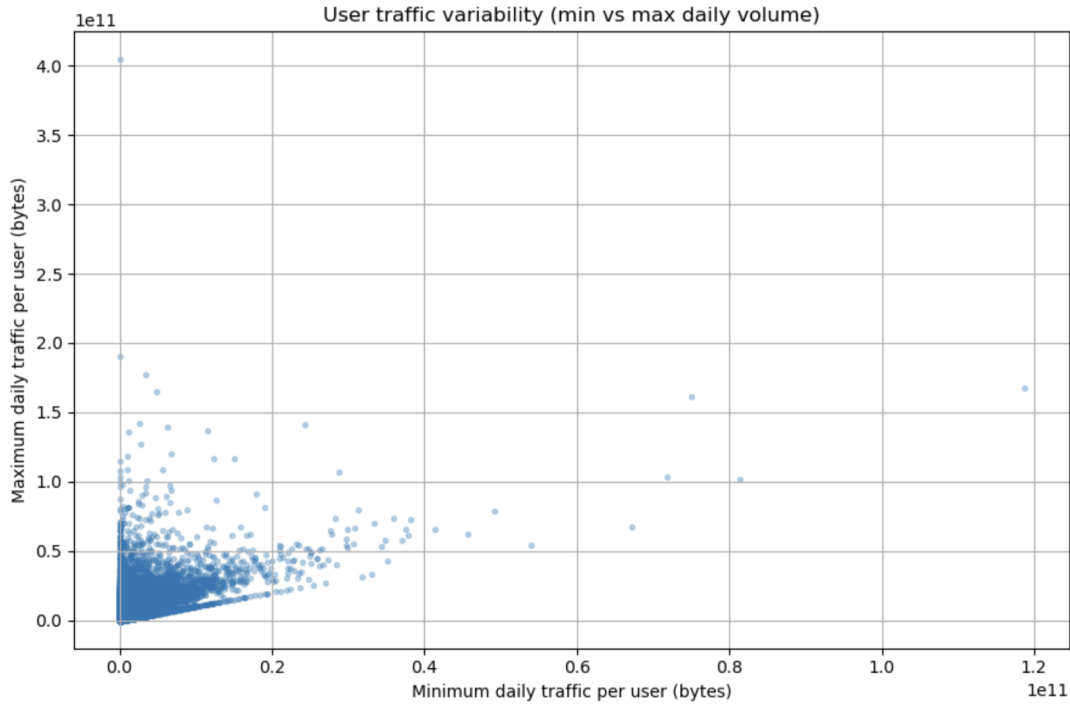


Figure 4.4: User-level variability in the (min_daily bytes, max_daily bytes) space.

- `total_bytes`: cumulative traffic volume in the observation window,
- `active_days`: number of observed active days,
- `cv`: coefficient of variation, computed as $\text{std_bytes}/(\text{mean_bytes} + 1)$.

This set captures two complementary dimensions: overall scale (how much traffic is generated) and stability (how regular or burst-prone traffic is over days). It provides the baseline representation for volume-space clustering.

4.2.2 Behavioural feature set

To move beyond pure magnitude, the notebook builds behavioural features from hourly aggregates.

Day/night and directional features. Hours are split into daytime (08:00–19:59) and nighttime. For each user and day, total/downlink/uplink bytes are computed separately in day and night periods. From these daily values, the following indicators are derived:

- `day_night_ratio`: $(\text{total_bytes_day} + 1)/(\text{total_bytes_night} + 1)$,
- `down_up_ratio_total`: global downlink/uplink ratio with additive smoothing,
- user-level summaries of these ratios (mean and median),
- `days_observed` and `mean_total_bytes`.

These descriptors encode temporal preference and traffic-direction asymmetry, which are often more informative than raw volume for identifying automated or policy-driven activity.

Advanced behavioural features. From per-user hourly traffic and flow counts, three additional indicators are introduced:

- `mean_flow_size`: total hourly bytes divided by total hourly flows,
- `flow_burstiness`: $\max(\text{hour_flows}) / \text{mean}(\text{hour_flows})$,
- `traffic_concentration_top2`: fraction of user traffic concentrated in the two most active hours.

Together, these variables describe whether a user behaves in a continuous regime or in concentrated bursts, and whether activity is spread across time or constrained to narrow windows.

4.2.3 Normalization and user signature representation

The engineered variables have very different scales and statistical distributions. Without normalization, high-magnitude features would dominate the representation and obscure weaker but behaviorally relevant signals.

For this reason, the feature matrix is standardized before downstream modeling, so that each component contributes comparably to similarity and deviation analysis. This step is particularly important in heavy-tailed traffic data, where raw magnitudes can span several orders of magnitude.

In methodological terms, normalization is not a cosmetic preprocessing step: it is required to make behavioral dimensions jointly interpretable and to preserve sensitivity to subtle deviations.

The final output of feature engineering is a user signature that combines temporal descriptors and service-related information in a single structured vector. In the notebook, this representation reaches high dimensionality (130 features) by integrating:

- hourly behavior profiles over the full daily cycle,
- traffic-direction components and flow dynamics,
- service-usage shares over major identified services.

This representation is designed to preserve both macro-patterns (overall volume and direction) and micro-patterns (hourly regularity and service mix), providing a robust basis for the anomaly-detection stage described in the following section.

Chapter 5

Anomaly detection and evaluation

This chapter presents the anomaly-detection stage as a combined workflow composed by clustering methods and deep representation learning. The objective is to assess whether abnormal users can be isolated through density structure alone and, when this is not sufficient, to capture more complex behavioural deviations through autoencoder-based reconstruction analysis.

The focus of this chapter is on models configuration, experimental protocol, and interpretation of results across both families of methods (clustering and autoencoder), including their integration for final anomaly prioritization. To preserve methodological clarity without duplicating content, each analytical block is organized in a consistent order: first the protocol and modeling choices, then the corresponding empirical evidence and interpretation.

5.1 Methodological roadmap

The anomaly-detection workflow is organized as a progressive pipeline with three complementary stages.

Stage 1: Clustering exploration (volume and behavioural branches). The first stage tests whether anomalous users can be identified through density structure alone. This exploration is explicitly divided into two branches:

- a *volume branch*, based on sole traffic-intensity descriptors;
- a *behavioural branch*, based on temporal and directional usage by the users.

The rationale is to separate anomalies linked to traffic magnitude from anomalies linked to patterns on how the traffic is generated.

Stage 2: Deep representation learning. Because clustering may fail when anomalies are subtle and non-cluster-forming, a deep autoencoder is introduced to model normal behaviour in high-dimensional user signatures. Reconstruction error is then used as a continuous anomaly indicator.

Stage 3: Internal anomaly score integration. Signals from clustering and autoencoder analysis are combined into a unified internal score, producing a single prioritized ranking of suspicious users.

Evaluation logic. In absence of ground-truth labels, model quality is assessed through cross-method consistency, stability of top-ranked users, qualitative plausibility of detected behavioural profiles and manual inspections.

5.2 Clustering-based anomaly exploration: methodology and results

Clustering is an unsupervised learning paradigm that groups observations according to similarity, without predefined class labels. Given a feature space, the objective is to partition data into internally coherent groups (high intra-cluster similarity) that are distinct from each other (low inter-cluster similarity).

From a methodological perspective, clustering algorithms can be organized into major families. Partition-based methods (e.g., k-means) optimize compactness around centroids and are efficient on large datasets, but require specifying the number of clusters and are sensitive to scale and initialization. Hierarchical methods build nested cluster trees and support multi-resolution analysis, at the cost of higher computational complexity. Density-based methods (e.g., DBSCAN and HDBSCAN) identify dense regions separated by sparse areas, naturally handling non-convex shapes and explicitly modeling noise points.

Clustering is widely used in network and security analytics. Typical applications include traffic profiling, user-behaviour segmentation, endpoint role discovery, botnet campaign grouping, and exploratory triage of suspicious entities. In these contexts, clustering is rarely a final decision mechanism by itself; rather, it is an exploratory layer that helps distinguish dominant operational patterns from minority behaviours requiring deeper inspection.

A critical aspect is that clustering quality is strongly dependent on representation. Feature scaling, transformation of heavy-tailed variables, and handling of sparsity can change cluster geometry substantially. Preprocessing therefore cannot be treated as a minor technical step, it is part of what makes clustering results defensible. Interpretation also requires care, since "distance" in a feature space reflects statistical proximity and need not correspond to behavioural meaning.

In the absence of ground-truth labels, evaluation relies on internal and stability-oriented criteria. Typical checks include the resulting distribution of cluster sizes, the share of points classified as noise, how strongly results vary with hyperparameter choices, and whether similar structure reappears under different projections or across subsamples. None of these criteria establishes correctness in any absolute sense; instead, they indicate whether the observed structure persists under reasonable perturbations or is largely a byproduct of the analysis pipeline.

Within this thesis, clustering is therefore treated as a structural analysis tool: it is used to test whether anomalies emerge as separable density patterns and to provide interpretable geometric evidence before introducing reconstruction-based anomaly scoring. In what follows, parameterization and protocol are presented first, followed by the observed outcomes in volume, behavioural, and PCA-supported spaces.

5.2.1 DBSCAN

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) identifies clusters as connected dense regions in feature space and labels sparse observations as noise. In contrast to centroid-based methods, it does not require the number of clusters to be

fixed in advance and can capture non-convex geometries, which is particularly relevant for heterogeneous network behaviours.

Operationally, DBSCAN relies on two control hyperparameters:

- **eps**: neighborhood radius used to evaluate local density;
- **min samples**: minimum number of neighbors required to define a dense region.

Given these two parameters, points are classified into three categories. A *core point* has a neighborhood dense enough to satisfy the **min samples** criterion within radius of **eps**. A *border point* is not dense by itself, but it is directly reachable from a core region. A *noise point* is not density-reachable from any core region and is therefore treated as an outlier candidate. In practice, in DBSCAN, an anomaly could be identified as a noise point.

Strengths and limitations

DBSCAN is widely adopted in anomaly-oriented studies because it provides an explicit geometric notion of rarity: observations outside dense regions are not forced into arbitrary clusters and remain identifiable as potential anomalies. At the same time, the method is sensitive to data representation and local-density heterogeneity.

Table 5.1 summarizes the main trade-offs in a form directly relevant to traffic analysis.

Aspect	Strength	Limitation
Model setup	No need to predefine the number of clusters.	Sensitive to eps/min_samples selection.
Cluster geometry	Handles non-convex cluster shapes.	Can underperform when local densities vary substantially.
Anomaly handling	Explicit noise labeling, useful for anomaly triage.	Strong dependence on feature scaling and metric choice.
Interpretability	Clear dense-core vs sparse-boundary interpretation.	Distance concentration may reduce contrast in high-dimensional spaces.

Table 5.1: DBSCAN trade-offs for anomaly-oriented traffic analysis.

Use in literature and thesis positioning

In network-security literature, DBSCAN is commonly used for exploratory anomaly detection, bot/scanner behaviour grouping, endpoint profiling, and preliminary triage of suspicious entities. Its role is typically diagnostic rather than definitive: it reveals structure and sparse deviations, but it is usually combined with complementary models for robust ranking.

This thesis follows the same strategy. DBSCAN is used as the primary density-based exploration tool, supported by preprocessing (log-transform and standardization), parameter diagnostics (k-distance analysis), and cross-checking with alternative methods in subsequent sections.

Volume-space exploration

The baseline density experiment is performed on the compact volumetric feature set (`mean bytes transmitted`, `std of bytes transmitted`, `maximum of bytes transmitted in a single flow`, `sum of total bytes transmitted in both direction`, `active days`, `cv`) over a 50,000-user IPv4 sample.

The *k-distance* diagnostic is used to guide the density threshold selection. Concretely, for each point the distance to its 5-th nearest neighbor is computed ($k=5$, coherent with `min_samples=5`), then distances are sorted and inspected. The interpretation is standard: low, slowly varying distances correspond to dense regions, while the right-tail increase marks transition toward sparse regions and potential outliers. In this experiment, the reported statistics are $\min = 0.0$, $\text{median} = 0.0$, and $\max \approx 41.32$, indicating a very dense core with a limited but non-negligible sparse tail.

Based on this profile, DBSCAN is executed with `eps=0.3` and `min_samples=5`.

The partition is strongly unbalanced: one dominant cluster contains 49,322 users, while 587 users are labeled as noise (1.17%); the remaining clusters are micro-groups (5–43 users). This configuration is coherent with the volume-space interpretation reported in the notebook: most users lie on a regular manifold where mean traffic and variability scale together, and only a limited fraction occupies sparse regions.

Figure 5.1 summarizes four complementary projections (*mean vs std*, *mean vs max*, *mean vs CV*, *mean vs burst ratio*) with DBSCAN noise highlighted. Two patterns are particularly relevant. First, in the *mean–std* and *mean–max* panels, noise points are largely aligned with the global trend and only a minority appears at clearly extreme magnitudes. Second, in the *mean–CV* and *mean–burst ratio* panels, many noise points fall near low-density fringes of plausible behavioural regions rather than in fully detached clouds.

Taken as a whole, clustering driven only by volume offers a helpful sense of structure, yet it does not separate anomalies particularly well. Traffic magnitude, by itself, fails to pin down what should count as abnormal, and a number of the users marked as outliers still display behaviour that fits comfortably within the dominant population. These observations motivate a shift toward descriptors defined in behavioural space, since temporal patterning and directional asymmetry contribute extra information with clearer discriminative value.

Behavioural-space exploration

The behavioural-space experiment is designed to test whether anomalies become more distinguishable when the feature space describes *how* traffic is produced, rather than only *how much* traffic is exchanged. DBSCAN is applied on a 50,000-user IPv4 sample using behavioural descriptors that encode day/night preference, download/upload balance, mean flow size, burstiness, traffic concentration over peak hours, and activity persistence.

At global level, the density structure remains dominated by one large cluster, with 49,378 users, and 541 points labeled as noise (1.08%) at the selected operating point ($\text{eps} \approx 0.4718$, `min_samples=6`). Therefore, the quantitative separation power is comparable to the volume-only setting. The improvement is mainly qualitative: outlier points become behaviourally interpretable, because their position can be linked to specific traffic styles (e.g., strong day-only usage, upload-heavy activity, or burst-concentrated patterns).

This is the key reason behavioural features are beneficial. Descriptors based on volume tend to register scale alone, leaving many users compressed onto a single monotonic manifold. By contrast, behavioural descriptors add information that is orthogonal to scale,

DBSCAN Volume-Space Projection: Clustered Users vs Noise

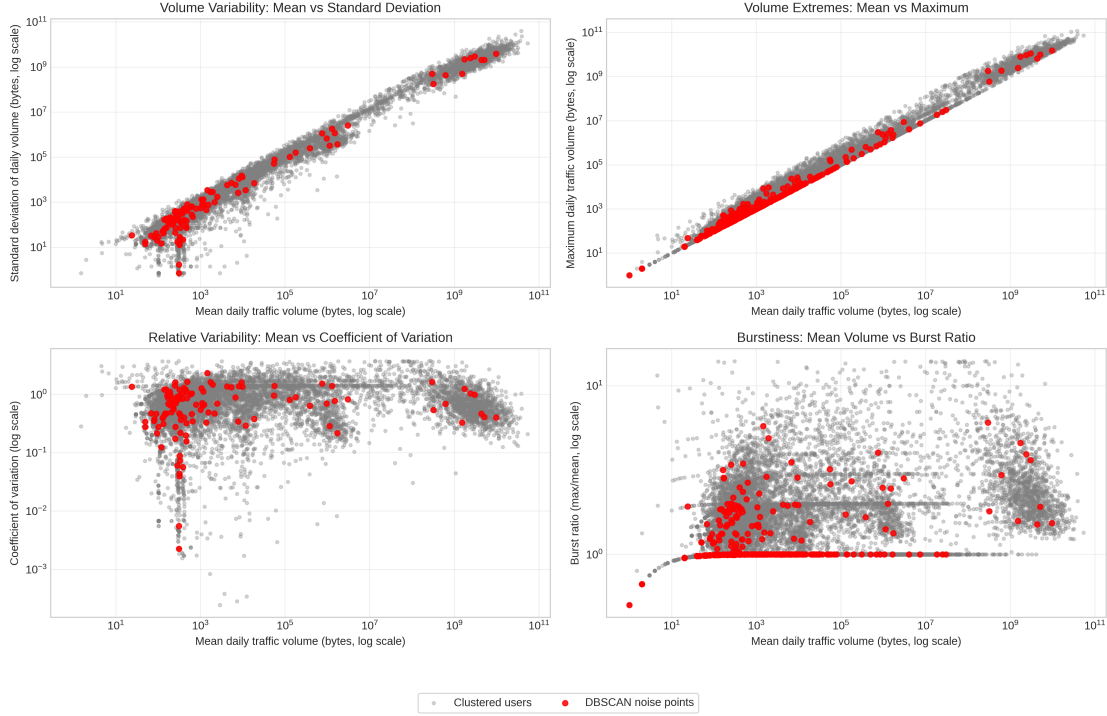


Figure 5.1: DBSCAN noise highlighted in compact volume space across four projections: mean vs std, mean vs max, mean vs CV, and mean vs burst ratio.

capturing temporal structure and directional asymmetry, and this raises explanatory resolution even in cases where the count of detected outliers remains broadly stable.

The day/night versus down/up projection (Figure 5.2) shows that the majority of users is concentrated around balanced temporal activity and download-dominated traffic, which is consistent with ordinary browsing and streaming behaviour. The noise points concentrate in two behaviourally meaningful areas. One cluster is characterised by download traffic that is heavily skewed toward daytime hours, a pattern that often aligns with intensive yet legitimate consumption of content. The other cluster shows a pronounced bias toward uploads, with activity restricted to either daytime-only or night-only intervals; this region is more informative for anomaly triage, since it can be consistent with scheduled backups, automated replication routines, or scripts whose timing resembles exfiltration.

The burstiness versus mean flow-size view (Figure 5.3) provides a complementary interpretation. Across the sample, user behaviour concentrates in a mid-range band, with neither flow size nor burstiness taking extreme values in most cases. Two recurring forms of noise are visible. One consists of observations with a large average flow size but only modest burstiness, which aligns with bulk movement of data or backup-style workloads. The other consists of cases where burstiness is pronounced while flow size remains typical, a profile that fits intermittent automated activity. The result’s interpretation also points out that some noisy observations sit close to the primary cluster and could reflect valid, if sparsely represented, behaviour. Greater attention is warranted for the small set combining very high flow size with high burstiness, since these points separate more distinctly from the central structure of the population.

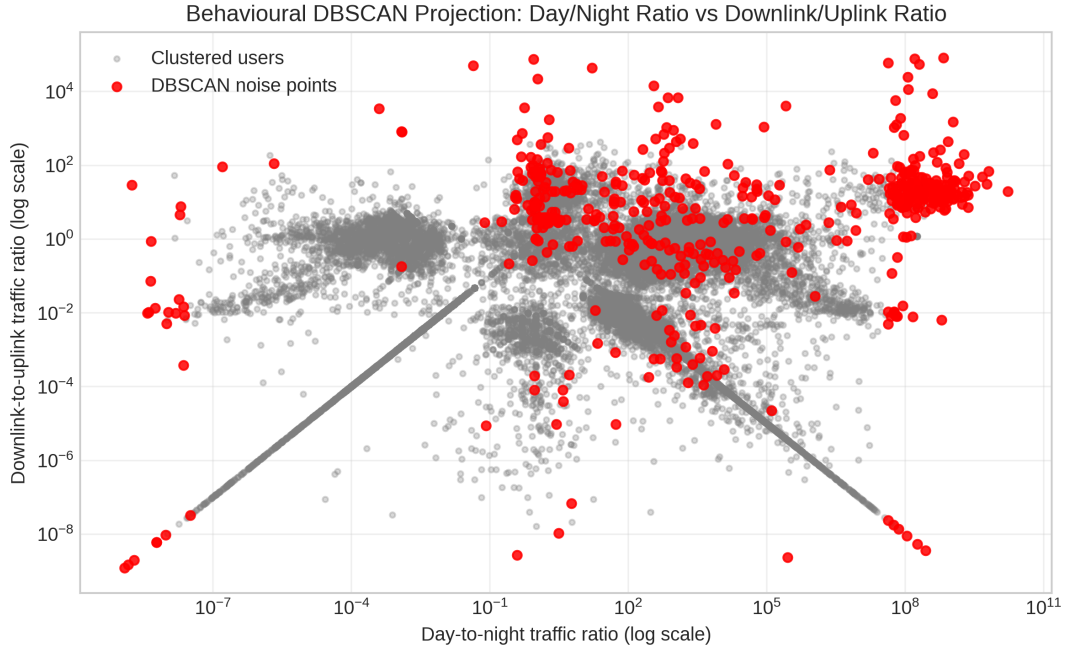


Figure 5.2: DBSCAN in behavioural space: day/night traffic ratio versus downlink/uplink ratio (log scale), with noise points highlighted.

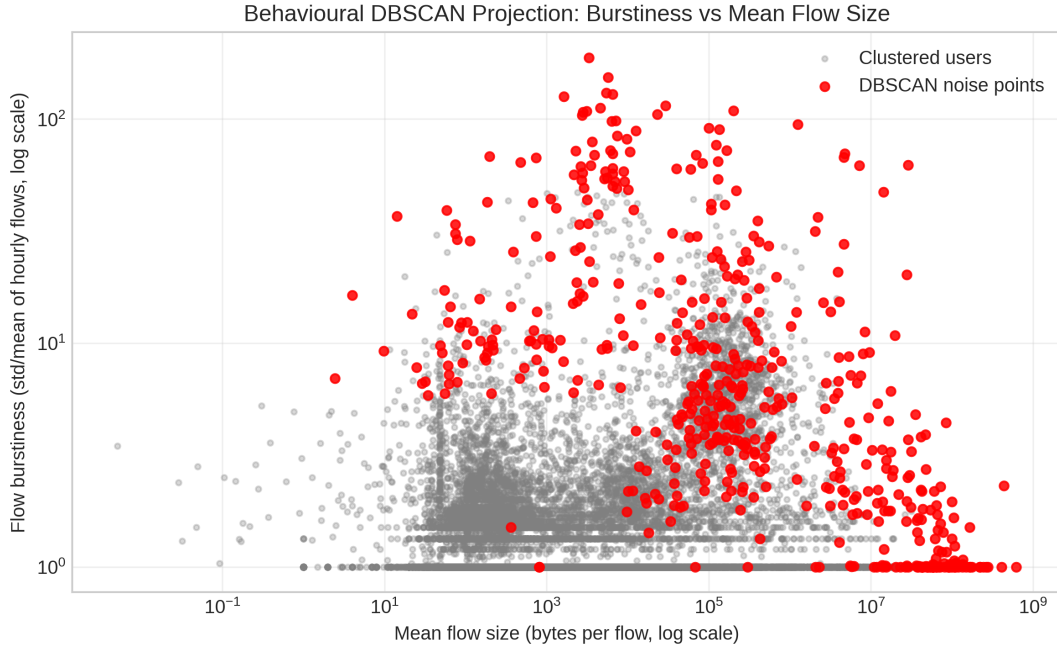


Figure 5.3: DBSCAN in behavioural space: burstiness versus mean flow size, with noise points highlighted.

5.2.2 PCA-supported DBSCAN

Principal Component Analysis (PCA) is employed as a representation-compression stage before density-based clustering. The objective is to reduce redundancy among correlated behavioural variables and to mitigate the effect of noisy high-dimensional directions on

neighborhood structure.

The procedure is applied to the behavioural matrix of the whole population and 15 descriptors (volume, variability, temporal ratios, directional ratios, flow-size and concentration indicators). Heavy-tailed magnitude variables are transformed through $\log_{10}(x+1)$ after non-negativity clipping, then standardized. PCA is fitted on the resulting matrix, and the retained dimensionality is selected through cumulative explained variance.

The explained-variance profile is concentrated in early components: PC1 explains 48.77%, PC2 explains 13.33%, and six components explain approximately 95% cumulatively. This pattern indicates that the population is largely organized along a compact latent manifold, with most behavioural variability captured by two components.

This result has a direct methodological implication. PCA preserves variance, not anomaly contrast; consequently, sparse anomalous profiles may remain close to the dominant manifold when they do not define independent high-variance directions. Dimensionality reduction therefore improves compactness and interpretability of the geometry, but does not imply a better separability of anomalies.

DBSCAN is subsequently applied in PCA space on a 60,000-user sample, due to its computational complexity limitations. The procedure follows the same density-selection logic adopted in previous experiments, with `min samples=15` and `eps` guided by k-distance inspection and quantile sweep.

The PCA-space k-distance profile reports `min = 0.0`, `median = 0.0`, and `max  $\approx 30.28$` , confirming a large compact core and a sparse tail. Candidate `eps` values between upper quantiles produce the expected progression: at low `eps` the clustering is highly fragmented (62 clusters, 11,298 noise points at `eps=0.1083`), while increasing `eps` progressively merges clusters and reduces noise (2 clusters, 782 noise points at `eps=0.6877`). The final configuration is set to `eps=0.6877`.

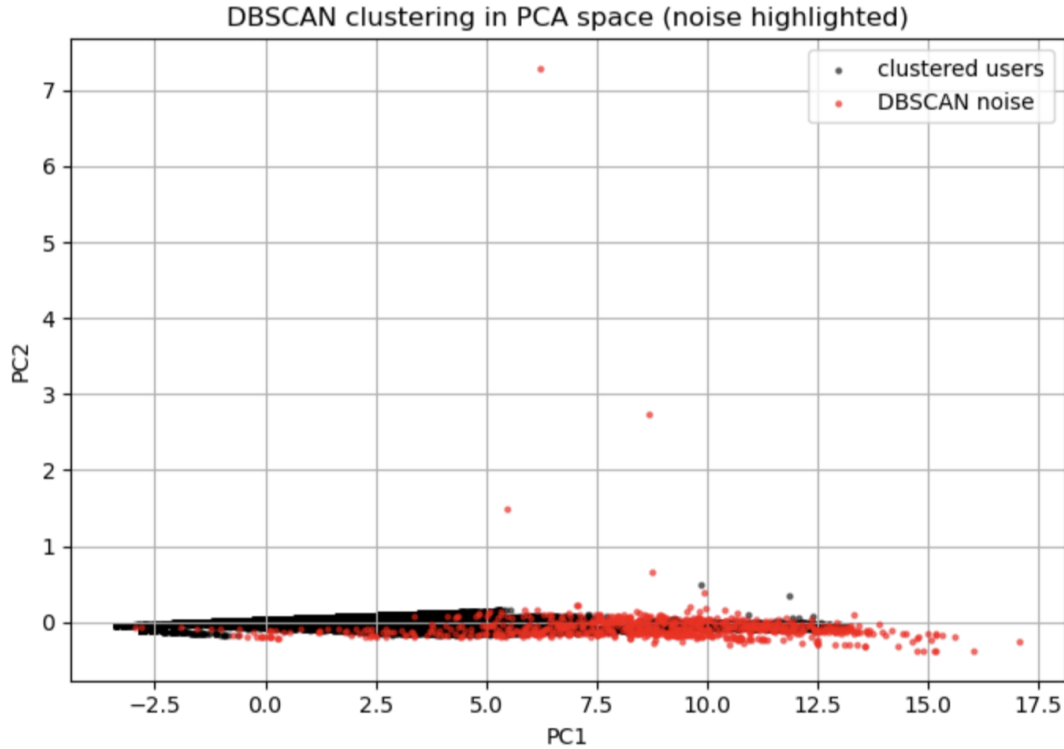


Figure 5.4: DBSCAN on PCA components

Under this setting, the final partition includes one dominant cluster (58,212 users), one secondary cluster (1,006 users), and 782 noise points (1.30%). The magnitude of the main cluster confirms that most users remain concentrated in a common latent regime even after projection and density filtering.

Figure 5.4 provides the 2D PC1–PC2 visualization with noise highlighting. Two aspects are relevant for interpretation. First, most noise points are not isolated far from the dataset support; they are mainly located at low-density margins of the dominant cloud. Second, only a very small subset appears at clearly extreme coordinates (e.g., high PC2 outliers), representing stronger anomaly candidates.

Also, the same figure clearly shows that the first component dominates the percentage of the whole explained-variance.

This behavior is coherent with the full pipeline evidence: PCA improves geometric compactness and visual interpretability, but anomaly structure remains mostly boundary-driven rather than cluster-separated. Consequently, PCA-supported DBSCAN contributes robust structural diagnostics, while full anomaly prioritization requires complementary scoring models.

5.2.3 Clustering results

Taken jointly, the clustering experiments on volume space, behavioural space, and PCA-reduced space converge to a consistent conclusion. The dataset is dominated by a compact high-density user manifold, while detected anomalies mostly appear as sparse boundary points or small residual groups. In this setting, clustering remains effective as a structural exploration tool: it supports user segmentation, provides interpretable geometric organization of the population, and helps prioritize local outlier inspection within and across clusters.

At the same time, the results highlight clear limitations. First, the methodology is operationally applied to controlled subsets (e.g., 60k samples) rather than directly to the full population, due to density-computation constraints and stability considerations. A second issue is that, even with hand-crafted features and dimensionality reduction in place, clustering still fails to yield consistent separation of intricate anomalous behaviours once the analysis is pushed to a global scale. In most cases, accounts flagged as suspicious sit close to the low-density edges of the dominant manifold, which implies that deviation is typically faint and interwoven with ordinary behaviour. For these reasons, clustering is kept in the pipeline primarily for exploration and diagnosis, not as a self-sufficient detection approach for the entire user base. Its contribution is mainly to surface plausible outliers and to describe cluster-level profiles, whereas the identification of higher-order, more involved behavioural structure is assigned to complementary reconstruction-based models introduced in the sections that follow.

5.3 Deep-learning anomaly modeling

The clustering stage established a robust geometric baseline, but also exposed its structural limits, greater reason in big datasets. Most users are concentrated in a dominant manifold, and anomalous candidates are frequently located at low-density boundaries rather than in clearly separated clusters. As a result, density-based methods provide useful insights, yet they show limited sensitivity to subtle, high-order behavioural irregularities.

This motivates the transition to deep representation learning. Unlike clustering in fixed feature projections, deep models can learn non-linear latent structure and capture inter-

actions among temporal, directional, and service-related variables. In anomaly-detection settings, this capability is particularly relevant when abnormality is not expressed as a distinct cluster, but as a complex deviation from normal behavioural regularity.

In this chapter, deep learning is introduced through an autoencoder framework. The model is trained to reconstruct typical user signatures; reconstruction error is then used as a continuous anomaly signal. This approach complements clustering by improving discrimination of weak but systematic deviations that remain entangled with normal users in density space. Consistently with the chapter structure, the theoretical and architectural setup is introduced first, and reconstruction/latent-space evidence is analyzed afterward.

5.3.1 Deep-learning theoretical background

Deep learning is a class of machine-learning methods based on multi-layer neural networks that learn hierarchical representations directly from data. In contrast with shallow models relying on manually designed transformations, deep architectures can model non-linear dependencies across many variables and can progressively extract latent features at different abstraction levels.

From a methodological perspective, the relevance of deep learning in anomaly detection lies in representation capacity. Network-traffic data are typically high-dimensional, heterogeneous, and weakly separable under linear projections. Under these conditions, abnormality may emerge as a complex configuration of temporal, directional, and service-related relations, rather than as a simple threshold violation on single features. Deep models are suited to this setting because they can encode interactions that are difficult to capture with purely distance-based clustering.

In the literature, deep learning is applied to anomaly detection through multiple paradigms. Supervised approaches are effective when labeled attacks are available, but they are often limited by class imbalance and poor generalization to unseen attack families. Semi-supervised and self-supervised approaches are therefore common in operational security settings, where labels are partial or absent. In these frameworks, models are trained to learn the structure of normal behaviour and anomalies are identified as deviations from that learned structure.

Among unsupervised deep methods, autoencoders are widely adopted due to their interpretability and practical robustness. They provide a compact latent representation of input behaviour and quantify abnormality via reconstruction error. This property is especially useful when anomalies are rare, diverse, and not naturally grouped into explicit clusters.

For these reasons, the following sections focus on an autoencoder-based pipeline: first the model setup and training strategy are defined, then reconstruction-error patterns are analyzed, and finally the latent representation is integrated with clustering signals for internal anomaly ranking.

At a high level, an autoencoder is a neural network architecture that learns to reproduce its own input after passing through a compressed internal representation. The key idea is that, to reconstruct data accurately from a restricted latent space, the model must capture the regular structure of the observed behaviour and discard irrelevant variability.

This mechanism is particularly useful in anomaly detection for network traffic. Normal user profiles tend to follow recurrent statistical patterns and are therefore reconstructed with limited error, while atypical profiles are harder to reconstruct and produce larger residuals. Also, This concept presumes that most traffic is normal. For this reason, autoencoders are widely used in the literature as unsupervised detectors when explicit attack labels are incomplete, noisy, or unavailable.

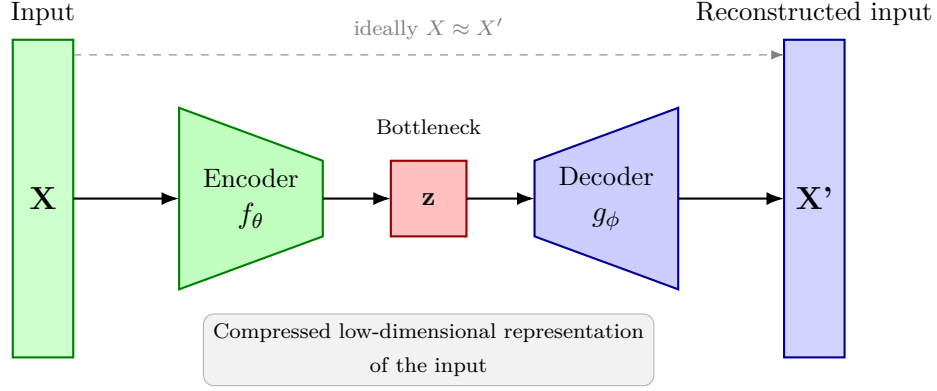


Figure 5.5: Conceptual autoencoder schema: input encoding, latent bottleneck, and reconstruction output.

After this conceptual view, Figure 5.5 summarizes the computational structure of an autoencoder. Let $\mathbf{x} \in \mathbb{R}^d$ be an input user signature. The encoder learns a parametric mapping

$$\mathbf{z} = f_{\theta}(\mathbf{x}), \quad \mathbf{z} \in \mathbb{R}^m, \quad m \ll d,$$

where $\mathbf{z}$ is a compressed latent representation. The decoder then maps latent codes back to input space,

$$\hat{\mathbf{x}} = g_{\phi}(\mathbf{z}) = g_{\phi}(f_{\theta}(\mathbf{x})),$$

producing a reconstruction $\hat{\mathbf{x}}$ of the original profile.

Training is performed by minimizing a reconstruction objective over the dataset $\{\mathbf{x}_i\}_{i=1}^N$. Two standard choices are mean squared error (MSE),

$$\mathcal{L}_{\text{MSE}}(\theta, \phi) = \frac{1}{N} \sum_{i=1}^N \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2,$$

and mean absolute error (MAE),

$$\mathcal{L}_{\text{MAE}}(\theta, \phi) = \frac{1}{N} \sum_{i=1}^N \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_1.$$

Since MSE squares the error, it emphasizes large reconstruction errors and is therefore more sensitive to high-magnitude deviations, which can improve anomaly contrast but also increases sensitivity to outliers and heavy tails. MAE is more robust to extreme values and often yields stabler optimization in noisy settings, but it can reduce emphasis on rare high-impact deviations.

In anomaly-detection settings, this mechanism has a direct operational interpretation. When the model is trained predominantly on normal behavior, common patterns are reconstructed with low error, whereas atypical profiles are reconstructed poorly and yield larger residuals. A per-user anomaly score can therefore be defined as

$$s_i = \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2,$$

or through feature-weighted variants of the same principle. The score is continuous and rankable, which enables fine-grained prioritization rather than binary decisions only and it is useful in a complex world like cybersecurity.

The bottleneck dimension and regularization strategy control the trade-off between fidelity and abstraction. If the bottleneck is too wide, the model may memorize noise and reduce anomaly contrast; if it is too narrow, relevant normal variability may be lost, increasing false positives. For this reason, architecture depth, latent size, dropout, and early stopping are treated as critical design parameters in the experimental setup.

5.3.2 Autoencoder setup

The autoencoder setup is designed to maximize reconstruction fidelity on regular behaviour while preserving sensitivity to structurally atypical user profiles. The implementation follows three coordinated design blocks: input representation, model architecture, and optimization protocol.

Input representation and preprocessing. The model input is the user-signature matrix described in Chapter 4, with 670,134 users and 130 features. Each vector combines hourly behavioural descriptors and service-profile shares (Top-50 services), so that both temporal organization and application preference are represented in the same space.

Before training, a dedicated preprocessing pipeline is applied to improve numerical stability: invalid values are sanitized, heavy-tailed magnitudes are compressed with `log1p`, robust scaling is performed through median normalization, and extreme values are clipped to a bounded range. This sequence is intended to avoid domination by rare large-scale values and to ensure stable gradient dynamics during optimization.

Network architecture. The autoencoder uses a symmetric fully connected topology:

$$130 \rightarrow 128 \rightarrow 64 \rightarrow 16 \rightarrow 64 \rightarrow 128 \rightarrow 130,$$

with latent bottleneck dimension $m = 16$. The symmetric design is adopted to keep encoder/decoder capacity balanced: since the task is reconstruction rather than classification, a mirrored decoder provides a controlled inverse mapping of the learned compression path and improves optimization stability.

The progressive reduction $130 \rightarrow 128 \rightarrow 64 \rightarrow 16$ is chosen to avoid abrupt information collapse. A shallow reduction could preserve too much redundant detail (weak anomaly contrast), while an excessively aggressive compression could remove relevant normal variability (higher false positives). The selected bottleneck dimension enforces abstraction but still preserves sufficient descriptive power for regular traffic patterns.

To stabilize learning under heterogeneous and heavy-tailed feature distributions, batch normalization and dropout are placed in both network branches. Dropout is fixed at $p = 0.2$, chosen as a middle setting, this means that only 20% of neurons are deactivated. At this level, it limits co-adaptation and reduces memorization of common patterns, yet it does not distort the reconstruction target to the point where the model underfits, a risk observed when dropout is stronger. In validation, this choice yielded a steady convergence profile, so the same rate was kept in the final experiment.

Training protocol. Training is performed with mini-batch optimization using Adam (`lr` = 10^{-3} , `weight_decay` = 10^{-5}) and MSE reconstruction loss. A reproducible user-level split is adopted with 90% training and 10% validation, resulting in 603,121 training samples and 67,013 validation samples.

Early stopping is used for model selection (`patience` = 5, `min_delta` = 10^{-3}): the best checkpoint is selected by minimum validation loss. In the executed run, training stops

at epoch 37 with best validation MSE ≈ 0.1376 , indicating stable convergence without late-epoch overfitting.

Expected outcome of the setup. This configuration is expected to learn a compact latent manifold of regular user behaviour and to produce low reconstruction error for frequent profiles. Conversely, users whose temporal-service structure is weakly represented in the learned manifold should yield higher reconstruction residuals. After training, the model outputs both latent embeddings $Z \in \mathbb{R}^{670134 \times 16}$ and per-user reconstruction errors, which constitute the primary deep anomaly signal for subsequent analysis.

5.3.3 Autoencoder results: reconstruction-error evidence

After defining the methodological setup, the analysis now turns to empirical evidence produced by the trained model. Results are presented in two complementary forms: reconstruction-error diagnostics and latent-space structural checks.

After training, the autoencoder is evaluated in inference mode on the full normalized dataset. For each user, two outputs are extracted: the latent embedding $\mathbf{z}_i \in \mathbb{R}^{16}$ and the reconstruction residual. The operational anomaly signal is defined as per-user reconstruction MSE,

$$r_i = \frac{1}{d} \sum_{j=1}^d (x_{ij} - \hat{x}_{ij})^2,$$

with $d = 130$ input dimensions.

Methodological role of reconstruction residuals. Reconstruction residuals provide a direct measure of mismatch with the normal structure learned by the model. Profiles consistent with dominant behavioural regularities are reconstructed with low error, whereas atypical profiles concentrate larger residuals on specific temporal or service components. Unlike binary cluster membership, this signal is continuous and therefore supports fine-grained prioritization.

Operational use in the proposed pipeline. The residual distribution is used in two complementary ways. First, users are ranked by r_i to identify the most suspicious profiles. Second, binary flagging is obtained through a geometric elbow criterion on the sorted reconstruction-error curve, rather than by a fixed quantile cut.

Let $\{r_{(i)}\}_{i=1}^N$ be reconstruction errors sorted in descending order. After min-max normalization of index and value axes, the threshold point is selected as the sample maximizing orthogonal distance from the chord connecting the first and last points of the curve. Denoting this index by i^* , the decision threshold is set as

$$\tau_{\text{AE}} = r_{(i^*)}, \quad \text{ae_is_anomaly} = \mathbb{I}\{r_i \geq \tau_{\text{AE}}\}.$$

In the executed run, $\tau_{\text{AE}} \approx 1.0543$, yielding 19,197 flagged users.

To integrate the deep model into the global framework, the reconstruction score is converted to a percentile rank (`ae_recon_pct`) and added to the internal anomaly score with dedicated weight ($w_{\text{AE}} = 3.0$). This design preserves comparability with other detectors while assigning stronger influence to reconstruction evidence, which is expected to capture higher-order behavioural deviations not isolated by density methods alone.

Global residual profile. The extracted tensors confirm computational consistency ($Z \in \mathbb{R}^{670134 \times 16}$, $r \in \mathbb{R}^{670134}$). The sorted reconstruction-error curve (logarithmic scale) exhibits a marked right-skewed structure, characterized by a compact body of low-error users and a sparse high-error tail. This geometry supports a rank-based interpretation of anomaly evidence: the majority of users is reconstructed within a stable regime, while a minority concentrates disproportionate residual mass.

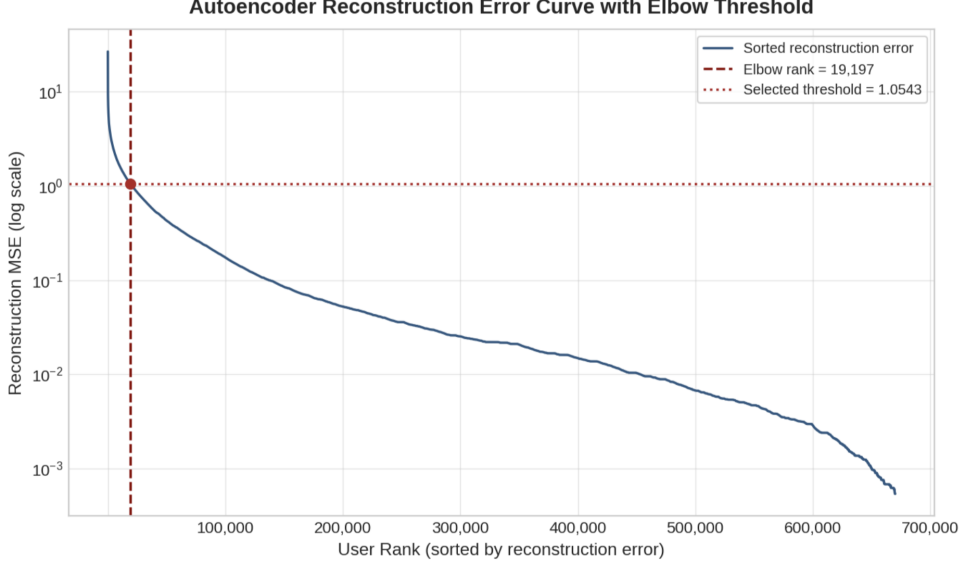


Figure 5.6: Sorted autoencoder reconstruction MSE (log scale) with elbow-based threshold position.

Figure 5.6 highlights the transition region between the high-error tail and the dense low-error body. The elbow criterion provides a data-driven operating point that adapts to the observed residual geometry and avoids dependence on an arbitrarily fixed percentile.

Top-tail selection and feature decomposition. Applying the elbow-based threshold on reconstruction MSE identifies 19,197 users. This subset is characterized through feature-level decomposition of squared residuals, averaged over anomalous users. The dominant components are reported in Table 5.2. The largest contributions are concentrated on hourly traffic descriptors, with recurrent activation of `mean_total_bytes` and `mean_bytes_up` in specific slots, particularly early-morning and selected evening/night intervals.

Distributional separation on representative features. For leading components (e.g., `(mean_total_bytes, 7)`), kernel-density diagnostics show a clear shift between normal and anomalous populations: normal users remain concentrated near low residual values, whereas anomalous users present broader distributions with heavier right tails. This pattern is consistent with localized reconstruction breakdown on specific temporal dimensions, rather than uniform error inflation across all variables.

Hourly error dynamics. The residual signal is further aggregated by hour and by macro-feature family (`mean_total_bytes`, `mean_bytes_up`, `mean_bytes_down`). Three robust patterns emerge: (i) persistent separation between anomalous and normal curves

Feature (hourly component)	Mean reconstruction error
(mean_total_bytes, 7)	8.791
(mean_bytes_up, 7)	8.056
(mean_total_bytes, 8)	7.395
(mean_total_bytes, 11)	7.242
(mean_total_bytes, 12)	6.989
(mean_total_bytes, 15)	6.875
(mean_total_bytes, 10)	6.731
(mean_total_bytes, 4)	6.720
(mean_total_bytes, 22)	6.714
(mean_total_bytes, 17)	6.710

Table 5.2: Top reconstruction-error features in the anomalous user set selected by elbow threshold.

across the full day, (ii) pronounced amplification around 07:00–08:00 in total/uplink components, and (iii) distinct night/late-evening structure in downlink residuals. These observations indicate that the anomaly signal is temporally organized and not reducible to pure volume effects.

Sanity check on raw hourly aggregates. An additional validation step compares raw hourly profiles of top-anomaly users and baseline users in terms of total volume, uplink share, and flow counts. At this aggregation level, separation is weaker and often not linearly interpretable. This negative evidence is methodologically relevant: it rules out trivial first-order explanations (“higher volume only”, “simple day/night inversion”, “persistent upload dominance”) and indicates sensitivity to higher-order structure, including burst scheduling, periodic irregularity, and subtle temporal synchronization.

Implication for the anomaly framework. Overall, these diagnostics indicate that reconstruction error captures behavioural mismatch beyond simple magnitude effects. The resulting signal remains continuous, rankable, and feature-attributable, making it suitable both for candidate prioritization and for interpretability-oriented follow-up analysis. On this basis, reconstruction residuals are retained as the primary deep-learning contribution in the integrated internal anomaly score.

5.3.4 Latent-space clustering

After extracting latent embeddings $Z \in \mathbb{R}^{670134 \times 16}$ and per-user reconstruction errors, a second-stage density analysis is performed directly in latent space. The goal is to verify whether the users with highest reconstruction mismatch form separable anomalous substructures.

Protocol. Users are first ranked by reconstruction MSE and the top $N = 60,000$ are selected:

$$\mathcal{U}_{\text{top}} = \text{TopN}(r_i, N), \quad N = 60,000.$$

DBSCAN is subsequently applied to Z_{top} , namely to the latent vectors associated with the high-risk subset. Hyperparameter selection follows a structured calibration procedure consistent with previous sections. First, a latent-space k -distance profile is computed using a 10-nearest-neighbor criterion. Then, **eps** is explored over an upper-tail interval of the

distance distribution, between quantiles $q_{0.90}$ and $q_{0.995}$, and each candidate configuration is evaluated through cluster count and noise proportion.

Selection criterion for `min_samples`. The choice of `min_samples` is based on sensitivity analysis rather than heuristic fixing. Multiple candidate values are tested, spanning low, intermediate, and high density constraints, and the resulting partitions are assessed according to three criteria: (i) proliferation of micro-clusters, (ii) relative amount of noise points, and (iii) structural stability under local variations of `eps`. Low values systematically lead to over-fragmented and weakly stable solutions, while excessively high values absorb sparse but meaningful behaviours into the dominant component.

Within this trade-off, `min_samples` = 15 is retained as the most balanced operating point. The selected value provides sufficient regularization against spurious micro-groups while preserving an interpretable sparse-tail signal. In addition, this setting remains aligned with the PCA-DBSCAN configuration adopted earlier, thus improving methodological consistency across experiments.

Epsilon sweep evidence. After fixing `min_samples` = 15, the `eps` sweep reported in Table 5.3 is used to identify the final operating region. The observed progression is coherent with density-based clustering behavior: increasing `eps` reduces fragmentation by merging adjacent dense regions and decreases the number of points labeled as noise.

<code>eps</code>	Clusters	Noise points
6.8199	4	3,505
7.6557	2	2,283
8.4915	2	1,490
9.3273	1	934
10.1631	2	533
10.9989	1	289
11.8347	1	143
12.6704	1	72

Table 5.3: DBSCAN `eps` sweep in autoencoder latent space on top-60k reconstruction-error users.

Final latent-space DBSCAN result. Using `eps` = 11.8347 with `min_samples` = 15 in the final run, DBSCAN yields one dominant cluster (59,857 users) and 143 noise points (0.24%). The corresponding PCA projection of latent vectors confirms that high-reconstruction users remain mostly concentrated in a compact support, with only a small sparse fringe.

This is an important finding: the autoencoder reconstruction score provides strong ranking contrast, but the associated latent geometry among top-ranked users is still largely continuous rather than split into many detached dense groups.

Figure 5.7 provides a complementary view of this behavior. The top-error users are not concentrated in a single isolated lobe; rather, they are distributed across multiple low-density regions of the global manifold. This supports the interpretation that anomalousness is mainly expressed as *deviation intensity from learned regularity*, and not necessarily as a compact standalone cluster.

From an operational perspective, the latent projection highlights two relevant aspects. First, high-error users appear more frequently near geometric margins and transition zones, where representation uncertainty is expected to be higher. Second, a non-negligible subset

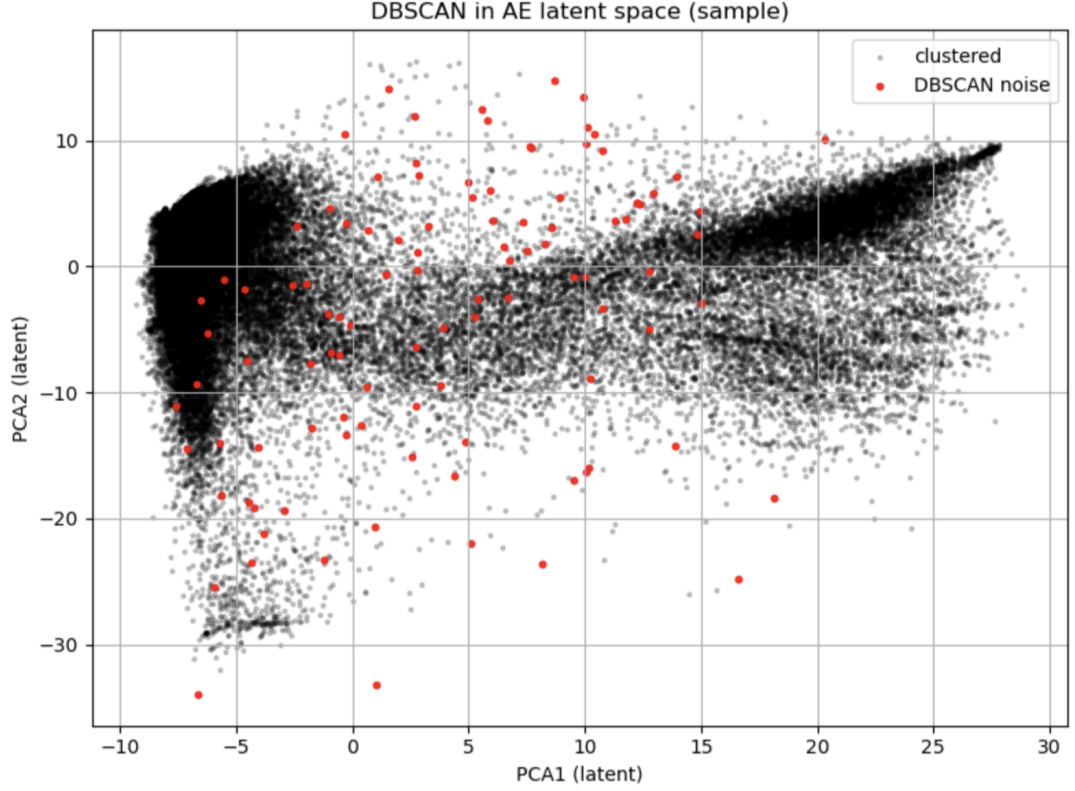


Figure 5.7: Autoencoder latent space (PCA projection): full user cloud in blue and top-500 reconstruction-error users in orange.

overlaps with the central support, indicating that elevated reconstruction error can also arise from subtle feature interactions that remain weakly visible in 2D projections. Consequently, latent-space visualization is informative for structural diagnostics, but it is not sufficient as a standalone decision mechanism.

For this reason, reconstruction ranking is retained as the primary deep-learning signal, while latent-space clustering is used as an auxiliary stratification layer to distinguish sparse outliers from structured high-risk groups.

Cross-check on the same top-user subset. For consistency, DBSCAN is also applied on the same top-60k users in original engineered spaces:

- *volume space*: 26 clusters, 985 noise points (1.64%);
- *behavioural space*: 18 clusters, 1,002 noise points (1.67%).

Compared with latent-space DBSCAN (143 noise points), these projections are more fragmented and produce larger boundary-noise sets. Operationally, this supports the adopted strategy: latent-space clustering is retained as an auxiliary structural check, while anomaly prioritization relies primarily on reconstruction ranking and multi-signal integration in the next section.

5.4 Internal anomaly score integration

The previous sections provide complementary anomaly signals with different inductive biases. To obtain a single operational ranking, these signals are merged into a unified internal score at user level. The section first defines design rationale and aggregation rule, and then evaluates the resulting ranking behavior.

The introduction of an internal anomaly score responds to a structural requirement of the entire pipeline: transforming heterogeneous unsupervised outputs into a coherent decision-support signal. In this thesis, no detector provides a complete and self-sufficient notion of abnormality. Density-based methods emphasize geometric sparsity in engineered feature spaces; reconstruction-based methods emphasize mismatch with learned latent regularity. Each perspective captures a different facet of behavioral deviation, but each also carries method-specific biases. A unifying score is therefore required to preserve complementary evidence while reducing detector-specific arbitrariness.

From a methodological perspective, this integration is motivated by three limitations that emerge when detectors are used independently. First, binary outlier labels from clustering are often sensitive to local density choices and can fluctuate near boundary regions. Second, continuous reconstruction error provides strong ranking power but may not directly encode cluster-level structure or subgroup typology. Third, in absence of ground-truth attack labels, model selection cannot rely on conventional supervised metrics and must instead privilege consistency, plausibility, and stability across independent signals. The internal score addresses these constraints by aggregating partially redundant, partially orthogonal indicators into a common ranking space.

The design objective is not to “average” detectors indiscriminately, but to formalize a hierarchy of evidence. Signals that represent higher-order behavioral mismatch are assigned greater influence, while signals more exposed to geometric boundary artifacts are retained with lower weight. This strategy enables two simultaneous effects: it preserves weak but meaningful support from clustering diagnostics and, at the same time, prevents local density fluctuations from dominating the final prioritization. The resulting score can therefore be interpreted as an evidence accumulator, where cumulative agreement strengthens anomaly confidence.

An additional motivation concerns operational interpretability. In applied anomaly detection, downstream analysis requires more than a binary decision: analysts need a ranked set of candidates, a decomposition of why each user is prioritized, and a mechanism to trace contributions of different detection families. The internal score fulfills this requirement because each component remains explicit (autoencoder vote, volume-space vote, behavioural-space vote, continuous percentile signal), allowing transparent post-hoc interpretation of ranking positions. This property is especially relevant in high-volume monitoring contexts, where explainability directly affects triage efficiency.

Finally, the score is conceived as an internal consistency layer rather than as a surrogate ground truth. It does not claim probabilistic calibration of attack likelihood; instead, it provides a principled ordering of users under uncertainty, aligned with the thesis objective of candidate prioritization for deeper behavioral and pattern-level investigation. In this sense, the anomaly score represents a bridge between exploratory unsupervised modeling and actionable forensic follow-up: it converts dispersed model outputs into a stable, comparable, and inspection-oriented prioritization criterion.

Input signals. The integration stage uses three binary indicators derived from the final anomaly-oriented analyses:

- **dbscan_vol_vote**: user labeled as noise in volume-space DBSCAN over AE-prioritized users;
- **dbscan_behav_vote**: user labeled as noise in behavioural-space DBSCAN over AE-prioritized users;
- **ae_vote**: user above the elbow-based reconstruction threshold in the autoencoder output.

In parallel, a continuous deep-learning signal is also retained:

$$\text{ae_recon_pct} = \text{rank}_{\text{pct}}(\text{ae_recon_mse}),$$

which contributes to an auxiliary continuous score (**anomaly_score**) with weight $w_{\text{AE}} = 3.0$. This preserves intra-class ordering among high-risk users even when binary votes are identical.

The final integrated score used for coarse prioritization is a weighted sum of binary votes:

$$\text{final_score} = 1.0 \cdot \text{dbscan_vol_vote} + 1.5 \cdot \text{dbscan_behav_vote} + 3.0 \cdot \text{ae_vote}.$$

This weighting assigns the largest influence to autoencoder evidence, while preserving explicit contribution from density-based outlier labels in engineered spaces.

The selected coefficients reflect the empirical behavior observed in the notebook:

- volume-space noise labels are informative but prone to boundary effects, thus assigned baseline weight;
- behavioural-space noise labels are more semantically discriminative than pure volume, thus moderately up-weighted;
- AE threshold crossing captures higher-order structural mismatch and is therefore treated as the dominant anomaly signal.

Operational output. The integration stage produces a single per-user ranking table (**final_ranking**) including component votes and aggregate score. This object is used as entry point for subsequent profile-level analysis, template matching, and candidate extraction in the next chapter.

Table 5.4 reports a compact excerpt of high-priority users from the integrated ranking.

User	A-Score	AE MSE	AE Pct	DBSCAN-V	DBSCAN-B	Final
U1	2.992	3.810	0.997	1	1	5.5
U2	2.979	2.515	0.993	1	1	5.5
U3	2.973	2.175	0.991	1	1	5.5
U4	2.980	2.565	0.993	1	1	5.5
U5	2.961	1.771	0.987	1	1	5.5
U6	2.990	3.484	0.997	0	1	4.5
U7	2.951	1.548	0.984	0	1	4.5

Table 5.4: Excerpt of high-priority users from the internal anomaly-score ranking (values rounded; user identifiers pseudonymized).

For clarity, Table 5.4 uses compact column labels. **A-Score** denotes the continuous internal score (**anomaly_score**), obtained by adding weighted normalized reconstruction evidence to previously accumulated signals. **AE MSE** is the raw autoencoder reconstruction error (**ae_recon_mse**), while **AE Pct** is its percentile-normalized version

(`ae_recon_pct` $\in [0, 1]$). **DBSCAN-V** and **DBSCAN-B** are binary DBSCAN votes in volume and behavioural spaces, respectively (`dbscan_vol_vote`, `dbscan_behav_vote`). **Final** corresponds to the weighted discrete aggregation (`final_score`) used for coarse ranking.

5.4.1 Evaluation perspective

In this study, the integrated anomaly score is evaluated through a diagnostic perspective centered on three concrete questions: whether the score is coherent with detector evidence, whether it provides effective prioritization granularity, and whether top-ranked candidates remain behaviourally interpretable in downstream analysis.

First, evidence coherence is assessed by inspecting score composition. The highest ranks are associated with cumulative support from multiple components, as shown by the operational excerpt in Table 5.4. In particular, the maximum observed tier (`final_score` = 5.5) corresponds to simultaneous activation of autoencoder evidence (`ae_vote` = 1) and both density-based votes (`dbscan_vol_vote` = 1, `dbscan_behav_vote` = 1). Lower high-priority tiers (e.g., 4.5) correspond to partial agreement, where reconstruction evidence is preserved but one clustering signal is absent. This behavior is consistent with the intended design of the weighted aggregation rule.

Second, ranking granularity is evaluated directly on the observed score distribution. Because `final_score` is built from weighted binary votes, it is intrinsically discrete and can produce broad tie regions. This is not only a theoretical limitation: in the current notebook run, an exploratory quantile cut defined as `thr_score` = `quantile0.99(final_score)` (implemented after score construction) yields a degenerate selection (top set equal to the full population), demonstrating that percentile thresholding on a low-cardinality score can become non-informative. This empirical result is critical for interpretation: `final_score` is effective for coarse evidence stratification, but insufficient as a standalone selector when score mass concentrates on few values.

For this reason, practical prioritization is treated as a two-level process. The integrated discrete score is used to define coarse evidence tiers, while intra-tier ordering is delegated to continuous signals (`ae_recon_mse`, `ae_recon_pct`, `anomaly_score`) or fixed-cardinality selection (k -best), depending on operational needs. Under this interpretation, the score is not a calibrated probability of attack, but an internal evidence index for triage.

Third, external plausibility is evaluated in the following chapter through taxonomy-driven characterization of high-priority users into four behavioural attack classes. This step functions as post-hoc validation-by-structure: if top-ranked anomalies map to coherent, recurrent behavioural patterns, confidence increases that the integrated score captures meaningful abnormal phenomena rather than random fluctuations. Methodologically, this remains weakly supervised validation and must not be interpreted as ground-truth performance estimation.

Chapter 6

Attack pattern recognition

This chapter extends the post-detection analysis stage by converting anomaly-ranking results into behavioral classes that admit direct operational interpretation. The aim is not to assert supervised attribution of attacks, but to examine whether the users prioritized in Chapter 5 show coherent, recurring traffic patterns that can be grouped into stable, engineering-facing categories.

A stepwise workflow structures the chapter. To improve readability without rewriting validated content, each block follows a fixed order: methodological definition first, empirical evidence and interpretation second. Anomalous candidates are first assigned to behavior-centered classes extracted from the internal anomaly score, then the taxonomy evidence is analyzed. The chapter then introduces the residential-proxy identification methodology and finally examines raw-traffic results through an engineering interpretation layer.

6.1 Anomaly-Driven Behavioral Classification: methodology and results

This section introduces the first post-ranking characterization stage. After anomaly prioritization in Chapter 5, the objective is to move from a pure “how anomalous” ordering to a “what type of anomalous behavior” interpretation. The classification step is therefore designed to improve operational meaning of the ranking and to prepare downstream investigation. The section first defines the class-construction protocol and then discusses class distribution and profile separation results.

The taxonomy is intentionally restricted to four classes. This design choice is methodological: with weak supervision and no packet-level labels, a compact set of behaviorally distinctive classes provides higher interpretability and lower overfitting risk than a fine-grained taxonomy with unstable boundaries. The selected classes are derived from recurrent temporal motifs observed in anomalous users and encoded as template behaviors over hourly signatures.

Concretely, four class templates are defined on hourly profiles of mean flows, mean uplink bytes, mean downlink bytes, and mean total transmitted bytes. Per-user signatures and templates are L1-normalized and compared through cosine similarity. Each user is assigned to the template with maximum similarity; assignment confidence is controlled through a similarity-threshold and margin rule:

$$\mathcal{M}_i = 1 \iff (s_i \geq 0.60) \wedge (\Delta_i \geq 0.05),$$

where s_i is the best template similarity and Δ_i is the difference between best and second-best similarity. This criterion reduces ambiguous assignments and enforces separation between competing classes.

The four classes and their operational interpretation are summarized in Table 6.1. The table also reports the observed class distribution on top-score anomalous users extracted from the integrated ranking ($N = 19,197$).

Class	Behavioral target	Count	Share
<code>sustained_high_activity</code>	Persistent high-intensity activity across most hours; candidate always-on automated roles.	13,963	72.7%
<code>nocturnal_uplink_dominant</code>	Night-time upload-dominant behavior; candidate off-hours outbound automation.	3,291	17.1%
<code>fixed_time_volume_concentration</code>	Strong concentration of traffic in narrow fixed-hour windows; candidate scheduled batch jobs.	1,856	9.7%
<code>nocturnal_spike_activity</code>	Sparse but sharp night spikes; candidate intermittent event-driven processes.	87	0.5%

Table 6.1: Four-class anomaly taxonomy and observed distribution on top-score users.

The resulting distribution is strongly imbalanced, with dominance of `sustained_high_activity`. From an engineering standpoint, this does not invalidate the taxonomy; rather, it indicates that the anomaly ranking is primarily capturing continuously active automated behaviors, while sharper temporal profiles are less frequent but still detectable.

Figure 6.1 visualizes the class-level separation in hourly space across complementary metrics (flows, uplink volume, traffic concentration, peakiness). The key result is that classes remain distinguishable through temporal organization patterns, not only through absolute traffic magnitude. This supports the use of the taxonomy as a structured interpretation layer above anomaly scoring.

The four panels in Figure 6.1 allow a class-by-class engineering interpretation.

The class `fixed_time_volume_concentration` is characterized by narrow temporal concentration windows and high peakiness relative to daily baseline. This profile is compatible with users executing strongly scheduled tasks (e.g., periodic backup/export jobs, timed synchronization, or orchestrated maintenance routines). In adversarial settings, the same structure may also hide batch exfiltration windows or time-triggered command execution designed to minimize visibility during business hours.

The class `nocturnal_spike_activity` exhibits sparse but intense nocturnal bursts, with high local flow peaks and low daytime persistence. This behavior may correspond to benign event-driven automation with irregular activation, but it is also consistent with low-duty-cycle malicious patterns, such as periodic beaconing, burst scanning, or short-lived staging exchanges concentrated at night.

The class `nocturnal_uplink_dominant` shows a clear off-hours asymmetry in uplink traffic, combined with sustained but moderate flow activity. From an operational viewpoint, this pattern is relevant because persistent night-time outbound traffic can indicate automated data transfer pipelines, relay behavior, or controlled outbound channels. Legitimate explanations remain possible (e.g., remote replication, off-peak upload policies), but the directional asymmetry makes this class a high-priority candidate for deeper packet-level validation.



Figure 6.1: Hourly behavior profiles by anomaly class: flows, uplink intensity, traffic concentration, and peakiness.

The class `sustained_high_activity` is the dominant group and presents elevated activity across most hours, with comparatively flatter temporal structure. This profile can represent infrastructure-like or service-like users (always-on gateways, shared endpoints, heavy multi-tenant usage), but it may also include proxy-like relays or long-running automated systems. The key anomaly signal here is not a single temporal spike, but continuous operational intensity combined with deviation from ordinary user regularity.

A critical methodological implication is that these class labels should be interpreted as behavioral hypotheses rather than definitive attack labels. The taxonomy identifies recurring temporal archetypes that guide prioritization and investigation: classes with concentrated peaks support scheduled-operation hypotheses, nocturnal and uplink-asymmetric classes support outbound automation hypotheses, and sustained high-activity classes support relay/infrastructure hypotheses. This semantic stratification is precisely the added value of classification over raw anomaly ranking.

6.2 Residential Proxy Identification: methodology

Residential Proxies: Definition and Detection Relevance

A proxy is an intermediary network entity that forwards requests from a client toward an upstream service [11]. A *residential proxy* denotes a specific case in which forwarding is performed through addresses allocated to household access networks rather than datacenter ranges. This distinction is technically relevant because the apparent source inherits the

reputation and geolocation properties of consumer connectivity, reducing the effectiveness of controls based exclusively on origin class.

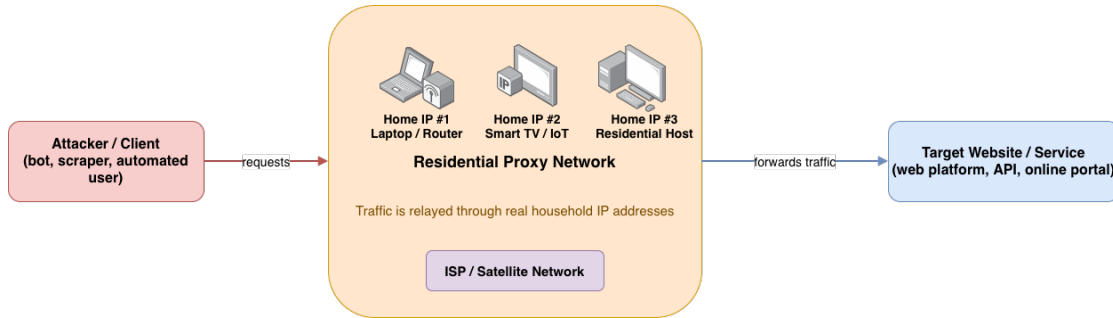


Figure 6.2: Conceptual architecture of a residential proxy workflow: client-originated automated traffic is relayed through residential endpoints before reaching the target service.

Figure 6.2 formalizes the mediation chain considered in this thesis. The controlling actor (left) does not interact directly with the target service (right); traffic is instead transited through a distributed residential layer (center), typically composed of heterogeneous household endpoints. The key analytical consequence is a structural decoupling between *traffic control* and *observed source identity*. Under this condition, source-based heuristics become weakly discriminative, and detection must rely on persistent behavioral signatures in time, directionality, and service usage.

Empirical studies show that residential-proxy ecosystems are heterogeneous and may include both legitimate commercial offerings and abuse-enabling infrastructures [23,24]. In both scenarios, attribution from source identity alone remains fragile, because operational control is displaced from the exposed origin.

From a cybersecurity perspective, residential proxies should not be interpreted as a single attack family, but as a high-risk *network role* that can support multiple malicious workflows, including anti-fraud evasion, high-scale automation, relay obfuscation, and staged outbound operations [8]. Consequently, the identification task is formulated here as role discovery under uncertainty, not as direct intent attribution from isolated indicators.

In passive monitoring settings, aggregate statistics rarely provide definitive proof of proxy usage. What can be established is convergent evidence: activity persistence, reduced short-term variability, directional asymmetry, service concentration, and ranking stability under perturbation. For this reason, residential proxy identification is treated as a hypothesis-driven process under weak supervision, followed by raw-data validation aimed at controlling false positives and strengthening interpretative confidence.

The residential-proxy analysis starts from the four-class behavioral taxonomy introduced in the previous section. The central hypothesis is that proxy-like entities should not appear as isolated volumetric outliers only, but as users with persistent, automation-compatible profiles: high activity continuity, low short-term variability, directional asymmetry (often uplink-heavy), and strong service concentration. Under this view, proxy detection is treated as a role-identification problem within the anomalous population, not as immediate attack attribution.

This hypothesis is motivated by the class distribution observed on top-score anomalies. Most high-priority users belong to [sustained_high_activity](#), while smaller but relevant fractions fall into [nocturnal_uplink_dominant](#) and [fixed_time_volume_concentration](#). These classes encode the exact temporal signatures expected from automated relay behavior: always-on operation, off-hours outbound transfer windows, or strongly scheduled

traffic bursts. Consequently, residential proxy candidates are searched primarily inside these behaviorally coherent subsets rather than in the full user base.

Operationally, the analysis follows a single sequential flow. First, a candidate-selection phase is executed on aggregated anomaly outputs, where high-score users are filtered through taxonomy coherence, robustness constraints, and explicit infrastructure disambiguation rules. This phase defines the residual proxy-plausible shortlist. Second, the shortlisted hosts are re-evaluated on raw records, where host behavior is reconstructed at finer temporal and service granularity to verify cross-scale consistency of the proxy hypothesis. The present section formalizes candidate-generation methodology; empirical raw-data outcomes are discussed in the following section.

An important interpretative constraint must be stated explicitly. The term “residential proxy candidate” denotes a behavioral hypothesis supported by traffic regularities and stability evidence; it does not constitute legal or forensic proof of malicious use. The objective is to provide a technically grounded shortlist for deeper analyst investigation, where classification confidence is progressively increased through multi-stage validation.

6.2.1 Candidate generation

The generation of residential-proxy candidates is designed as a progressive reduction process, with the objective of preserving behavioral plausibility while limiting spurious inclusions. The initial subset is extracted from the integrated anomaly ranking by applying a fixed score constraint. This criterion is preferred over a quantile-based cut because the integrated score assumes a discrete set of values and exhibits wide tie regions; under these conditions, thresholding on an absolute score provides a more stable operational boundary. The threshold value is also semantically grounded in the weighting scheme of Chapter 5: **final score** ≥ 4 requires at least simultaneous evidence from the autoencoder anomaly vote and one DBSCAN anomaly vote, thereby excluding single-detector signals and enforcing a minimum level of cross-method agreement. In the current run, this first selection step yields $N = 1,019$ users.

The selected set is then aligned with taxonomy outputs (dominant class and associated similarity) to enforce coherence with the behavioral model introduced in Section 6.1. After excluding deprecated exploratory aliases and retaining only users assigned to **sustained_high_activity**, the working set is reduced to 820 users. This restriction is methodologically motivated by the proxy hypothesis adopted in this chapter, which prioritizes persistent, automation-compatible activity profiles over episodic anomalous patterns.

For each user in the filtered set, a proxy-oriented descriptor vector is computed from hourly aggregates. The considered dimensions are temporal persistence (active hours), directional asymmetry (mean upload share), temporal regularity (coefficients of variation on flows and bytes), and service concentration (entropy and top-service share). These components are integrated into a composite proxy score used for ranking. The empirical distribution is markedly right-skewed (median = 0.330, 90th percentile = 0.900, 95th percentile = 0.950, 99th percentile = 0.990), indicating that strong proxy-like signatures are concentrated in a limited upper tail.

Ranking stability is then evaluated through perturbation analysis with 500 random weight configurations. For each configuration, top-30 membership is recomputed and aggregated into a robustness inclusion rate. The mean inclusion rate is 0.036, the median is 0, and the maximum observed value is 0.862. Agreement between manual and robustness-based top-30 selections remains limited, which confirms that ordering is reliable only for a narrow high-confidence prefix and less stable in the mid-rank region.

The stability curve in Figure 6.3 is used to define the transition between a stable

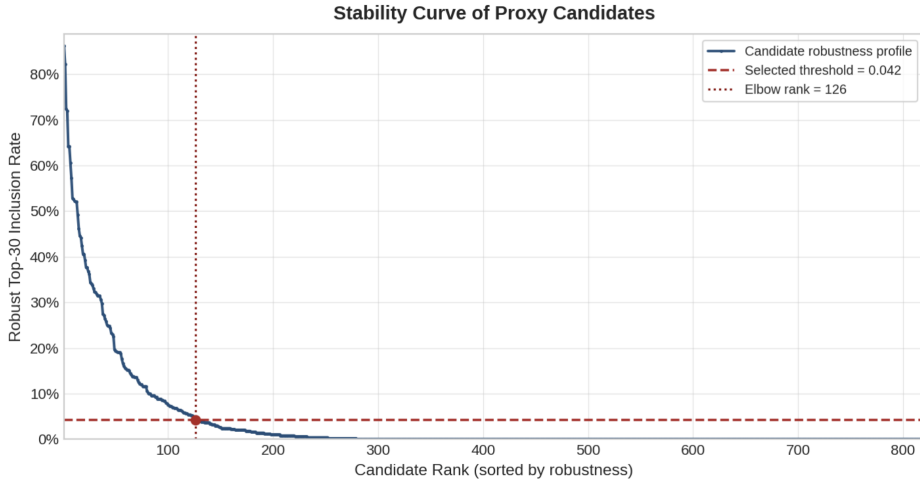


Figure 6.3: Stability curve of proxy candidates ranked by robustness inclusion rate.

prefix and a low-confidence tail. The elbow is located at rank index 125, corresponding to a robustness level of 0.042.

At this point, an explicit infrastructure-pruning rule is introduced to prevent conflation between residential-proxy hypotheses and structurally persistent infrastructure endpoints. In passive measurements, both groups may present high continuity, low variance, and strong service concentration, but their operational interpretation is different. Without a dedicated exclusion layer, infrastructure-like entities would dominate high ranks and reduce the specificity of proxy-oriented triage.

For this reason, the notebook applies a deterministic mask over the candidate pool using three joint constraints:

$$\text{top-service share} \geq 0.98, \quad \text{service entropy} \leq 0.05, \quad \text{active-hours ratio} \geq 0.9.$$

The first condition enforces near-monoclass service behavior, the second enforces very low service diversity, and the third enforces near-continuous operation in time. Their conjunction isolates entities whose behavior is more compatible with infrastructure persistence than with residential endpoint dynamics.

Application of this rule yields 5 infrastructure-like hosts and 95 residual candidates. The split should not be interpreted as a benign/malicious dichotomy; rather, it constitutes a role-disambiguation step that improves the semantic precision of subsequent analysis. Methodologically, this marks the transition from anomaly-centric prioritization to proxy-centric prioritization under explicit infrastructure control: the retained set is no longer “most anomalous” in general terms, but “most proxy-plausible after infrastructure exclusion”.

6.3 Engineering Analysis on Raw Traffic Evidence: results and interpretation

The final stage of the chapter is the raw-traffic analysis, with the explicit goal of testing whether the behavioral hypotheses derived from aggregated features remain valid when evaluated on connection-level evidence. This transition is methodologically necessary: aggregated representations are effective for large-scale ranking and clustering, but they can

mask short-lived structures, service-level concentration effects, and temporal irregularities that are critical for robust attribution.

From an analytical perspective, raw data introduce both limitations and advantages. The main limitations concern data volume, computational cost, and increased sensitivity to preprocessing choices (window definition, session normalization, and filtering criteria). At the same time, raw traffic provides decisive benefits: higher temporal fidelity, direct inspection of destination dispersion and service usage, and stronger control over false positives generated by aggregation artifacts. In this sense, raw-data analysis is treated as a validation and disambiguation layer, not as a replacement of the anomaly-detection pipeline.

Operationally, the workflow follows a sequential procedure: ingestion of shortlisted candidates, reconstruction of host-level indicators from raw records, computation of proxy-consistency metrics, separation of infrastructure-like profiles, and extraction of a residual set for engineering interpretation.

6.3.1 Data scope and reconstruction protocol

This stage re-processes raw connection records only for the candidate hosts selected in the previous section, using a 14-day window. The purpose is explicit: verify whether the hosts that appear proxy-plausible in aggregate space also exhibit relay-compatible behavior when observed at finer temporal and service granularity.

The preprocessing pipeline is structured to maximize robustness and reproducibility. Input files are first selected from raw Tstat outputs through filename parsing, and restricted to the last 14 days of observation. During ingestion, malformed records are discarded and only the columns required for proxy validation are retained (client IP, destination IP, destination port, directional byte counters, and flow start timestamp). The dataset is then restricted to shortlisted hosts through an inner join with the candidate list, using a broadcast table to keep the filtering step computationally stable at Spark scale. At feature-construction level, timestamp and traffic fields are explicitly sanitized before aggregation. Flow start time is converted to timestamp and mapped to hourly bins; directional byte counters are cast with fault-tolerant conversion and null values are replaced with zeros before computing upload share. This design avoids propagation of parsing errors and guarantees that derived indicators are computed on a numerically consistent basis.

After temporal alignment, events are aggregated per host and per host-hour to reconstruct the proxy-relevant descriptors used in this chapter: destination diversity, mean hourly destination diversity, active-hour persistence, mean upload share, service entropy, and top-service concentration. The resulting indicators are normalized to a common range and combined in the proxy-consistency score used in the next subsection. Consequently, filtering decisions are driven by reconstructed raw behavior rather than by upstream anomaly scores alone, which is the key requirement for a defensible residential-proxy validation stage. The feature set is selected with a strictly functional criterion: each variable must contribute to separating relay-like behaviour from ordinary client activity and from infrastructure false positives. Destination diversity and mean hourly destination diversity capture fan-out structure, which is expected to be elevated in mediation roles. Active-hour persistence captures temporal continuity, a necessary condition for stable proxy operation over the observation window. Mean upload share captures directional asymmetry and is used to detect outbound-heavy profiles that are compatible with forwarding dynamics. Service entropy and top-service concentration are introduced as complementary controls on service heterogeneity: together, they distinguish broad relay-like activity from near-monoclass, highly deterministic infrastructure endpoints. In this configuration, the raw

feature space is not descriptive for its own sake; it is explicitly constructed to test a falsifiable operational hypothesis on residential-proxy plausibility.

6.3.2 Residual candidate profiling

The objective is to determine whether the remaining hosts exhibit raw-traffic signatures that are structurally compatible with residential relay operation, and to separate high-confidence proxy-plausible entities from weaker borderline profiles.

The interpretation is driven by joint reading of five dimensions: destination fan-out (by watching at unique destinations ips and mean hourly destinations), operational continuity (mean active hours), directional asymmetry (average of the upload share), and intra-day variability (hourly destinations cv). Under the adopted hypothesis, high-confidence candidates are expected to combine large destination dispersion with sustained temporal presence, while borderline entities typically satisfy only a subset of these conditions.

Raw tables reveal three recurrent residual archetypes. The first is a *high fan-out, medium-high persistence* pattern, where hosts contact very large destination sets and remain active across many hourly bins; this is the profile most coherent with relay-like mediation. The second is an *always-on, high-variability* pattern, in which continuity is strong but hourly destination dynamics are unstable (hourly destination cv close to or above 1), suggesting mixed workloads or burst-driven automation rather than stable forwarding behavior. The third is an *uplink-biased persistent* pattern, where active-hour coverage is broad and upload share is comparatively elevated; these hosts remain relevant for proxy triage but require careful contextual validation to exclude specialized outbound services.

Table 6.2 reports representative residual profiles with pseudonymized identifiers. Values are extracted from raw-host summaries in the Spark notebook and are used here to illustrate interpretation logic.

Within the first archetype (*high fan-out, medium-high persistence*), hosts R1 and R2 represent the strongest relay-like signatures in the residual set. Both exhibit very large destination dispersion (30,000 and 21,868 unique destinations, respectively) combined with sustained temporal presence (23 and 13 active hours). The key point is that high fan-out is not observed as an isolated burst: it is distributed over multiple hours with moderate intra-day instability (hourly variation coefficient below 0.7).

For R1, the joint presence of extreme fan-out and near-full daily activity supports a high proxy-plausibility hypothesis. A benign explanation remains possible (e.g., enterprise gateway or shared service edge), but this explanation becomes weaker after the upstream infrastructure-pruning step. Under current evidence, R1 should therefore be treated as a top-priority candidate for analyst validation.

For R2, fan-out remains very high but temporal persistence is lower than R1 (13 active hours), suggesting a long-window but not fully continuous operation. This profile is compatible with scheduled relay usage or partially time-constrained automated forwarding. A benign interpretation could involve high-traffic orchestrators active in limited windows; nevertheless, the combination of dispersion and continuity keeps R2 in the high-risk residual subset.

The second archetype (*always-on, high-variability*) is exemplified by R4 and R5. Both hosts are continuously active (24 active hours), but their destination dynamics are markedly unstable (hourly variation coefficient equal to 1.262 and 1.118). This combination weakens proxy plausibility: continuity is present, but forwarding behavior is not temporally regular enough to support a stable relay interpretation.

Host	Dest. IPs	Mean hourly destinations	Active hours	Upload share	Hourly CV	Interpretation
R1	30,000	5,004.38	23	0.265	0.494	Very high fan-out with sustained activity; strong proxy-plausible profile.
R2	21,868	4,522.57	13	0.232	0.607	High dispersion and medium persistence; plausible relay-like behavior.
R3	10,465	1,359.12	24	0.423	0.470	Persistent and uplink-biased profile; high triage priority.
R4	12,317	1,178.72	24	0.118	1.262	Continuous activity but strong intra-day instability; borderline profile.
R5	15,376	1,852.60	24	0.211	1.118	High continuity with high variability; requires deeper contextual inspection.
R6	13,700	1,806.48	24	0.425	0.623	Persistent with moderate variability and outbound bias; proxy-plausible.

Table 6.2: Representative residual host profiles from raw-data reconstruction (pseudonymized identifiers).

For R4, low upload share (0.118) combined with high variability suggests a profile dominated by volatile inbound-heavy traffic bursts. Such behavior may be associated with content-consumption or cache-like access patterns rather than outbound relay mediation. For this reason, R4 is best interpreted as a borderline candidate with relatively higher probability of benign behavior.

For R5, the pattern is similar but slightly less extreme: high fan-out and continuity are counterbalanced by unstable hourly dispersion and only moderate upload asymmetry (0.211). A malicious interpretation cannot be excluded, but the evidence is ambiguous and compatible with mixed-use automated workloads. Operationally, R5 should be retained in a second-priority review tier rather than in the primary proxy shortlist.

The third archetype (*uplink-biased persistent*) is represented by R3 and R6. Both maintain broad temporal coverage (24 active hours) and elevated upload share (0.423 and 0.425), with fan-out that remains substantial (10,465 and 13,700 destination IPs). Compared with the previous archetype, their intra-day variability is lower (hourly variation coefficient equal to 0.470 and 0.623), which increases behavioral coherence with forwarding-like operation.

For R3, low variability and persistent outbound asymmetry make the relay hypothesis technically coherent. A benign explanation may involve large-scale backup or synchronization services, but this would typically exhibit stronger service concentration than observed in residual candidates. In absence of such concentration, R3 remains a high-confidence proxy-plausible host.

For R6, evidence is slightly less strict than R3 due to higher variability, yet still consistent with persistent mediation behavior. The combined signal (24 active hours, high

upload share, substantial fan-out) supports inclusion among top residual candidates. The most plausible alternative benign explanation is a specialized outbound infrastructure node; this possibility should be checked through ownership/context enrichment in downstream analysis.

Taken together, the host-level analysis shows that the residual set is internally stratified and cannot be interpreted as a uniform risk block. The practical implication is that prioritization should be based on joint coherence across fan-out, persistence, directional asymmetry, and temporal stability. This is the decisive contribution of residual profiling: it transforms a filtered candidate list into a structured evidence hierarchy, where each host is associated with an explicit technical rationale for further investigation.

Chapter 7

Conclusion and future work

This thesis has addressed a recurrent methodological problem in network-security analytics: how to derive reliable and interpretable evidence from traffic data when explicit malicious labels are scarce. In such conditions, anomaly detection alone is insufficient. A high anomaly score may indicate genuinely suspicious activity, but it may also capture structural heterogeneity, benign automation, or infrastructure-specific regularities. The core contribution of this work is therefore not the isolated use of a specific algorithm, but the definition of an analysis framework in which detection, interpretation, and validation are treated as successive and interdependent stages of the same inferential process.

Within this perspective, the most relevant result is the empirical confirmation that anomalous behavior is multi-dimensional and cannot be robustly represented by a single detection paradigm. Density-based detectors, latent-space structures, and reconstruction residuals reveal different projections of the same phenomenon. Their controlled integration into a unified internal score has shown a practical advantage: it reduces detector-specific fragility and produces a ranking that is more stable for analyst prioritization. At the same time, the integrated score remains interpretable, because each component retains a clear operational meaning and can be traced back to explicit behavioral evidence.

The role of the autoencoder deserves particular emphasis. Reconstruction error proved to be an effective continuous signal for prioritization, capable of preserving granularity where binary cluster assignments become coarse. However, the latent-space analysis also showed that strong ranking contrast does not necessarily imply a fragmented latent geometry. This observation has two implications. First, anomaly intensity and geometric separability are related but non-equivalent concepts. Second, reconstruction-based evidence is most useful when interpreted jointly with structural checks, rather than as an autonomous proxy for class separability. The adopted methodology reflects this distinction and, in doing so, avoids over-interpreting latent projections as direct attack clusters.

An additional contribution concerns the transition from anomaly detection to behavior characterization. The four-class taxonomy introduced in Chapter 6 was not conceived as an attack ontology, but as an intermediate semantic layer between numerical irregularity and operational reasoning. This distinction is crucial in weak-supervision contexts: a taxonomy is useful when it structures hypotheses, not when it claims attribution certainty. The resulting class distribution, although imbalanced, is coherent with the expected prevalence of persistent automated activity in large traffic populations and supports analyst triage by separating recurrent temporal archetypes with different investigation priorities.

The residential-proxy analysis constitutes the most stringent stress test of the entire framework, because it requires moving from generic abnormality toward a role-specific interpretation. The evidence collected in this work suggests that such a transition is feasible

only if candidate selection is constrained by cross-detector agreement, ranking robustness, and explicit infrastructure disambiguation. In this sense, the most important methodological outcome is the redefinition of the candidate semantics along the pipeline: from anomalous entities in a broad sense to residual entities that remain proxy-plausible after exclusion of deterministic infrastructure patterns. This change in semantics is essential for technical defensibility and directly addresses one of the most common failure modes of unsupervised detection pipelines, namely the conflation of infrastructural regularity with adversarial behavior.

The raw-traffic stage confirms this point. Reconstructing evidence at connection level does not merely add descriptive detail; it modifies the epistemic status of the results. Aggregate features are effective for scalability and global screening, but they compress the temporal and service-level structure that often determines whether a hypothesis is plausible or not. Raw reconstruction restores this structure and enables a falsification-oriented analysis: candidates are not accepted because they are highly ranked, but retained only if their behavior remains coherent under higher-fidelity observation. This multi-resolution consistency is, arguably, the most scientifically relevant property of the proposed approach.

From a broader research standpoint, the thesis supports a pragmatic but rigorous position: in unsupervised cybersecurity analytics, interpretability is not a post-hoc add-on, but a design constraint that must be encoded from the beginning. Score construction, feature selection, thresholding, and filtering rules are not neutral implementation details; they determine the validity of downstream claims. The framework developed here demonstrates that it is possible to reconcile scalability and interpretability, provided that each stage is explicitly tied to a falsifiable question and to observable behavioral criteria.

These results should nevertheless be read within their limits. The absence of externally validated labels constrains performance assessment to internal consistency, robustness diagnostics, and engineering plausibility. This is realistic in many operational settings, but it does not replace formal supervised evaluation. Moreover, some decisions remain sensitive to context-dependent assumptions, including window length, normalization choices, and threshold calibration. Even when justified empirically, these choices may require re-tuning under different network baselines. Finally, the current validation layer is primarily traffic-structural; integrating independent enrichment channels would likely improve disambiguation power and reduce residual uncertainty.

In conclusion, the thesis delivers an anomaly-driven pipeline that can concretely support passive-traffic security analysis from end to end. By integrating multi-signal anomaly evidence, behavior-oriented taxonomy, robustness-aware ranking, and raw-data validation, the framework transforms large and weakly labeled traffic corpora into a prioritized and interpretable investigation space. Its practical value is therefore substantial: it reduces analyst search space, improves consistency of triage decisions, and provides traceable criteria for distinguishing high-risk behavioral profiles from infrastructure-driven chaos. In passive monitoring contexts, where direct labels are rare and attribution is intrinsically uncertain, this contribution is not marginal; it provides a reproducible analytical methodology that enables informed, defensible, and operationally actionable decisions.

7.1 Future works

Future research should extend the framework in directions that preserve its current strengths while addressing its main constraints. A first direction concerns anomaly ranking, where additional unsupervised or semi-supervised estimators could be incorporated into the inter-

nal score with explicit calibration against tie effects and temporal drift. A second direction concerns taxonomy evolution, by validating class stability over longer horizons and by testing whether new recurrent archetypes emerge under different traffic conditions. A third direction concerns residential-proxy validation, where richer raw descriptors and graph-based interaction models could provide stronger separation between relay-like behavior and infrastructural persistence. A fourth direction concerns external validity, through replication on heterogeneous datasets and deployment scenarios to evaluate transferability of thresholds, ranking stability, and interpretative consistency.

Bibliography

- [1] Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, and Joerg Sander. Optics: Ordering points to identify the clustering structure. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 49–60, 1999.
- [2] Paul Barford, Jeffery Kline, David Plonka, and Amos Ron. A signal analysis of network traffic anomalies. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurement*, pages 71–82, 2002.
- [3] Theophilus Benson, Aditya Akella, and David A. Maltz. Network traffic characteristics of data centers in the wild. In *Proceedings of the ACM SIGCOMM Conference*, pages 267–280, 2010.
- [4] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Joerg Sander. Lof: Identifying density-based local outliers. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 93–104, 2000.
- [5] Ricardo J. G. B. Campello, Davoud Moulavi, Arthur Zimek, and Joerg Sander. Hierarchical density estimates for data clustering, visualization, and outlier detection. *ACM Transactions on Knowledge Discovery from Data*, 10(1):5:1–5:51, 2015.
- [6] Guilherme Oliveira Campos, Arthur Zimek, Joerg Sander, Ricardo J. G. B. Campello, Barbora Micenkova, Erich Schubert, Ira Assent, and Michael E. Houle. On the evaluation of unsupervised outlier detection: Measures, datasets, and an empirical study. *Data Mining and Knowledge Discovery*, 30(4):891–927, 2016.
- [7] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. Technical Report TR 07-017, Department of Computer Science and Engineering, University of Minnesota, August 2007.
- [8] Taeshik Choi, Jiyoung Lee, and Taekyoung Kim. Understanding the proxy ecosystem: A comparative analysis of residential and open proxies on the internet. *IEEE Access*, 8:111368–111380, 2020.
- [9] Zakir Durumeric, J. Alex Halderman, et al. The security impact of https interception. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2017.
- [10] Martin Ester, Hans-Peter Kriegel, Joerg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD '96)*, pages 226–231, 1996.
- [11] Roy T. Fielding, Mark Nottingham, and Julian Reschke. Http semantics. RFC 9110, IETF, 2022.

- [12] Giacomo Giuliani, Martin Klenze, Markus Legner, David Basin, and Adrian Perrig. Internet backbones in space. *ACM SIGCOMM Computer Communication Review*, 50(1):75–82, 2020.
- [13] Markus Goldstein and Andreas Dengel. Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm. In *KI-2012: Poster and Demo Track*, 2012.
- [14] Guofei Gu, Roberto Perdisci, Junjie Zhang, and Wenke Lee. Botminer: Clustering analysis of network traffic for protocol- and structure-independent botnet detection. In *Proceedings of the 17th USENIX Security Symposium*, pages 139–154, 2008.
- [15] Mark Handley. Delay is not an option: Low latency routing in space. In *Proceedings of the 17th ACM Workshop on Hot Topics in Networks (HotNets '18)*, pages 85–91, 2018.
- [16] Oltjon Kodheli, Eva Lagunas, Nicola Maturo, Suvra Sekhar Sharma, B. Shankar, Juan Fernando Montoya, Jeremy Duncan, Danilo Spano, Symeon Chatzinotas, Bjorn Ottersten, et al. Satellite communications in the new space era: A survey and future challenges. *IEEE Communications Surveys & Tutorials*, 23(1):70–109, 2021.
- [17] Hans-Peter Kriegel, Matthias Schubert, and Arthur Zimek. Angle-based outlier detection in high-dimensional data. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 444–452, 2008.
- [18] Anukool Lakhina, Mark Crovella, and Christophe Diot. Diagnosing network-wide traffic anomalies. In *Proceedings of the ACM SIGCOMM Conference*, pages 219–230, 2004.
- [19] Anukool Lakhina, Mark Crovella, and Christophe Diot. Mining anomalies using traffic feature distributions. In *Proceedings of the ACM SIGCOMM Conference*, pages 217–228, 2005.
- [20] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM)*, pages 413–422, 2008.
- [21] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297, 1967.
- [22] Arash Mani, Micah Sherr, Amir Houmansadr, and Negar Kiyavash. An extensive evaluation of the internet’s open proxies. *arXiv*, 2018.
- [23] Xiang Mi, Xuan Feng, Xiaojing Wang, Zhiyun Qian, XiaoFeng Zhang, Xiao Chen, et al. Resident evil: Understanding residential ip proxy as a dark service. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2019.
- [24] Xianghang Mi, Siyuan Tang, Zhengyi Li, Xiaojing Liao, Feng Qian, and Xiaofeng Wang. Your phone is my proxy: Detecting and understanding mobile proxy networks. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2021.

- [25] Francois Michel, Martino Trevisan, Danilo Giordano, and Olivier Bonaventure. A first look at starlink performance. In *Proceedings of the 22nd ACM Internet Measurement Conference (IMC '22)*, 2022.
- [26] Vishwa Mohan, Alessandro D’Alconzo, Danilo Giordano, and Marco Mellia. Measuring mobile starlink performance: A comprehensive look. *IEEE Open Journal of the Communications Society*, 2025.
- [27] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14*, 2002.
- [28] Diego Perino, Matteo Varvello, and Claudio Soriente. Proxytorrent: Untangling the free http(s) proxy ecosystem. In *Proceedings of the Web Conference 2018 (WWW)*, pages 1021–1030, 2018.
- [29] Marco A. F. Pimentel, David A. Clifton, Lei Clifton, and Lionel Tarassenko. A review of novelty detection. *Signal Processing*, 99:215–249, 2014.
- [30] Haakon Ringberg, Augustin Soule, Jennifer Rexford, and Christophe Diot. Sensitivity of pca for traffic anomaly detection. In *Proceedings of the ACM SIGMETRICS Conference*, pages 109–120, 2007.
- [31] Lukas Ruff, Robert A. Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib A. Siddiqui, Alexander Binder, Emmanuel Mueller, and Marius Kloft. Deep one-class classification. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 4393–4402, 2018.
- [32] Bernhard Schoelkopf, John C. Platt, John Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.
- [33] Robin Sommer and Vern Paxson. Outside the closed world: On using machine learning for network intrusion detection. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 305–316, 2010.
- [34] Andrea Sordello, Zhihao Wang, Kai Huang, Alessandro Cornacchia, and Marco Mellia. Poster: The potential of erroneous outbound traffic analysis to unveil silent internal anomalies. In *Proceedings of the 2025 ACM Internet Measurement Conference (IMC '25)*, 2025.
- [35] Bernhard Tellenbach, Martin Burkhart, Dominik Schatzmann, Brian Trammell, and David Gugelmann. Accurate and adaptive network anomaly detection. *IEEE Transactions on Dependable and Secure Computing*, 8(5):719–733, 2011.
- [36] Marina Thottan and Chuanyi Ji. Anomaly detection in ip networks. *IEEE Transactions on Signal Processing*, 51(8):2191–2204, 2003.
- [37] Mohammed Tieby, Pierre Khoury, and Elias Bou-Harb. Characterizing and analyzing leo satellite cyber landscape: A starlink case study. In *Proceedings of the IEEE International Conference on Communications (ICC)*, 2024.
- [38] Giorgos Tsirantonakis, Panagiotis Ilia, Andreas Apostolopoulos, Jason Polakis, and Sotiris Ioannidis. A large-scale analysis of content modification by open http proxies. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2018.

- [39] Zhihao Wang, Alessandro Cornacchia, Andrea Bianco, Idilio Drago, Paolo Giaccone, Dingde Jiang, and Marco Mellia. Chamaleonet: Programmable passive probe for enhanced visibility on erroneous traffic. *IEEE/ACM Transactions on Networking*, 2021.
- [40] Houssam Zenati, Chouki Romain, Chuan Sheng Foo, Benjamin Lecouat, and Vijay Chandrasekhar. Efficient gan-based anomaly detection. *arXiv*, 2018.
- [41] Xinyi Zhang et al. Shining a light on dark places: A comprehensive analysis of the open proxy ecosystem. *Computer Networks*, 215:109188, 2022.
- [42] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeyoung Cho, and Haifeng Chen. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International Conference on Learning Representations (ICLR)*, 2018.

Acknowledgements

First of all, I would like to thank my supervisors Professor Marco Mellia and Professor Danilo Giordano for their support and guidance throughout the development and writing of this thesis. A very special thank you also goes to Martino Trevisan, who supported me from the very beginning and, through his experience, taught me a great deal about the analytical approach that I tried to develop and make my own during this time.

From now on, I will write in Italian so that I can better convey my words to those directly involved.

Le prime persone che voglio ringraziare sono Mamma e Papà. Da sempre mi hanno incoraggiato e supportato nello studio e, anche se non sono mai stato uno studente modello, mi hanno sempre dato l'amore e la forza per permettermi di raggiungere questo risultato. Vi voglio tanto bene. Dopodichè vorrei ringraziare mio fratello Alberto, che ho sempre avuto come punto di riferimento, ho sempre ammirato, e da lui ho imparato a stare al mondo. Vorrei ringraziare anche i miei amici di sempre Rick e Dogs con cui sono cresciuto insieme e da piccoli e stupidi siamo diventati grandi e stupidi.

Adesso è il turno degli Astroboyz. Anche se ci conosciamo da relativamente poco tempo, devo dire che mai mi sarei aspettato di legare così tanto con altre 3 persone dopo i 20 anni. Avete il potere di farmi passare il tempo insieme senza pensieri e divertendomi come un bambino. Siamo tutti diversi ma per qualche ragione tutti uguali. Da anni, ammiro tutti voi e mi insegnate ad essere una persona migliore, ognuno per un aspetto diverso. Nessuno mi sprona, anche involontariamente, ad essere un informatico migliore meglio di Cle. Par ha una forza interiore che riesce a darmi energia come nessun altro. Andreone più volte mi ha salvato, parlandomi e facendomi capire cose che non sarei mai riuscito a capire da solo. Voglio ringraziarvi con tutto il mio cuore perchè siete gli amici che non voglio perdere.

Infine, voglio ringraziare Andreea. Questa tesi è dedicata a te, perchè anche se sei arrivata improvvisamente meno di due anni fa, mi hai cambiato la vita come nessun altro. In poco tempo sei diventata la mia ragazza, la mia migliore amica e la mia famiglia. Per quanto per la maggior parte delle persone sarebbe un incubo, per me condividere tutti i giorni, sia quelli ordinari che quelli difficili, con te é il mio piacere più grande.

Prima di incontrarti non avrei mai pensato di stare così bene e di poter amare così tanto un'altra persona. Sei riuscita a farmi scoprire una versione di me che non conoscevo: più sereno, più completo, più felice

Spero di avere l'onore di passare con te tutto il resto della mia vita, costruendo insieme il nostro sogno. Non vedo l'ora di scoprire cosa la vita ci prospetterà. Grazie per esistere e sopportarmi, ti amo.

Vorrei chiudere ringraziando tutti i presenti, che hanno dedicato del tempo per questo mio momento importante.