POLITECNICO DI TORINO

Master Degree in COMPUTER ENGINEERING

CyberCraft, a security serious game

Advisors
Prof. Antonio Lioy
Eng. Andrea Atzeni

Candidate
Lu Yang

Dicembre 2018

Acknowledgements

I need to express my great thank to my guide, professor Antonio Lioy and engineer Andrea Atzeni, for their patience and their help during the thesis. Also, special thanks to the company MAVI Interactive for the kind offer of access to its games, as well as Circadence for additional information about its products. Finally, my thanks are dedicated to my parents, for their great concern and their constant reminder and suggestions.

Contents

List of Figures

Chapter 1

Introduction

This is an era of a digitalized world. Smartphones, APPs, websites, automated management systems, E-commerce and computer-aided design are found all around the world. Not to say the trend of exploiting data mining and artificial Intelligence. With the booming Internet of Things, the whole world will go digital. Nevertheless, promising as the ICT field has shown, a great threat comes with it. The more widely the cyber world covers, the greater danger its malfunction can cause. Especially, among all the factors of computer system malfunctioning, cyber-attack, performed by people of evil intention and extraordinary skills, can cause more and greater disasters to the world. The 2010 Stuxnet attack caused severe facility damage; 2013 Yahoo data theft leaked billions of Yahoo account information; 2017 Wannacry attack encrypts files for ransom on computers worldwide, especially British hospitals... What's worse, cyber-attacks have been more and more frequent, with a yearly increase of 23% at 2017 [55]. Attackers have also become smarter in choosing more critical targets [55]. The academia requires much more security experts to fortify the defenses of commercial and governmental data and systems; the main street wants also the general public to be security aware, so as to better survive in the numerous security attacks; the software industry also need more security aware programmers, to create software and systems not too much susceptive to security attacks.

For the work of raising the security awareness of the public, one common practice is through the creation of security websites, where security tips are listed. There can also be security lectures or propaganda. Unfortunately, these methods are sometimes considered dull and unattractive to the target audience. For the work of training for security experts, the academia usually follows the traditional way of education. Nevertheless, despite their effort, the number of security experts trained is far from being enough to meet the ever-increasing demand for more [58]. For the work of fostering more security aware programmers, universities offer optional or compulsory courses in computer security to students of ICT field. However, students, especially the new learners of computer programming, are usually engaged with syntax, algorithms, logic flows, and care little about the robustness of the code, not to say the code's resistance to security attacks [58]. The security courses are sometimes considered quite theoretical, with a large amount of separate knowledge points to memorize. These knowledge points, if just memorized in preparation for the exams, will not last long in the future.

The term "serious game" represents a type of games that are used for the purposes besides just entertainment. After the first idea of using a serious game in the 70s [11], it has been gradually accepted and adopted in the 21th century, especially along with the boom of digital game industry [10]. Since then, serious game has been receiving an increasing attention yearly. Game-based learning especially, as the use of serious game for educational purposes, grows even

faster. Game-based learning generates a world revenue of 2.6 billion in 2016, with a five-year compound annual growth rate of around 22.4% [31]. It's 2021 estimation boosts to 7.3 billion [31]. Serious games made for the purposes of education have been more and more widely witnessed on the market. Those like *The Tactical Language Training System*, *Perfect World*, *You Make Me Sick!* and the recent *Variant: Limits* are all examples of this.

The advantage of the pedagogic serious game is that, it's created by combining the theories in education with the psychological techniques from the game industry. Made with the purpose of education, players of serious games gain more than just fun. Being intrinsically a game, it attracts and motivates the players with animation, sound, story, as well as gameplay, where the creators of traditional materials think little of. By actively participating in the game, players of serious games seldom get absent minded or sleepy, which is a big problem with traditional education. By nourishing interest in the topic, a serious game can boost motivation in traditional learning or even promote self-study.

To overcome the difficulties and challenges in the training and education in cyber security as mentioned above, serious games in cyber security are also created. MAVI Interactive concentrates on creating immersive training game series like *Info Sentinel* and *Agent Surefire*, targeting the cyber security awareness of common people. Naval Postgraduate School has developed *Cyber-CIEGE*, which teach the students the knowledge in security including encryption, patching, access control and many others. Circadence has created *Project Ares* to provide teams of workforces with cyber security training and hand on practice.

In order to alleviate the pressing demand for the delivery of knowledge in cyber security, this article concentrates on the development of an educational serious game in cyber security, as well as how some related principles in education and in game design can be adopted in the game development. The primary target audience of the serious game is the undergraduate and graduate students.

Chapter 2 lists some knowledge about the learning principles that is related to the students' involvement and effectiveness of learning, which lies as a foundation for the pedagogic serious games.

Chapter 3 introduces the term of serious game and the usage of serious games. Special attention is given to the pedagogic serious game, with a special methodology called tangential learning. Some examples of serious games are also discussed, in terms of how the principles and rules related to learning and have been used to these games.

Chapter 4 supplements some other theories that come from the field of entertainment game design, e.g. flow, decision and skill, operant conditioning and Fogg behavior model. These theories are more often shared among entertainment game designers, but are somewhat overlooked by serious game designers. These theories and techniques present as missing points for the pedagogic serious game.

Chapter 5 illustrates the current urgent situation calling for more security experts, as well as the security-aware public and software developers. With this demand, security serious game can be proven as an effective tool on the emergent issue. Some serious games specifically targeting computer security are mentioned here.

Chapter 6 discusses the choice of platform and the tools for the security serious game that is developed along with this thesis article.

Chapter 7 elaborates the design of the serious game called CyberCraft, with special attention to how the pre-mentioned theories and techniques can be applied to the serious game.

Chapter 8 looks into the future for the potential work that worth doing to improve this serious game or to create other better pedagogic serious games.

Appendix A is the user manual, which is supposed to help anyone who is interested in playing or modifying the game. It including installation (if needed) and other game-related data that is not elaborated in the tutorial of the game.

Appendix B is the programmer's manual, which targets people intended to modify, reuse or extend the game.

Appendix C is the successor's manual, which is made for any people, potentially also non-programmers, who want to extend the game by adding more levels to it. The format of the configuration files for the scenarios is discussed.

Chapter 2

Why not learn

Centuries after machinery driving many manual workers off work, the recent rise of artificial intelligence is also going to bring a great impact to the low-level knowledge workers. On the other hand, the new century is an era of knowledge explosion. It has been an even higher demand for people that are more educated and trained. Learning, as an unavoidable path to get educated and trained, has been an essential part to one's life. Just take a look at how many students going to universities every year, as compared to the last century. Theories about learning and teaching, however, have to be firstly discussed, before one can effectively create something to boost learning. Centuries after machinery driving many manual workers off work, the recent rise of artificial intelligence is also going to bring a great impact to the low-level knowledge workers. On the other hand, the new century is an era of knowledge explosion. It has been an even higher demand for people that are more educated and trained. Learning, as an unavoidable path to get educated and trained, has been essential part to one's life. Just take a look at how many students going universities every year, as compared to last century. Theories about learning and teaching, however, have to be firstly discussed, before one can effectively create something to boost learning.

2.1 Principles of learning

Edward Lee Thorndike, a psychologist specializing in educational psychology, has developed three laws of learning: the law of readiness, the law of exercise and the law of effect [1].

The principle of readiness tells that, the more a person is ready to learn, the better she/he will learn. Emotionally speaking, if the learner is interested in the subject, and has a clear goal, the learning effectiveness grows. Contrarily, if the learner finds no value in learning, and is unwilling to learn, the learning effectiveness dramatically drops. Physical readiness also matters, for the fact that, if the learner lacks sleep, or have some basic needs not met, she/he could not perform very well in learning either. When it comes to mental readiness, it's clear that external distractions or anxieties also hamper the learning.

The principle of exercise says that students learn best with meaningful repetition. It just shares the similar ideal with the idiom "Practice makes perfect".

The principle of effect means that, the learning effectiveness is enhanced by incidental favorable emotions, and impaired by incidental unfavorable emotions. An example of this is that success encourages a person to continue, while failure discourages. The law of effect is closely related to operant conditioning, which will be elaborated in section 4.4.1.

Many other principles of learning have also been discovered and summarized, like the principle of primacy and principles of intensity. After all, pedagogy itself is a deep subject that worth exploring generation by generation. But they are not to be discussed here.

2.2 Driving forces

It has long been known that, the process of study is usually tough or even frustrating, especially when one encounters confusions or ambiguities among the knowledge. Whereas, why are there so many students going to schools and colleges every year? They are five incentives:

1. Punishment (pressure). Students are told to continue, because if they don't, they have to endure punishment. They may fail the exam, they may fail to achieve the degree, and they may be unemployed in the future. The students suffer during the study, but the fear for what will happen if they stop frightens them more, and this fear gets them back to the desk. The punishment is an external force pushing the students forward.

2. Positive feedback (awards). Students work hard, because by doing so, they get good marks, they get praised, they get good jobs. The students still suffer in the process, but the pursuit for awards drives them to continue. The positive feedback is an external force pulling the students forward.

3. Confidence and competence. When the students finish the study, they became capable of handling difficult situations. They get confident in themselves, and they get a feel of competence and a feel of control. The students still suffer the harshness of study, but they enjoy the moment when they have acquired the knowledge. Confidence and competence is an internal force pulling the students forward.

4. Interest. Students work hard, but they don't feel any pain in the process. On the contrary, they enjoy engaging this subject. They may sit in front of the desk for hours, with joy, unaware of the sun's falling or the belly's call. The interest is another internal force pulling the students forward.

5. Habit. Students have already developed a habit, an everyday routine to study. Habit is actually not a motivation to study. The students are just (effortlessly) following their natural tendency to keep with their routine.

The punishment has long been used among human society, not only in learning, but also in governing as well as pupil education by parents. The punished individual certainly does not like this, and it may raise hostility and a sense of rebellion. Positive feedback is more welcomed by people, and some parents have already learnt to use awards instead of punishment for the education of their pupils, which yields more effectiveness and less sense of hostility. Game makers know a lot about positive feedback, and they provide instant feedback (as compared to delayed feedback) to enhance its effectiveness. This will be elaborated in section 4.4.1, where operant conditioning is addressed.

Confidence and competence (personal development and growth) is an internal force, which works better than external forces mentioned above. This force works even better for language studies, because when one is confident in his/her language skill, she/he will not slip away at the chance to practice the language. Practice and repetition, on the other hand, has been acknowledged as the best way to progress one's language skill (as the principle of exercise mentioned

above). Unfortunately, seldom any techniques are found exploiting the pursuit for confidence and competence. It's usually left to the learner themselves to discover such a force.

Interest is probably the most effective driving force to the learner. Think of someone who is weary of geographic lessons, can remember very well the map of Azeroth. Someone who usually forgets English words can spell out the long cheat codes of a Game. Don't just blame them for not studying hard. There has to be a reason within it, and the reason is that, the more interested you are in a subject, the better you learn and remember (as applied to by the principle of readiness). It's really uncommon that students talk about academic knowledge in their leisure time, given that there are no exams or jobs that require them to do so. However, they do discuss game strategies, fighting styles, both face to face, and even over the internet. It's even more intriguing to see that, for games as serious as Kerbal Space Program, which builds a solid foundation over the real world science, the players' discussions do involve academic knowledge, but they just happy talking about it. It's Interest that attracts them to engage. Nevertheless, the interest differs from people to people and differs from subject to subject. There have not been many discoveries on how to cultivate interest. The only known method is just to expose someone to the subject. Games have the potential to nurture interest, as will be addressed in the follow section. Though not a real motivation, habit impels greatly to a person's behavior. TalBen Shahar in his Harvard course of Positive Psychology (lesson 11) [2], have emphasized that, seldom any "good students" study hard because of their strong self-discipline. They do so only by following their habit (the word ritual was used in place of habit as he put to it). He also claimed that, for the vast majority of people, the amount of self-discipline is quite limited, and this amount cannot be improved. Self-discipline is consumable. Whenever one uses self-discipline to refrain an action, a portion of self-discipline is expended. To fight another distraction, or to battle more unwillingness with the current work, another portion of self-discipline is used. If the distractions are so much, or the current task is so much unsatisfying, one's quota of self-discipline will soon deplete. By then, the person just does what his/her nature tells, not being able to regulate anything. Thereby, the use of self-discipline works only in a short term, and one can only rely on habits for the long term.

The formation of a habit has its psychological explanation, in the sense that, habit conventionalizes our behavior, saving brain intervention and speed up actions in the future [3] [4].

2.3 Active learning

Teaching is not just an issue of the teacher, but rather more of the students. It happens that, the teacher is talking passionately, while the students feel bored or puzzled. The same thing happens also to asynchronous media like books, articles, slides, or recorded lectures. The students could feel it difficult to concentrate on the topic, and they get absent minded easily. When the video is over, or the article is finished, they feel a sense of void, as if the video has not been watched and the article not been read. The key problem here is that, the students' thought has not followed the thought of the teacher or author.

There are many reasons for this, like lack of sleep, dull speech, lack of interest, insufficient background knowledge or distraction. It could also be that the teaching process emphasizes on rote memorization, lacking the presentation of internal logic or causation. The students could also feel that the topic is far away from them, and probably even worthless to them (no motivation to learn). But one thing needs special attention here: all these media are passive (as related to rote learning and didactic learning method alike). Passive media means that the students don't really have to think, when the progress bar moves forward. What's more, the author or the lecturer reasons in their own thought patterns, which may be not consistent with that of the student (e.g. someone maybe prefer top-down approach, and dislike bottom-up approach, which student feels a

little unaccustomed to). As a result, the audience needs to readjust the thought to match that of the author, creating a delay before understanding. This will be even worse if the audience lacks sleep, is distracted, or has some preliminary knowledge not fully understood. Thereby, delivery through passive media suffers a penalty from inter-brain communication. Confucius, the ancient Chinese teacher once said, "Learning without thinking leads to confusion". After all, been-taught alone is still exterior to the learner, only her/his own thinking matters most for the learner.

There is also a psychological interpretation of this. According to Mihaly Csikszentmihalyi, human brain can discriminate around 126 bits of information per second. Unfortunately, we speed 40 bits per second just by decoding speeches [5] (In his TED lecture however, 110 bits/s and 60 bits/s respectively [6]). This has implied that knowledge transferred by speech actually suffers a considerable penalty, as the listener's brain has, at maximum, no more than two-third its power left for the task itself. This is one of the reasons why it's difficult to read while listening, except that the two channels deliver the same sentence. On the other hand, when the learner is trying to think and deduce all by himself/herself, at least this 40 bits/s bandwidth can be saved.

In order to solve this problem of didactic inefficiency, the term active learning was introduced by Reginald William Revans. Active learning emphasizes on directly involving the students to the learning process [7]. This includes the student's doing, and thinking after doing. It underlines the fact that, in the process of teaching, the teacher can play a role of guidance, but never replacing the student's thinking, because the latter actually matters the most.

> I never enlighten anyone who has not been driven to distraction by trying to understand a difficulty or who has not got into a frenzy trying to put his ideas into words.
>
> — Confucius

It's hard to believe that one long ago at 500BC is already aware of the importance of the students' own deduction to the progress of study. These words emphasizes on the fact that, it should be the learner's thinking that leads the teacher's guidance; rather than the teacher's explanation leads learner's thinking. Active learning also avoids the situation of "not really understand". An un-neglectable portion of students prepare the exam with rote memorization without really understanding the knowledge points. The knowledge was put in short term memory and fades with time quickly, so that they can't remember a single word just after a month or two. With active learning, everything changed. Jerome Bruner(1961) has also stressed that, the thinking and problem solving skills, rather than the knowledge itself, should be the real target of the education. He also claims that students are active learners who construct their own knowledge, by organizing the information with a coding system. The coding system, however, can be more quickly built by his own discovery than by the teacher's instruction. This is the main concept of discovery learning and constructive approach. [9]

With students' real experience and own thinking, analysis and deduction, the knowledge become meaningful. The students really understand the knowledge and the new knowledge is built over and closely linked to their prior knowledge. With this experience, the neural paths in their brain are built. As a consequence, the knowledge is put in the long term memory and the retention dramatically prolonged.

It's quite the case in the field of computer programming. It's usually unavoidable, especially for the new learners of a programming language, to read the codes written by other people. Nevertheless, new learners nearly always find some difficulties concentrating on the codes. As a matter of fact, the idea that "reading other's code is tough" is widely acknowledged among programmers. It happens that a thousand codes come from a thousand programmers' hands. Each programmer has his/her own logic, preference and style. Reading other's codes means the reader have to follow other people's style. On the other hand, the readers of code tends to stay

shallow (e.g. this line of code calculates a value), without deep thinking of why it has to be written in such a way (e.g. the data format allows me to do this calculation) or what constraint it has (e.g. but the calculation will fail if the data is in another format). As a result, even after having read other's codes multiple times, the learners are still challenged writing their own codes. Only by the moment they have written their own version, have them really learnt the skill. Programming on one's own is an inescapable step in the process to learn programming.

> What's learned from books is superficial after all. It's crucial to have it personally tested somehow.
>
> — Lu You

As opposed to traditional lectures, exercise classes or laboratory sessions (as related to experiential learning) are active activities or say, interactive activities. During these sessions, the students are supposed to actively participate in the activities, searching for information themselves and applying their newly acquired knowledge to solve a real problem. In this case, their brains are always working on the task, which makes them nearly impossible to feel sleepy. Some teachers also ask in class questions or leave off class problems for the students to solve. This is also a practice of applying active learning. If unfortunately the students failed to work out a solution on these questions because of some reasons, a later reveal of the solution (together with a detailed procedure to solve) will complete the circle.

> Students remember only 10 percent of what they read; 20 percent of what they hear; 30 percent, if they see visuals related to what they are hearing; 50 percent, if they watch someone do something while explaining it; but almost 90 percent, if they do the job themselves, even if only as a simulation.
>
> — Menn, Don. [27]

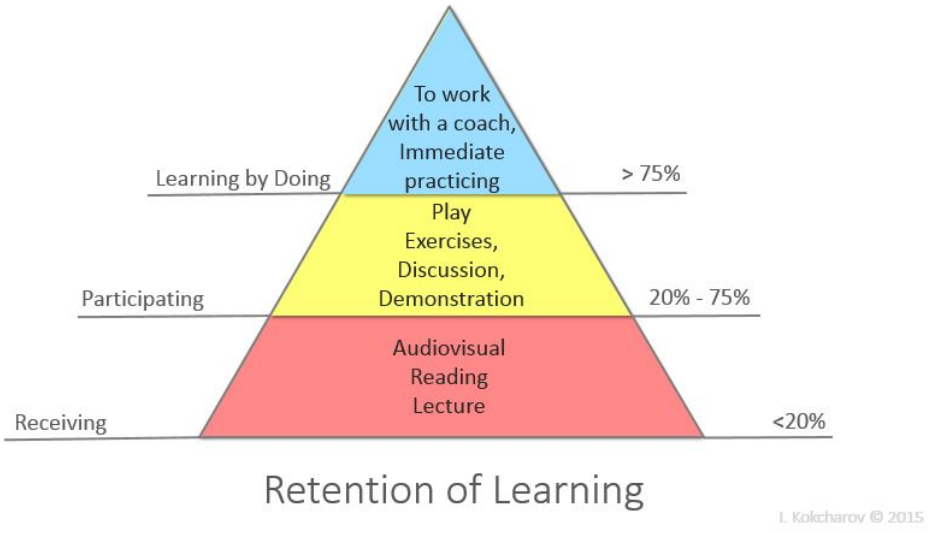

Figure 2.1. learning retention pyramid [8]

All in all, the comparison between different communication media is given as follows:

	passive	**active**
less attractive	verbal or textual material	lab sessions
fancy	infographic/animations/movies	games

It can be seen that, games has better attraction (compared to lab sessions) and it introduce an interactive environment (compared to graphic added material or animations or movies), which bestow it the most potential in rendering knowledge.

Chapter 3

Serious game

3.1 What is serious game

Serious games, sometimes is also referred to as immersive learning simulation, digital game-based learning, gaming simulation [28] or edutainment [29].

The term "serious game" can date back to 1970, when firstly introduced by Clark C.Abt [11]. In Abt's 1975 book *Serious Games*, where he clarified the idea of using games for non-entertainment environment for something more than just fun [29] [12]. A more widely used definition was given by Zyda as "a mental contest, played with a computer in accordance with specific rules, that uses entertainment to further government or corporate training, education, health, public policy, and strategic communication objectives" [11]. After that, Ben Sawyer has formally created the term "serious game" in 2003 [12]. Afterwards, Michael Zyda gave a more open definition in 2005 as "Serious games have more than just story, art, and software, however. (...) They involve pedagogy: activities that educate or instruct, thereby imparting knowledge or skill. This addition makes games serious" [29] [12].

Serious games were firstly created for military purposes, while latter been extended to the educational and business world in 20th century [10]. Early in the 2000s, the rise of digital game industry has brought serious games with new life, entitling it a substantial industry and an independent field of research [10]. Now serious games are adopted in many fields including military simulation, health care training, emergency resolution, corporate training, virtual tour, education, as well as behavior change.

Eliane Alhadeff [30], owner at Future-Making Serious Games, which is considered the thought leader in the fast-growing Serious Game Market, expected in 2008, 1.5 billion of dollars would have been invested worldwide to the field of serious game; while others expected billion dollars by 2011 [28]. Sam S.Adkins, the Chief Researcher Officer at Ambient Insight, reported in his 2016-2021 estimation in game-based learning that, products featuring game-based learning experiences worldwide revenues of 2.6 billion in 2016, with a five-year compound annual growth rate (CAGR) of around 22.4%. An 7.3 billion revenue is expected by the year of 2021 [31]. The private investments in game-based learning reached 322.6 million just by the first half of 2016, nearly twice as much as the amount of the full year 2015. Sam S.Adkins also emphasizes that, (considering the 2016-2021 forecast) the greatest growth of learning technology focus on simulation-based learning, game-based learning, cognitive learning and mobile learning. Among them, game-based learning witness the highest growth rate of 22.4%, much more than its 2006-2011 forecast period, and

simulation-based learning sees the second growth rate as 17.0%. SRI international, reported in 2014 in its research paper that, simulation-based learning and game-based learning were proven to be dramatically more effective in knowledge transfer than those without simulation or game play [31]. A Stanford study in 2015 tested the effectiveness of edugames to third grade math. It has discovered that, those students played *Wuzzit Trouble math* (for no more than 2 hours in total), experiences a 20.5% improvement rate, as compared to those students with the same material in traditional formats [31].

Sam S.Adkins informed that [31], game-based learning products, together with other simulation products, required great effort and money to create. This situation changed recently, especially in 2015, thanks to many new game engines coming to the market. These engines, as are also called as authoring tools or authoring systems, usually put on no requirement in coding skills from their uses. Optimized for their targeted game design or e-learning, the users can design and implement their own learning products, with less effort and in a shorter amount of time. Examples of authoring tools include *gamesalad*, *construct*, *elucidat* and many others.

3.2 Gamification

There are many names similar to and partially overlap with serious game. These include e-learning, game-based learning and gamification.

The term gamification especially, is intrinsically different to serious game. Gamification is also a synonym for gameful design. One of its definitions is "the process of using game thinking and game mechanics to solve problems and engage users" [10]. A more accurate definition is given as "Gamification is the use of game design elements in non-game contexts." [10] It can be found that, the core difference before serious game and gamification is that, serious game is primarily and substantially a whole game, and people add more serious values to it. On the contrary, gamification means the work is primarily a non-game product, and people add game-related stuffs to it, potentially piece by piece, enhancing the product's attraction to the users. Gamification more often found with these elements from the game design: the use of points, badge, skills, leader board, exp & levels, pictures and other fancy staffs are also bonus point.

In this sense, *Civilization* series and *Kerbal Space Program* can be considered serious games, while *Duolingo* is gamification (or say, a language learning software with gamification). Nevertheless, the readers of this thesis should not overemphasize the difference between serious games and gamification, as there are also products that tends to stand in between the two, just like *Civilization* series standing in between serious and entertainment games.

3.3 Why serious game

Serious game, in the form of simulation, creates a virtual environment, which assembles to the real situation. The players can act freely under the simulated environment, just like in real life, but does not have to suffer loses when the players performed badly [28]. e.g. in a military simulation game, the players will not get injured or even kill when they are exposed to enemy's fire; in a medical simulation, no real patient will die of a mistake in the operation; in an emergency simulation, the building will not blow up because of the failure to dismantle the bomb. Serious game also helps to create and to reproduce the situations that is hard to meet in real life (e.g. a street battle or a patient with a heart attack)

The current young generations are called "digital natives". They grow up with digital technologies all around them: computers, smart phones, instant messages, social networks, etc. They are already used to harness the power of internet as the a patient tutor and a powerful support, when they encounter problems with homework or investigations [27].

This generation is also familiar with digital games. 30% of the 6-17 year-old children play games once a week [Summit on Educational Games: Harnessing the power of video games for learning]. Not only children, 50% of Americans and 75% of American heads of household play computer and video games [27].

According to research, 8-18 year-old children play video games 50 minutes every day; adult male 7.6 hours per week and adult female 7.4 hours [27]. Computer and video games are not just favorite toys of the youth either. 35% of them is no more than 18 years old; 43% of them is 18-49 years old. The popularity of LOL (on PC) and Arena of Valor (on smart phone), which absorbs players ranging from little kids to company employees, has reaffirmed these statistics. In this digitalized world, full of computer and video games, it's obvious to see that people can nurture their affinity to serious games also.

3.4 Pedagogic serious games

> "All forms of play are learning and all forms of learning are play"
>
> — Wong et al [29]

There is a common tendency, especially by the parents, to consider learning and playing as two opposite and even contradictory sides [29]. They consider playing as something joyful and entertaining, but also unproductive and addictive. Learning, on the other hand, is considered beneficial, but requires great effort and dedication. The proverb that "no pain, no gain" is an example of this concept. While it is correct to get the students prepared for the harshness in learning, learning does not always has to be tough, boring and without fun. As a matter of fact, playing itself is strongly associated with learning. The player repeats the game, practice after practice, meeting the challenge and polishing the skill, until the point of mastery (denoted as skill ceiling as section 4.2.3)

> Anyone who tries to make a distinction between education and entertainment doesn't know the first thing about either
>
> — Marshall McLuhan [32]

It's suggested by cognitive and learning science and the panelists of National Summit on Education Games that, some features of games should be emphasized on when developing a game for learning [27]:

- Clear learning goal. Good objectives helps the players know why they need to learn the knowledge.
- Opportunity to practice and challenge. In the majority of the games, the players are never expected to deal with abstract and dull question and answers. Instead, they respond concrete situations and with their own hand.
- Monitor the player's level of mastery and adaptive challenge to it. Keeping the learners at learning zone (as opposite to comfort zone or panic zone) makes the learning most effective.

- Encourage inquiries and motivate the learners to seek out solutions.
- Context bridging. By putting players in the situations where they need to apply the knowledge to overcome the challenges, they are undoubtedly aware of where the knowledge can be used.
- Time on task. Games are adept in keeping its players continuously engaged, and as a result for serious game, the learner engaged. As the player continuously stays in the game, the law of exercise is met.
- Motivation. Players are driven by the objectives, they could retry even after several failures. Perseverance like this is not widely seen in the field of education.
- Hints and extra support. Prompt, cues can be provided to the players when they are found stuck.
- Personalization. It has been already widely acknowledged that, each person has his/her unique potential. For example a person may do excellently in politics, while terribly in art. This leads to one indisputable fact, that different students learn certain knowledge at different paces. In traditional education, the teachers, however, usually have to face a big bunch of students at the same time. If the teacher really provides dedicated time for each single student, offering personalized courses specially for each of them, the amount of work and time needed is beyond imagination. What a compromise the teachers have done is that, they try to teach in an average pace of the students, leaving the quick learners bored and the slow learner lagged behind (this also has a lot to do with flow, which will be elaborated in section 4.1). Only personal tutor can adjust its pace according to an individual, but their high wage has already revealed the great cost of time and effort within. To each another student, another chunk of time and effort is dedicated. A pedagogic serious game, or educational software, however, once completed, can serve numerous students, giving private and customized pace for each one of them, while having no additional cost for each additional student.
- Infinite patience. Teachers and professors cannot afford their time for detail problem solving for each students. Not even with the help of some teaching assistants. The current teacher-student ratio makes it nearly impossible for the teachers to dedicate their time and effort for their students (like the figure of the teacher in The Chorus and in Taare Zameen Par). Reversely, the teachers (unavoidably) get impatient and annoyed if they are to face constant questions and calls for help. They may carelessly and unintentionally say something like "you are a silly boy". These words detriments self-esteem and greatly discourages the students' motivation on the subject (see operant conditioning 4.4.1), and the students will perform even worse (Pygmalion effect). Fortunately, this is never a problem for computer.

There are also other features of games that help boost learning:

Game design usually features the flow. The flow requires intrinsic motivation, which fits the law of readiness.

Games use instant feedbacks. They award the players from time to time on engagement. The instant feedbacks (especially positive ones) conforms the law of effect.

The panelists of National Summit on Education Games [Summit on Educational Games: Harnessing the power of video games for learning] also points out that, educational games and simulations works better in developing higher-order skills than rote memorization. The higher-order skills, including strategies, decisions and problem solving, are barely tested by traditional tests. As a result, the power of educational games is underestimated by the current test mechanism.

3.5 Tangential learning

Klopfer and Osterweil, in their article on *the boom and bust of educational games* have pointed out that, some educational games encounter failures, because they just try to blindly present the academic knowledge to the play, but without thinking of how a game should be made [15]. Papert has even claimed that, those games inherited the worst feature of their parents [14]. Kerawalla & Crook explain that those serious games cannot be compared with traditional education in the sense of educational efficiency, and they cannot be compared with Commercial Off-the-shelf games in the sense of attractiveness [14]. The problem with these games is that they try to exploit rote memorization from repetitive practice (drill-and-practice exercise, as they put forward) to meet up with their academic goal, but with little care about how a game should be made [15]. In this sense, these games are more gamification than serious games (as they are intrinsically not games). They are nothing but educational software with some elements of games added. On the other hand, featuring rote memorization means these games work badly on the courses that require students' understanding and deduction.

James Portnow and Daniel Floyd have also mentioned in their video lesson series Extra Credits [16] that, the current development of educational games and entertainment games have diverged their ways. While they are supposed to meet together to deliver a product that is both resourceful and fun, they just each pursue their own value without caring about the other part. The entertainment games usually ignore its opportunity to put educational elements inside; while many educational games still seems "stupid" in trying to blend the knowledge and game mechanism. Habgood and Ainsworth has pointed out that, some educational game makers break the attractive gameplay into pieces, inserting mandatory learning sections between them. As a result, the players have to pass the learning phase, before continue with the gameplay [14].

The problem here is that, those educational games bring out knowledge in an intrusive way. Think of the situation when you enjoy the game's mechanism when you are fighting an ogre, then a difficult math problem pops up. You have to solve the math problem before you can attack the ogre and continue with your adventure. You will undoubtedly feel angry for being pulling out of the fight (this is because the flow has been interrupted. The flow will be talked about in detail later in section 4.1). You will also feel bored for having to face the math problem (probably, again). The player will feel like been tricked. It's just like one spends a large sum of money buying an expensive gem, which turns out to be normal glass. This is just an intrigue we used tricking the baby to eat vegetables, not any more suited for grow-ups. In this case, the students still dislike math. The only thing the game provides is just a promise, a reward to the student for his engagement in the study. Unfortunately, similar effect can be achieved just by saying to the student "when you have finished your math homework, you can play an entertainment game for a while". In addition to the unfavorable emotional effects, when the player switch between the ogre's fight and math, the player's brain suffers a switch cost [17]. This means that, whenever the player switch from the fight to math, he/she needs some time before setting up the mind to solve the problem; whenever the player switch from math to the fight, he/she also needs some time before recalling what was going on. The switching cost is so annoying that, there are already some employees in the company who try to avoid multi-tasking, just to reduce the switching cost between different tasks.

This problem with these educational games is substantially because there is totally no relation between the fight with ogre and the math. Therefore, the blend turns out to be a stupid mixture, like a mixture of water and oil. As a result, the students would still pursue entertainment games over the educational games for fun, and they may even prefer traditional study materials over the game for knowledge. The former is easy to understand, but the latter is more subtle.

The traditional materials are knowledge intensive, as they are made to convey knowledge; educational games however, have sacrificed knowledge density in order to add the "fun" part. As a consequence, when the students seek knowledge, they have to spend more time on the game than from the textbook, in order to obtain the same amount of knowledge. Thereby, a poorly made educational game could be disliked by the field of entertainment and education alike.

Here is when tangential learning comes into play. The term "tangential learning" was firstly introduced by Daniel Floyd and James Portnow on a video lesson series called Extra Credits [18] [16]. The key aspect of tangential learning is that, it does not force learning, but rather, to enable learning and to facilitate learning. It emphasizes more on nurturing interest than directly delivering knowledge. It has already been widely acknowledged that, people learn and do badly in topics that they are bored with, while pretty well in topics that they like (which also conforms to the principle of readiness 2.1). If we can make a learner interested in a topic, the learner will be found with great passion and great concentration, and the efficiency in studying (measured in knowledge learnt per unit of time) will greatly improve. What's more, being willing to engage the topic means that, diligence is not a problem anymore. The learner will spontaneously spend more time on the topic (time on task). This conforms to the principle of exercise 2.1. Finally, the final effect of learning, as the product of efficiency and time dedicated to the subject, will greatly increase.

Sugata Mitra, a professor at Newcastle university carried out the "Hole in the wall" experiment in 1999. This experiment followed the methodology called "minimally invasive education". Similar to tangential learning, "minimally invasive education" also concentrates on fostering curiosity and interest in the students mind. The experiment has proven that, children can be teach themselves, even withou the assistance from the adults/teachers. This reaffirms the possibility of minimal interference as well as the contribution of interest to the effect of study.

> If children have interests then education happens.
>
> — Arthur C. Clarke [19]

From Portnow & Floyd's tangential learning concept for learning contents in videogames [20], we can see four rules expected to be followed when a game was developed targeting tangential learning:

- The videogame still targets primarily on entertainment
- The references to the academic knowledge should not scare the players off by being a challenge to the player
- The game itself does not demand learning
- The references to the academic knowledge should be all accessible to the players

Point 1 stresses that a serious game should be primarily a game. It's obviously against the methodology followed by those "failed" serious games mentioned above. Point 2 and 3 underlines minimal intrusion, just as Sugata Mitra has proponed. Point 4 is actually based on Fogg behavior model (see section 4.4.2). According to Fogg behavior model, the more easily accessible one functionality is, the more likely it will be used by the audience. For tangential learning, its references play primary role in knowledge delivery. So it's intuitive that the references should be accessible.

Tangential learning embraces active learning (see 2.3) and student-centered learning (or called learner centered learning). It does not trying to directly teach the students with knowledge,

which is the common practice of traditional education (or sometimes dubbed as teacher centered education). In tangential learning, people are exposed to the knowledge, but they are not forced to learn it. The knowledge, however, is closely related to the gameplay that the player is engaged with. As a consequence, the game nourishes player's curiosity and interested in the related knowledge, which is the 4th driving force for the students to study, as mentioned in section 2.2.

Another benefit of tangential learning is continuous pursuit for knowledge. Many people cease learning the day they graduate from schools or universities. They take study as terrible memory where they were compelled [13]. Hardly have they understood that learning is a lifelong activity, and that they should always be ready to learn new things and constantly improving themselves. For this aspect of view, to teach someone just some specific knowledge makes only a limited use, possibly only up to the point when the exam is passed. The ancient Chinese saying says, giving one some fish is not as much as teaching him to fish. Similarly, teaching one some specific knowledge is not as much as encourage him/her to seek the knowledge. Tangential learning, planting a seed of interest and willingness to learn, has a better potential to lifelong study.

Yazan AlJilani and Rieko Kadobayashi have also highlighted some other aspects to be taken into consideration when talking about raising players interest [12]:

- The bigger role a person plays in the game, the more attention he will receive from the player.
- A playable character receives more attention than an NPC.
- The more important an item is in the game, the more attention it receives
- The reading materials in the game should be brief and short, not intended to teach, but rather, to enrich the experience and to nourish curiosity

It is not difficult to understand that an important character or an important item receives more attention. It's human nature to care more about genius person or useful item. This phenomenon is also witnessed in the field of films or animations, where the leading role is potentially more preferred than the supporting role. This is even strengthened, as the leading role is designed as the strongest or at least one of the strongest one (the enemy boss may be the strongest one at the start, but the major character is to defeat the boss, and become the strongest).

The major reason why students get uninterested in the topic and always forget what they have learnt is that, the students subconsciously consider the knowledge irrelevant to them, and thus the knowledge seems useless. This situation is even more for those theoretical and abstract courses, where the students cannot see solid objects or do experiment by their own hands. Contrarily, the knowledge with connection to the learner's previous experience means much more to the learner, which is learnt faster and retain longer in the learner's memory [21]. For example, students may not find it meaningful when studying the history of ancient Rome Empire. They may consider it just another strange name from the textbook, which has nothing to do with them, and finally get really absent-minded and sleepy. Contrarily, if they have already paid a visit to the city of Rome, have already watched a video on ancient Rome, or have played a game with selectable Roman Empire, things will be totally different. They will immediately recall the ruins and relics they saw with their own eyes, the picture of the Civilization presented in the video, or even the empire in the game with its special national ability. Further study and extensive study now means a lot, as they are all related to the memory already in the learner's mind. As a matter of fact, this is what meaningful learning requires us of. The ancient Chinese saying "It's better to travel ten thousand miles than to read ten thousand books" is to emphasize on the value of students' own

experience, that only to see with one's own eyes and to do with one's own hands, can one really understand the knowledge.

An in-game cyclopedia is a nice method to combine gaming and learning, creating a source of knowledge that is quite accessible to the players. Examples of this include *Civilizations*, *Romance of the Three Kingdoms 11*, *Zero Escape* and *Age of Empires*. Note that, the cyclopedia should have an entry point to the player, when they game is just paused. If the player has to quit the current gameplay, and open the cyclopedia only from the main menu, it turns out not that much accessible. As a result of this little complexity, the player will access the cyclopedia less, compromising its value.

One can also use external sources of knowledge like Wikipedia, which is as much professional as the tutor expect. However, one should note that, the farer the source of knowledge is from the game, the less possible players are going to use it. In-game cyclopedia is space limited, but player needs just a pause to read it. External material is filled with tons of authenticated knowledge, but players are less willing to switch out of the game to the browser. They may even be frightened away seeing pages of textual description. N.B. In a multiplayer game, the pause functionality may be disabled. This makes it unlikely that the players will have time to read the materials.

It has been demonstrated in an analysis that tangential learning could contribute to a large extent to the learning of knowledge [14]. Whereas, currently tangential learning still encounters some difficulties and limitations [20]:

1. Cost-efficiency. This is actually a problem faced by all serious games. Traditional textbook or lectures are quite knowledge intensive, when thinking of the effort to create the materials or to deliver the lecture. Explanation with language is after all, easier and quicker. On the contrary, to deliver the same amount of knowledge to the player, the creation of educational games needs more effort. After all, a whole mechanism, fancy art and other stuffs are also required.

2. Range of Knowledge. Currently, good examples of games with tangential learning are mainly strategic game on historical simulation. More games involving other knowledge are also required.

3. Distinguish between real knowledge and fiction. All games, even the history-based games needs adaptation, maybe to simplify, maybe to balance between different characters/factions, maybe just to make the game more interesting to play. There should be a way in which the player will be able to know which thing or property has real life reference, and which one is just a fictional creation just for the game.

4. Probabilistic effectiveness. As shown above, tangential learning puts no requirement on the play to learn the knowledge in the game. It just leaves the knowledge there, handy. On the good side, the player not is forced to learn, with reluctance; on the bad side, it's still possible that the player just does not get curious and just does not learn. Therefore, the success of knowledge delivery is probabilistic.

Therefore, the author's suggestion is that, tangential learning to be used together with traditional education, possibly a little earlier than the lectures (to be used in informal context, as some others put to it). When the students' interest are raised, they are supposed to learn much better also in class. Class following gameplay, on the other hand, clarified the knowledge points, eliminating the problem of "distinguishing between real and fiction".

If reversely a pedagogic serious game is to be a standalone work (called as in formal context), and it's required to guarantee its effectiveness, it's not a good idea to strictly follow every single

point of tangential learning. e.g. One have to relax the constraint of "The game itself does not demand learning". Nevertheless, the general concept and other guidelines of tangential learning can still contribute to this type of serious game. e.g. One should never blend the gameplay with knowledge clumsily and abruptly; non-compulsory knowledge can be still delivered with tangential learning.

3.6 Some serious games on the market

US military released a first person shooting multiplayer game called America's Army (see https://www.americasarmy.com/) in 2002. The game was the first trial of US government to use game for recruitment, and it turns out to be quite effective. Apart from being a recruitment tool, the game also targets recruit training and simulation [27] [28], but it has then become one of the most popular online entertainment games, with 13 million players registered. Its virtual medic training course, which is based on real training for soldiers, has been proven contributing to life-saving situations [33].

The Tactical Language Training System (currently containing *Tactical Levantine Arabic* and *Tactical Iraqi*) was funded by DARPA and developed by Center for Advanced Research in Technology for Education, to teach foreign language and culture. In the game, the player encounters virtual characters and interact with them thanks to the speech recognition technology [27] [34].

Gamelearn [35] claims to be the first retailer of online game-based learning. It creates serious games for corporate training, especially managerial skills for more than 1000 companies from more than 50 countries including LG, Burger King and Coca Cola. The target skills include communication, coaching, conflict management, leadership, teamwork, time management and many others. The serious games can be also integrated with user's LMS (Learning Management System) for better monitoring and management of the learning process. In its serious game course *merchant*, the player becomes incarnate as a merchant in 15 century Venice. The player has to learn to build trust between the client, to resolve a conflict, to obtain information at sensitive encountering, or to bargain and negotiate. In its serious game *pacific*, the player is trapped in a desert island with some other persons after a plane crash. The player has to learn of leadership and team management, in order to amass the power of the team to escape from such a pretty pass. The player is to learn of listening to feedbacks, making critical decisions, improving effectiveness, setting rules, establishing personal and team level objectives, nurturing mutual understanding, detecting individual strengths and weaknesses, etc. In its serious game *Triskelion*, the player The player has to learn goal setting and the way to keep with the goal, deal with email messages, to manage the limited amount of time, to balance among different tasks, to act immediately, to manage stress level, and to be more productive.

Daesign is a French digital company. Starting from 1995, it creates cloud based serious games among conflict management, entrepreneurship, digital security, remote management, assessment and many others [36].

Breakaway or Breakaway games[37] is a video game developer founded in 1998 in the state of Maryland, USA, which initially worked with strategy entertainment games (e.g. *Sid Meier's Antietam* and *Sid Meier's Civilization III: Conquest*). After the 911 attack, Breakaway began to shift to serious games. Now it has already been considered one of the largest serious game developers, with many serious games for government, military, and commercial clients, health and charity organizations. AAI, Boeing, Booz Allen Hamilton and many others also have relationship with Breakaway. Breakaway's serious games cover emergency management, corporate training, healthcare simulation, military training, science education and social change. Its virtual clinic

learning game *pulse!!* for Texas A&M University at Corpus Christi has won 2008 Serious Games Showcase and Challenge Finalist [38]. Its educational game *perfect world*, made for Batimore Country Public School has won 2013 Serious Games Showcase and Challenge Finalist [39]. Its training game *Vision* made for Essilor, which trains undergraduate and graduate level business students of operation management, has won 2012 Serious Games Showcase and Challenge Finalist [40]. Its training game *Aflac Trivia* made for Aflac tests the player of details about Aflac insurance policies, which has won 2017 Serious Game Showcase and Challenge Finalist [41].

Ijsfontein [42] is a company located in Marine Land Amsterdam targeting playing and learning. Products include Security training Shell, a 3D animated films for Shell employees to recognize security risks; *HoloLens Experience*, a mixed reality product for the sale force of Mercedes-Benz Global Training to quickly get ideas about the newest models and technologies; *National Waterline Museum*, a interactive product including both software and wearable devices for museum visitors to experience with their own eyes The Dutch Waterline.

pixofun [43] is a gamification company located in San Francisco and founded in 2005 designing serious games for companies to train their employees. It produces pre-made Game as a Service and user customized games as well. It focuses corporation training and has cooperation with many companies like Coca Cola Icecek, KFC and Pizza hut.

Filament Games [44] is an educational game developer founded in 2005 in Madison of Wisconsin in USA. It specializes in accurately represent educational contents, as its games have clear references to American teaching standards. Tips of pre-game, in game, post-game and reentrant are also given, in order to maximize the teaching effectiveness of the game. Samples of its games include *Molecubes*, where the player is to solve the puzzle by changing a matter among different states, change its flammability or pH values; *You Make Me Sick!*, where the player learn of the knowledge of virus and bacteria, by using their strategy to customize a bacteria or virus, and infect the target host as much as possible; *Planet Mechanic*, where the player adjust the orbit distance, rotation rate, atmosphere and many other properties of an alien planet, learning of how the solar system affects the phenomena on the planet.

Pinduoduo is an e-commerce platform facilitating the users in group buying deals. It released a social game in 2018 called DuoduoGuoyuan, which is an advergame (game for advertising) on the mobile phone. A player of the game grows virtual fruit tree at his/her choice in the game, and when the tree fruits, some real-life fruits will be sent to the player. In order to grow the tree, a player need to gather water. The intriguing part is that, water is earned from the actions like inviting friends to the App, browsing or purchasing commodities, drawing a prize. The wisdom in the game is that, in the pursuit for the fruits, the players firmly remember Pinduoduo, actively distribute the App to their friends, and unintentionally or intentionally make purchase with Pinduoduo, which contributes to the commercial success of Pinduoduo.

3.7 Case study

Duolingo

Duolingo is a free language learning platform, including its web and app versions [22], and it has a digital language proficiency assessment exam with it. It provides learning courses for 31 languages (by April 2018) as self-tutoring tool for learners across the world.

The game falls more in gamification category rather than a real serious game category, for the fact that it's created adding the game elements into the normal e-learning process. It exploits the skill tree from the game world, indicating what a particular field where the language is to be used.

Examples of these skills include “family 1”, “location 4”, “health 2” (as can be guessed, achieving “family 1” means one can start the most basic conversation about family, while unlocking “health 2” means one can communicate in a little higher level when it involves health care). Players who has already some language capabilities or has improved their language skills out-of-band can also quickly level up, unlocking bunch of skills, just by successfully passing corresponding intensive tests (called tree shortcut).

Duolingo also let you set a daily goal for your progress (as shown in Figure 3.1). In a progress circle it shows your completeness of your daily goal, and in a chart it record your everyday advances. This feature, which is barely found in other serious games, brings a unique contribution in forming a good habit. According to operant conditioning (which will be elaborated in section 4.4.1), a positive reinforcement enforces on a behavior, and it helps the nurturing a habit. Habit, on the other side, dramatically guarantees the student to learn (as discussed in section 2.2). The learner, seeing the daily goal nearly met, the weekly progress recorded, and day the streak kept, gets more motivated to continue with the goal, until a habit of studying the language a bit more every day is formed.

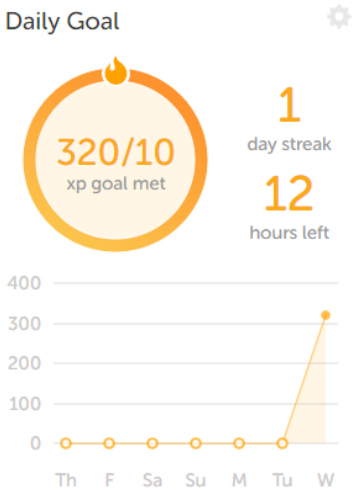

Figure 3.1. duolingo daily goal with weekly progress

There are also achievements to be acquired, for example, “sharpshooter” will be unlocked by not giving a wrong answer in a lesson or a practice. The use of achievement system clearly sets a goal for the learner, and also benefit from operant conditioning.

Duolingo has also a visual currency called lingot. Learners earn lingots by leveling up, finishing a skill or continuously keeping their plan. There are also optional “power-ups” like “Double or nothing”, which costs 5 lingots initially, but awards 10 when one keeps at the schedule for 7 days. There is also one power-up called “Streak Freeze”, which enables you to rest for a whole day without any penalty on the streak (continuously following with the plan). If the learner just rest, he breaks the streak; if he/she uses “Streak Freeze”, it cost him/her 10 lingot. The learner will surely think twice before any lazy day. Besides, again, according to operant conditioning, the penalty on lingots (whether with or without “Streak Freeze”) discourages the action of lazing. It contributes to good habits from the unfavorable side (positive punishment). Nevertheless, this feature of building a study habit is something missing in many serious games. Therefore, it’s advocated that, more serious games should take forming a good habit into consideration, which

is supposed to be a missing opportunity.

Levels, skill trees, awards, achievements and power-ups are all elements well known in the field of entertainment games. These elements set the aims for the player, keeping them to continue playing, or keeping them to always come back to the desk every day. Games have been criticized in using these elements, causing addiction. But when these are used to education, things can totally change.

Darfur is Dying

Darfur is Dying is a flash-based game released in April 2006, addressing common people caught in the crisis of Darfur in western Sudan. Especially, in the category of serious game, it's considered a news game. This work is created in order to raise people's awareness of the genocide that has actually happened in the real world. During the game, the player has to firstly choose one avatar from the members of the Darfur family and then, try to survive in the conflict, while also maintaining the refugee camp supplied for 7 days.

There are two scenes in the game. In the desert scene, the player has to forage for water, which is crucial resource to be used in the second, refugee camp scene. The desert scene, however, is full of danger. The player has to learn to, all along the way, hide among the bushes from Janjaweed militia, who is backed by the Sudanese government. If the player succeeds in fetching water in one piece, the camp's water storage is replenished. However, if the avatar is unfortunately caught by Janjaweed militia, a message tells what abuse or slaughtering the victim may face, as what has happened to the refugees in the crisis in Sudan. After that, the player can only continue with another member from the Darfur family, for the previous avatar is removed. This reflects the fact that that member of the family is probably lost forever.

In the refugee camp scene, the player is safer. The player can use the foraged water to make bricks and repair shelters, or to cultivate farms in order to fill the belly. The avatar can't stay in this safe place forever, though, as the camp will soon run out of your water storage. The player has to react quickly also, because as the time passes, the food and water are consumed, and the Janjaweed comes to raid your resources from time to time.

In a large number of games, the players work and earn, becoming stronger and plentiful, making the life easier and better. Darfur's Dying however, does the opposite. The players just mend and repair, feed hungry belly and dodge assaults. It's about keeping what little they had for as long as possible. What's worse, the constant rampaging Janjaweed militia raiding is unavoidable and uncontrollable, bringing the players sense of powerless and despair, which serve the author's purpose well. The stress that one may live another day even more wretched and barely feed, enforces a great sense of unsteady and fear, which, the players living in peace and in prosper, seldom knows of. This is exactly the power of a medium, the way to evoke emotional echo in the player's heart. And emotional echo is one of the reasons that the game was created for.

It also worth noticing that, listed among the in game prompts to teaching how to play, there is also "take action now" button (as shown in Figure 3.2). The button leads to a different page where the players are implied to take real life actions in order to stop the crisis. This is coherent with Fogg Behavior Model 4.4.2, in the form of a facilitator. The gameplay get the players emotionally touched (motivation), and the prompt "take action now" brings player to the action within a click. An in-game link is just at the right time.

Nevertheless, there are people who blame this game for oversimplify the actual situation or being ineffective [23] to the real crisis. Nevertheless, the resolution of a crisis involves many parties

Figure 3.2. page shown after clicking on "take action now"

including governments, militaries, peace keepers, and it requires many people's collaboration and dedication. One cannot expect a flash-based game alone to be the panacea to the whole problem. So long as the game plants a seed of compassion and a sense of responsibility in the players' heart, which may one day blossom in their later lives, the work is worthy.

Chapter 4

Theories and techniques from entertainment games

Playing has always been a serious issue. Animals develop a fond of playing more in their young age. This is also the stage they learn to adapt to the environment. Moreover, they learn fastest during this stage. This superposition is not a coincidence, for playing IS closely related to learning. As a matter of fact, the play that young animals do, whether alone or with their peers, is usually mimic or simulation to the adult life. They get physically trained, they learn hunting, they learn of the fear of danger, they learn socializing with other companions. They practice and learn all of these, in a non-failure-critical environment, so that they will be prepared when they do have to cope with the situation seriously the future. Similarly, in human society, it's widely known that the childhood shape a person greatly, whether in personality, in desire and motivation, as well as in the form of a home environment. Children learn, to quickly adapt to the ever changing environment. From this aspect of view, the education given to the children that tends to fight against playing or the fond of playing (e.g. an overemphasis on diligence and hardworking) could be on the wrong track, which is somewhat against human (or even say, animal's) nature. Education can do better.

Game design, on the other hand, is the theory to cater to this nature. The entertainment game industry, as boosted by many companies for decades, has accumulated a big amount of knowledge in the form of theories and techniques for the game design. The commercial success of many of them has undoubtedly demonstrated the effectiveness of those techniques in attracting players. It's only that, with so much knowledge in hand, they seldom use that for more significant purposes (e.g. education). They just missed a great opportunity [16].

Serious games, sitting in between educational software and games, find it an excellent opportunity to take advantages of those discoveries in developing games. This chapter will focus on some theories and techniques already applied by the entertainment game industry, but are still infrequently addressed by serious game developers. Not surprisingly, the reader will witness many similarities between game design techniques and learning principles.

4.1 Flow

Flow is probably one of the most commonly discussed words in the field of game design. Nevertheless, it was primarily introduced in psychology, which is closely connected to the term of

hyperfocus. for the first definition of flow was given by Mihaly Csikszentmihalyi in 1975 as "FLOW is a state of concentration so focused that it amounts to total absorption in an activity" [45]. Though not named the same, the concept of flow has merged far before Csikszentmihalyi, among different cultures across the world, especially in eastern religions. In those cultures, practicers are instructed to dedicate all their attentions to one particular thing, leaving all other trivial matters off their minds.

Concentration means, the person has his/her senses all dedicated to the specific thing, and his/her brain also running in full power. This leads to the best productivity (if the person is creating something new) and the best learning effectiveness (if the person is learning). In this state, the person sees no other thing, and hears no other thing. It's as if the person, with all his/her body and soul, has escaped the immediate surroundings, and is totally placed in the engaged topic. Flow is powered by intrinsic motivation [24]. Flow is exclusive, so that he/she will never get weary, sleepy, distracted, and he/she turns a blind eye to scorching heat or the belly's call, until the physical limit (or until the flow is broken). This has been found on engrossed scientist and artist as well as on playful children or on gamblers. Even if flow is finally interrupted e.g. by a really big noise or pain, the person turns quite resistant to get away from his/her current work, to engage incoming change. It's just like a fatigued person waken up by an alarm clock, not willing to get out of bed.

There are many factors affecting the state of flow [24] [13]:

1. Clear goal
2. Immediate feedback
3. Matched perceived challenge and perceived capacity
4. Unawareness of the time
5. Loss of self-consciousness
6. Unawareness of other needs
7. Autotelic activity
8. Great concentration
9. Actions integrated into awareness
10. Feeling of the potential to success
11. Sense of control

Autotelic means that the activity pursues no other thing; just experiencing itself is the ultimate goal. The first three are considered also as the conditions necessary for achieving flow. It's even argued that, the third factor (the match) actually implies the first two, in the sense that, one need to have clear goal and see immediate feedback on it in order to perceive the challenge matching the skill.

Therefore, a basic graph of the challenge and capacity inducing flow is as Figure 4.1:

In 1987, Csikszentmihalyi, together with Massimini and Carli, published a newer, 8-channel version of graph for the flow state, which is also named as, the Experience Fluctuation Model, presented as Figure 4.2:

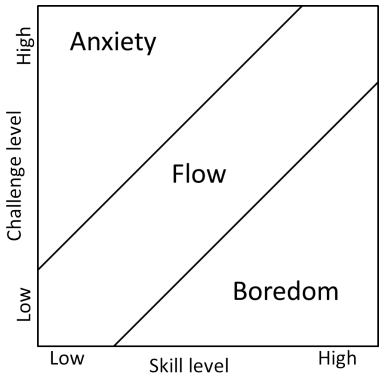

Figure 4.1. 3-channel version of challenge vs skill

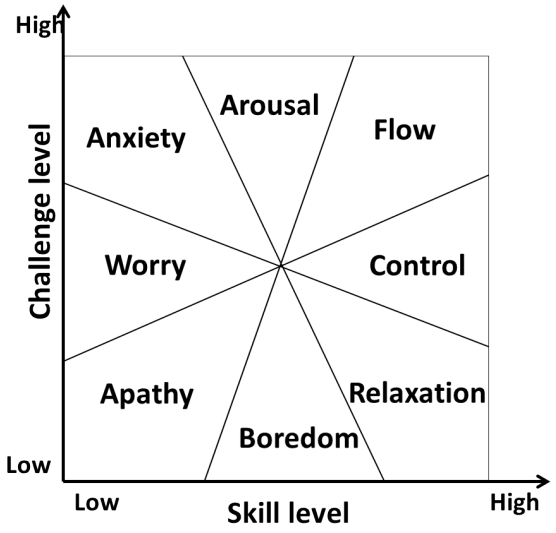

Figure 4.2. 8-channel version of challenge vs skill

Here, it worth noticing that, when both the challenge level and capacity level are low, the individual is in an apathy state, which works far less effective that flow. This indicates that, one need to have a minimum level of skill, doing a challengeable-enough activity, will one be able to really enter the flow, and enjoy the activity. Take drawing as an example. When someone just start draw for the first time, he probably choose to draw some simple geometric shapes. He may find it interesting for a while, but after some time, the interest strains, and he can be distracted by other things easily. Only until the point that he has constantly polished his skills, and targeting complex pictures, will he really be able to full immense in drawing, spending hours and hours on it, without feeling any hunger or tiredness.

The state of control is also favorable, for the individual can easily enter the flow state by just increasing a little the challenge. Similarly, the state of arousal works well, as the individual may learn a skill in the process, increasing his/her skill level, and she/he enters the flow state. In the

arousal state and flow state, the individual is activated, and ready to act. The heart rate rises, the eyes widen, the breath quickens, the attention concentrated and the level of adrenaline increased. This usually happens to an athlete, when the starting gun is about to fire.

Flow lies in the region where the challenge matches the player's skill level. When the challenge is too high, the player's brain is overloaded, and he/she cannot catch up, and will soon get exhausted. When the challenge is too low, the player's brain come to a pause, with nothing to do, and he/she feels bored.

Csikszentmihalyi describes the flow state as "Optimal experience", in the sense that, in the flow state, a person experiences great satisfaction (actually, exhilaration, as he put to it)[13].

4.2 Decisions and skills

[45] Flow is volatile, however. The brain power is concentrated on the topic issue, which means, when that particular issue has been processed, the brain comes back to a rest, and the flow breaks. On the other hand, if a brain is overburdened by the things to process, it feels exhausted, and unable to catch up. The brain will sooner or later quit from the topic, and this also causes the break of flow. Therefore, in order to maintain the flow state, a game needs to continuously feed the player with things to process (which is decision) in a pace that matches the speed of processing.

When it comes to pedagogic serious game, instead of the single word "decision", "decision and deduction" is preferred. Normal entertainment games uses the term decision, because they, especially non-strategic ones, does not require that much of careful and deep thinking. Here decision and deduction is used, to emphasize the fact that, in pedagogic serious games, players are more required to think, analysis, calculate, or deduce. This involves the recalling and application of academic knowledge, which usually takes much longer time.

4.2.1 Decision scope

Different decision requires different amount of deduction, and take different amount of time to make. Decision scope is just its measurement.

The quickest decision requires no thinking. People may actually not consider it as a decision. It includes innate behaviors as well as actions that we are so much used to. When you want to eat a cake in front of you, reaching out your hand is an example of instant decision. This decision can be performed just by habits, without any involvement of the human brain. Therefore, no matter how many instant decisions are there in an activity, flow will not occur. There is a bad example of using countless instant decisions in game. In this game, players are relieved of the responsibility of controlling its character. Instead, the character can take missions, fight around, picking up loots, all automatically. The only think that really requires the player's involvement is to click on "ok", when a mission is completed, when a reward is granted, or when character levels up. The character does level up quickly (compared to normal games), and the player is indeed fed with constant decisions (clicking on "ok"), but the game experience is terrible.

Quick decisions are made in a short amount of time, usually no more than one second. Actions games, shooting games feature this kind of decision. The player's eyes widened, heart beat doubled, attention concentrated, in order to react faster and more accurately to the upcoming events. Athlete start running at the sound of a starting gun is a quick decision (expected event, not knowing exactly when); but it is also a quick decision to stop attack, but to dodge at the sight of the rival's attack (potentially unexpected event). Flow can occur in this kind of decision.

There are also more complex decisions that require seconds or even minutes to make. When making these decisions, the brain searches across its memory, trying to match the pattern with previous knowledge and experience. The brain could also perform compound deductions and computations in the process of making decision. Example of these decisions includes evaluating whether a product has a good cost performance or thinking whether the man talking on the phone is a fraud. It should be noted that those consideration involving academic knowledge is among this category of decision. Flow can also occur in this kind of decision.

Nevertheless, if a decision requires more knowledge than what the person has, or the decision need so much consideration that overloads the person's brain, the decision cannot be effectively made. In this case, the person just stops thinking, simplifying the deduction process into a most basic guess, in which case, the flow does not occur.

4.2.2 Skill level

Skill level is a term used to describe the player. Each person has a different level of skill in performing each action. An experienced cyclist can pedal a bicycle at high speed, while also give regards to the spectator; a starter on the other hand, has to use all that he can just to maintain the balance of the bike. When a person learns of something and becomes experienced, the old memory is saved in the brain, and the neural pathway has grown. In the future, fewer brain cells are needed for the task. Moreover, it will also take less time for the person to recall such piece of information or to make such decision. If a game system feed the player with decisions or deductions in a certain speed, the expert and the untrained will feel greatly different. The expert's brain processes the decision or deduction in a flash, and then come to a stop, which breaks the flow state, leaving the person void and bored. On the contrary, the untrained brain processes slowly and with the majority of brain cells involved, but is still not enough to catch up with the time, until finally being flooded with too much decisions and deductions. So decision and skill level gives a good explanation of the flow theory.

Yet, the skill level for a person on a certain task is not fixed either. Through drill and practice, a person learns, and becomes more practiced with that task. As a result, what was considered a compound decision may in a later day becomes a quick decision. This is one of the reasons why we played our toys with joy at childhood, while feeling bored and uninterested as adult. As the decision becomes easier and quicker, a person seeks more challenges, with more difficult decisions. For a game that is not designed to be played only once or twice, it will witness a dramatic grows in the skill level of its players. If the game is not able to adjust itself to the change of the player, it will not be able to continue inducing flow, and it will follow the step of the childhood toys. This is the reason why there are some games which only appeal interesting for the first several playing through, which are thrown away when the player has full explored every possible strategy. Players are always on the way to be more skilled, but it's the process of seeking, rather than the result of being more skilled, that interest the most.

4.2.3 Skill range

Skill range is a term to describe a game. It is the range of skills in which the player perceives matching challenge. It's bounded by the skill floor and skill ceiling.

Skill floor (or say skill barrier) describes the minimum level of skill needed for a person to start enjoy the game. An example of relatively higher skill floor is the first day of skiing without any preparation or guidance. The new learner only returns home with countless memory of falling on

the ground and a bruising body. She/he could think skiing as something terrible and will never retry. This is because the skier has not yet learnt basic balancing on the snow, and clearly has not passed the skill floor of ski. Games have skill barriers two. The new learner has to firstly endure an unpleasant experience before reaching the minimum requirement on the level of skill. Therefore, the game designers are usually expected to create an artifact with low skill floor (this property is also called as accessible). For example, DOTA (Warcraft III version), the first widely acknowledged MOBA game has more than 100 characters to choose from and a complex equipment synthesis system. The players need to remember some hero skills and remember many synthesis formulas in order to effectively fight or synthesis equipment in the game, which drives many new comers off (despite its popularity in those years). DOTA2 learnt from this, and provided a more user-friendly synthesis tree, together with recommendations for equipment, which greatly relieved the new comers. The field of education is also aware of the presence of skill floor. They set for each course prerequisites, and not offering the course to those who has not met the prerequisites. If someone who doesn't know multiplication try to engage calculus, the result is obvious.

Skill ceiling (or say skill cap) indicates the maximum level of skill that the player can possess, with which the player overcomes the challenge in the game. When a player's skill level reaches the skill ceiling, there is no more room for any improvements. The player has fully understood the details of the game, with all the strategies that can be used, and can react as fast and accurate as the mechanism allows. A chance to perform better has always been a driving force for people to engage something. On the other side, players tends to feel bored and leave the game, when there is no more room for improvement. Thereby, a game with high skill ceiling (also called a deep game, or a game with depth) is more favored, as it can keep its players for a long time. The extreme case is that, if a game has its skill ceiling beyond the human's power, no player can reach the skill ceiling. Consequently, its players always have something to learn from the game. For example, the board game Go and chess were considered games with infinite depth. Even spending all his life studying the game, a player will not reach the skill ceiling. This is also the reason why alphaGo have the possibility to defeat the best human player, and the reason why alphaZero can still defeat alphaGo. Whereas, different from entertainment games which aiming at maintaining its players, pedagogic serious games allows their players to leave when the intended knowledge was successfully delivered. Therefore, one does not that much have to pursue a pedagogic serious game with great depth.

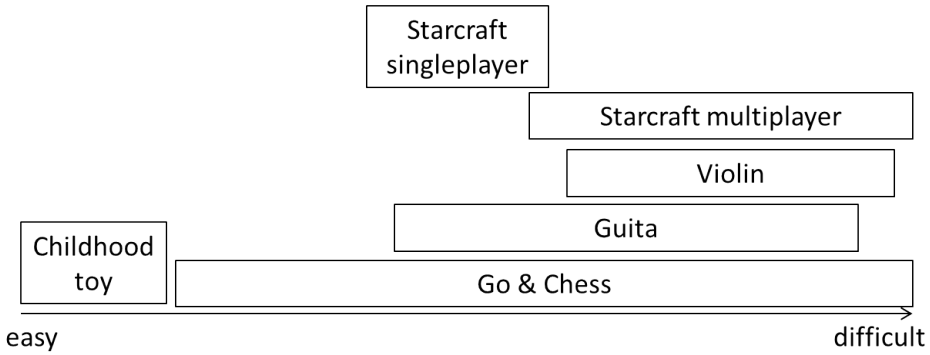

Figure 4.3. the skill ranges of some activities

It can be found that, an multiplayer version of the game is usually more challenging than the single player version. This is because competitive multiplayer game involves psychological challenge between the two human players. Although there are some activities with both skill

floor and skill ceiling low (e.g. playing with childhood toys) and some activities with both high (e.g. playing the violin), something with a low skill floor and high skill ceiling is possible. The board game Go, as mentioned above is a good example. Newcomers need only to learn a most basic rule to start. She/he will gradually learn capturing tactics, reading ahead and many other strategies during his playing, but she/he will never reach its limit. Activities with low skill floor and high skill ceiling are always preferred (fit people of both the Main Street and ivory tower, both newcomers and masters).

Elastic challenge

There is an interesting term called elastic challenge here. A game with only one definitive win and one definitive lose (e.g. only hit or miss) have a very narrow skill range. Players with lower level of skill (definitively) fail the challenge; players with moderate level of skill (definitively) succeed in the challenge; players with higher level of skill succeed with no sweat, feeling bored and unchallenged. Thereby, the majority of the game designs use elastic challenge. Elastic challenge provides different level of success and failures. For example, the player has a HP (Health Point), they success by defeating the enemies with their HP anywhere above zero. There could be also a grading system, where the player wins by achieving any scores above a threshold, and fails by achieving any scores below the threshold. This means, the expert wins in the game just like the skilled, but he/she can still pursue better performance, e.g. a higher score. On the other hand, the one almost succeeded are much more motivated to retry than the one with absolute failure. This concept of elastic challenge is so widely used in game design that people take it for granted. Nevertheless, by knowing of the knowledge of elastic challenge, one can apply it further, to even more extend the skill range to match players of different skill levels. For example, adding a achievement system where the expert can pursue additional titles for his excellent performance as he pass through the levels; giving scores for each piece of gold earn and each friend made, even if these does not define the final victory; providing a difficulty selection for the player, where "easy" leads to unlimited number of retrials, and "hardcore" leans to game over at the first death.

Difficulty change

It's discussed before, that challenge needs to meet the level of skills in order to induce flow, and that different player has different levels of skills with different task in different time. Thereby, a game should be able to adjust its difficulty with respect to the player dynamically.

The easiest and most frequently seen approach is to provide explicit difficulty selection screen at the start of the game, especially, easy, normal or difficult.

This approach however, runs the risk of having done a wrong estimation and thus a wrong choice. Thereby, here comes another approach: adaptive difficulty. Adaptive difficulty changes the difficulty according to how well the player has performed inside the game. If the player did really badly, the system silently provide a bonus; if the player did very well, the system silently generate a challenge. An example is the bounty for killing in DOTA. If a player is quite experienced, she/he can defeat multiple rivals in a short amount of time, receiving multiple bounties and multiple exp points, which could lead to higher level than the rival. Nevertheless, when she/he is killed, the game mechanism will grant the killer with extra bounty, based on the victim's level as well as streak. As a consequence, the disadvantaged side gets a bonus in bounty. The advantaged side gets also a penalty, in the sense that the higher level character loses more money and waits longer before revival.

Failure trap Failure trap is a state, in which however the players perform, they are bound to fail. This happens to many sports or competitive games, in the sense that, when one side becomes dominant, the disadvantaged has totally no hope of winning. This creates a great sense of helplessness among the disadvantaged, but they still have to endure even longer, until the point when the game really ends.

One solution to this is to use adaptive challenge to help the disadvantaged, saving them from the state of "hopeless" to "still with a glimmer of hope". It could be a hint message or a material assistance to the player. It' can also provide optional paths to get around failure trap. For example, the player needs to collect 3 units of gold or 4 units of silver to win; or the player is required to identify 4 out of 6 matches to win. If the player is stuck on one path, she/he can just retry with the other, may be it will be successful.

The other approach is to terminate the game once and for all. There is of course no problem if a single-play player wants to quit a game immediately when she/he performed terribly. But things will be different when multiple players are working together to a common goal. Some may have already viewed the game as hell, while some others still want a last struggle. To deal with this issue, a surrender mechanism is created, which terminate the game with defeat, when all the players in the group vote for surrender. If anyone of them want a last fight, then everyone need to still work together for the last hope.

Nevertheless, one failure trap needs special attention here: the failure trap due to insufficient information. For example, a player is stuck inside a room, not knowing how to escape. What was expected of him was to push the box to the wall, step on the box, jump and climb up the wall to exit. Yet, there is no clue that the box can be pushed, and the player just continuously searching around the door firmly locked. In this situation, a hint message should be provided to the player, saving the player from this dead lock, especially when the player has been found trapped here for minutes.

4.3 Tutorial

A more commonly seen practice to lower the skill barrier is to create a tutorial session. Although the skill floor of the formal game remains the same, tutorial itself gets a really low skill floor, low enough to accept most of the players. The player learns control with constant guidance, quickly obtaining the expected skill level.

Perfect as the tutorial should be in theory, however, it encounters problems in reality. It even happens that some players, even knowing the existence of tutorial, deliberately skip the tutorial. They'd rather grope in the formal gameplay than walk through the tutorial section. To address this problem, James Portnow and Daniel Floyd have listed in Extra Credits some points for the game designers to take into consideration [26]:

- Similar the formal game experience. Many tutorials are made at the last step, when the game is nearly finished. Game creators have already dedicated great care and effort for the formal gameplay and probably also the CG, making them as fancy as possible, but they get impatient with the tutorial. A commonly seen quick practice is to provide only a piece of text as tutorial. Pure text is extremely easy and fast to make, but is really unattractive, no wonder why some players used to skip the tutorial. The player will feel even more resistant to read if the text is of a big chunk. That's why sometimes the player are found just keep clicking on "ok" buttons, just to get through a compulsory tutorial, without really have read the guiding text. One possible relief to this is to add enough graphical elements. It has already been known in the field of education that adding infographic enhances understanding and brings attraction, so why not here? An even better approach is to create an interactive scenario, possibly similar to the formal gameplay. It has been discussed in previous chapters that lab sessions or say put-into-practice sessions following the theoretical talk strengthens learning, so why not here?

- Guidance coherent with the game fiction. There are currently three ways to instruct the players: through prompts, through in game characters and through indirect control. Instruction through prompts is the most direct approach. Game creators tell explicitly to the player with on screen words on how to control (e.g. "press 'w' to move forward"). This however, interrupts the flow. A better (and vastly adopted) approach is to let the in-game characters say in place of the game creators. If the story of this tutorial is crafted so that it's coherent to the learning topic, the player can still remain in flow, in the fiction. For example, in the historical simulation game Romance of the Three Kingdoms 11, there is a tutorial section on defense. The scenario setting is based on the real history when your controlling faction is under attack. Your newly acquired military counselor has shown his excellent military talent and drove the enemy away. In the game, the counselor instructs the player, just like he instructed his faction leader back that day. Instruction through indirect control however, is usually more complex to make, and more on decision making and on strategic level, so it's rarely found in tutorial where the players often learn basic controls. An effort to apply indirect control could be: instead of an on-screen prompt as "use the artillery to take down enemy buildings", create allied artillery in the player's sight, which is bombarding the enemy building.

- Teach gradually. To teaching everything intensively is also a bad practice, which happens more also when the game creators get impatient with tutorial. They pour to the player a flood of information, without thinking of whether they can digest. In the skill level part we have said that someone already skilled and familiar with a topic can deduce quickly, considering it not a big deal. New players, as one totally untrained and unfamiliar with the topic, will think and deduce much slower. Thereby, the gameplay creators need to keep it in mind that they tend to underestimate the burden on the player, so that they deliver a big tube of new information, beyond the player's power to digest. Some games introduce only most basic units at the first level, while gradually unlocking more units and skills in latter ones. That is a nice example of distributing learning burden across levels.

- Optional tutorial and multi-section tutorial. Another reason why those players skip tutorial is they believe that they have already acquired many skills in games other than this. They are not going to go through the tutorial section starting from click and select, just to mend some small missing points (recall that high skill and low challenge results in boredom). This feeling will be even stronger if the tutorial is compulsory. Non to say the situation when the player just changed his computer, reinstalled the game, seeing the boring tutorial coming again. Thereby, an optional tutorial is more preferred. Besides, if the tutorial is divided into multiple sections, the player can start at the tutorial that she/he think of use, and skip those first ones that teaches basic controls. Multi-section tutorial also avoids teaching too fast (point 3), allowing the player to learn only a specific topic in each section.

- Handy tutorial. This time it is not an independent tutorial session, but more of something that player can easily access during the formal gameplay. The players may not have comprehended all the things in the independent tutorial session, or they may have forgotten how to control. It's inconvenient that the players always quit the game and restart the tutorial session whenever they encounter a missing point. What the players actually need is, when the formal gameplay is paused, a quick refresh of what was taught in the independent tutorial session or just a hint to help passing the challenge (it will be also useful against failure trap, which will be detailed in section 4.2.3). Moreover, according to Fogg behavior model, the more easily accessible a functionality is, the more likely it will be used (see section 4.4.2). Therefore, a handy tutorial will take effect whenever the players need assistance.

There is also another approach that does not follow the above points exactly, which still works

very well. Blizzard has created Starcraft and Warcraft III with no independent entrance called tutorial. However, the early chapters of the campaign actually act as tutorial. In the very first chapter, little units are unlocked, and the game pace is really slow (point 3 above). There is also on screen prompts and dialogues guiding the player, just like the textual tutorial would do. Being a campaign, however, it is substantially a formal gameplay (point 1 above), allowing the player to instantly put guiding instructions into real practice.

Finally, whether a game is made for entertainment or for education, its tutorial section is always one that teaches something to the player. In this sense, the witness and discoveries in the design of tutorial can be of great enlightenment to the designer of educational games.

4.4 Motivation

4.4.1 Operant conditioning

B.F. Skinner's Operant conditioning is similar to well-unknown Pavlov's classical conditioning (or respondent conditioning). However, they are substantially different. Classical conditioning builds a connection from the stimuli (e.g. the bell' ringing) to the response (the dog's salivation), which is usually unintentional. The operant conditioning, however, builds a connection from the presence or absence stimuli (presence of favorable stimuli or absence of aversive stimuli) to the behavior (response), which is usually intentional.

Here comes the skinner's box. Skinner's box, or called as operant conditioning camber, is a box made by Skinner to train test subject inside to do a certain thing. When the test subject does the designated action, the box provides with reward such as food. An alternation is that, the box enforces punishment on the action. By experiencing with this box, skinner has found out that, operant conditioning can condition volition. With rewards or punishment, he can get the test subject making a certain choice. The finding is the following [46]:

Positive reinforcement When the designated behavior is followed by a favored stimulus, the chance that the behavior is repeated increases.

Negative reinforcement When the designated behavior is followed by a removal of unfavorable stimulus, the chance that the behavior is repeated decreases.

Positive punishment When the designated behavior is followed by an unfavorable stimulus, the chance that the behavior is repeated decreased.

Negative punishment When the designated behavior is followed by a removal of favored stimulus, the chance that the behavior is repeated decreases.

Extinction When the bond between the behavior and the stimulus vanishes, the chance that the behavior is repeated will gradually return to normal.

This can be illustrated with a graph as Figure 4.4.

The words "encourage", and "discourage" themselves, are intrinsically based on the theory of operant conditioning. Encourage rewards on the action, "encourages" the subject to repeat, and discourage penalizes on the action, "discourages" the subject to repeat, both orienting at impacting on the test subject's willingness.

It's also discovered that, the schedule of reinforcement also makes a difference to the effectiveness of operant conditioning [46]:

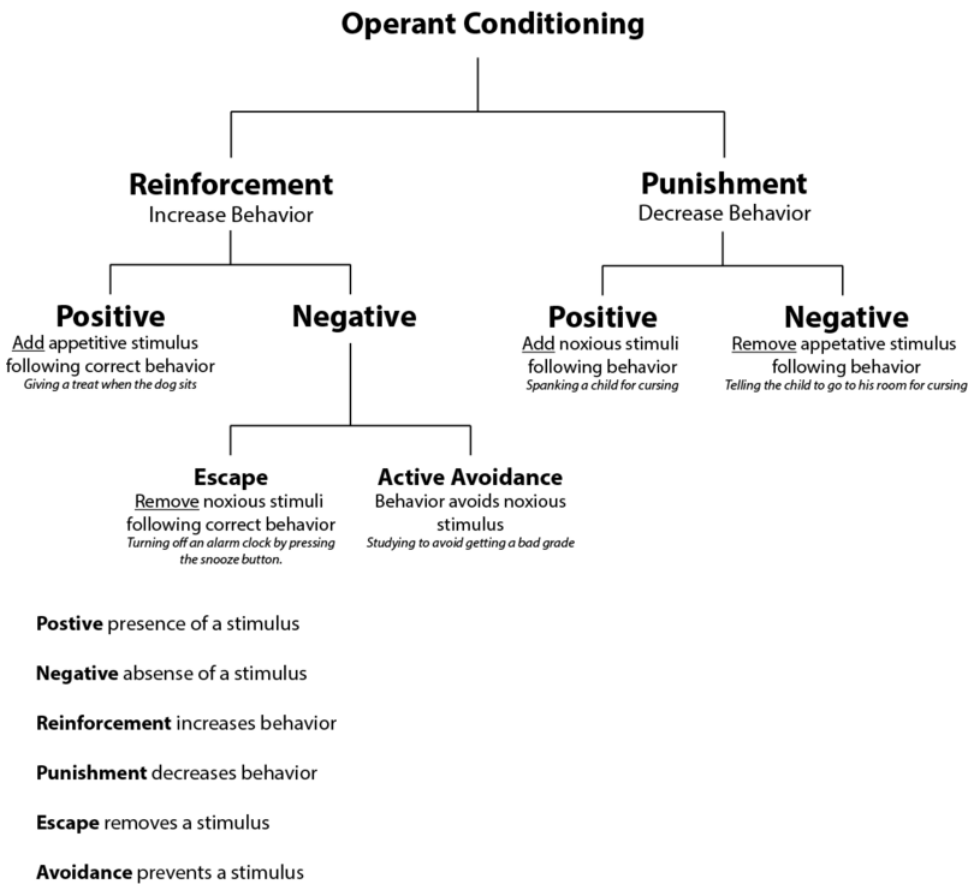

Figure 4.4. the diagram of operant conditioning [47]

Continuous reinforcement : The subject is rewarded at each occurrence of designated behavior. This schedule rewards the subject at the rate of 100%, which brings the most willingness to repeat the behavior. However, it gives out rewards so rapidly, that the subject can soon enter the state of satiation, where it becomes irresponsive to the stimulus (e.g. the subject becomes irresponsive to food when it gets full).

Fixed ratio schedule : The subject is rewarded at every several occurrences of the designated behavior. The problem with this schedule is that, the subject experiences a variable level of motivation to perform the behavior, based on the number of occurrences to reward. E.g. when the number of occurrence is nearly reached and the reward is about to be given, the subject is greatly motivated. However, by the moment the reward is given, the subject is aware that the next reward will be far, and it gets really unmotivated. It's quite likely that the subject will stay at this unmotivated valley, and the behavior change ends here.

Variable ratio schedule : The subject is rewarded at an undetermined number of occurrences of the designated behavior. This includes the case that for each repetition of the behavior, there is a fixed chance that the subject will be rewarded. This schedule provides a steady motivation for the subject. It's the most preferred schedule for reinforcement for the game developers, that's why the players usually see the words like "have 20% chance to stun

enemy", "have 30% chance to dodge attack".

There are also other schedules like fixed interval schedule (e.g. skill Cool-Down from the game design) and variable interval schedule, which will not be elaborated here.

Operant conditioning has also been widely used by circus for a long time, when they use food as lure to teach the animals performing challenging actions. Operant conditioning has also been widely used by parents and teacher trying to educate the children, long before Skinner discovered it as a theory. Some people prefer the use of punishment on misbehaviors, but the use of reward on correct behaviors is more accepted. This leads to some parents deciding to reward the child from time to time. Nevertheless, it should be known that, praising or rewarding the child indiscriminately does not lead to a better performing child, it could even be more harmful [25]. After all, all the behaviors before the praise or reward are enforced, independent of whether it's a good behavior. What's more, the child would get satiated for the indiscriminate rewards, and be less motivated to pursue the reward bonded to good behaviors. Moreover, with two much praise and rewards, the child may get arrogant and taking the rewards for granted.

Operant conditioning is totally consistent with Edward L.Thondike's principle of effect, which was explained in section 2.1. As a matter of fact, operant conditioning itself is a process of learning. It's the individual's learning of performing a certain action, so as to yield a more favorable result (or to avoid an unfavorable result), which is consistent with the instinct to pursue benefits and avoid harm, which is the nature of all animals.

The limitation of operant conditioning to human is that, this mechanism brought from the ancient era is usually short-sighted. It seldom works with long-term rewards. That's why operant conditioning could drive people do things that is not rational.

In the field of game design, operant conditioning has already been widely acknowledged and applied. The games usually exploit positive reinforcement to reward the player for the engagement in the game. For fighting in the game, the player gets experience point, gold and equipment, which the player pursues in order to strength the controlled character. There are also spiritual rewards like achievement systems, badges, scores, sharing to the game community. Moreover, the designer can create multiple stimuli in the game, so that the player is always driven by at least one of the stimuli to continue with the game. For example, *Diablo* is a game where the player controls one character to fight monsters and obtain loots. For each monster killed, the play gets experience point to level up (and finally get bonus attributes and skills). Money and equipment are also obtained from the killed monsters. Moreover, killing some special monsters will result in the completion of some missions, which grant the player additional resources, enhancements, mercenaries, precious gems and so on. In this way, the player always has at least one reward to pursue, so that they will not easily quit playing.

4.4.2 Fogg Behavior Model

[48] [49] [52] Dr. BJ Fogg from Stanford University has improved on Skinner's theory, and has developed a Fogg Behavior Model on how a person behaves. The model points out that, in order for a behavior to occur, three key elements are necessary: motivation, ability and prompts. Noted as B=MAP.

Motivation is in general, similar to what was discussed above, including pleasure/pain (immediate sensation), hope/fear (anticipation for the future) and social acceptance/rejection (sense of belonging).

Ability is also called as simplicity factors, which measures how easy the behavior can be performed. It involves costs like time, money, physical and brain effort, as well as penalty on social deviation or non-routine (change of habit). Specially, the simplicity factor in brain power can be measured in capacity over challenge, as reference to their definitions in flow theory. That means, high simplicity factor defined here corresponds to the state of relaxation or boredom in the flow theory; while low simplicity factor corresponding to anxiety there. There would have been two ways to enhance simplicity factor: to increase capacity or to decrease challenge. However, Fogg pointed out that increasing capacity, or say, teaching or training is a path too difficult. He advocated decreasing challenge by the use of tools or by the change of behavior actually needed to be performed. As a matter of fact, the field of commerce has been working on simplicity for decades: promotion attracts customers by reducing the cost on money. Dedicated shuttle bus relieves the cost on time and physical effort though the use of tools. As for the field of UI design, it has been acknowledged a guideline that the more one functionality is designed to be used, the simpler and clearer it should be presented.

With motivation and ability, the person is willing for the behavior. But for the behavior to actually occur, there need to be a third factor: prompt (or trigger). For example, a product is useful (motivation), and cheap (ability), but the person just does not know it, or the purchase option is not offered to the person, the person will not buy (yet). Take the example of World War 1, the motivation and simplicity are the rising conflict between the Central Powers and the Triple Entente and their military powers respectively. The war was about to start, but it haven't until the Sarajevo Assassination. The case is similar in chemical reaction, for the difference of energy before and after the reaction conforms to motivation and simplicity, and the ferment assembles trigger. Nevertheless, Fogg has latter changed the word from trigger to prompt, which act more as a reminder, alarm or suggestion. This change stresses on the contribution that designer or guider should make. There are three kinds of prompts:

- Signal works at high motivation and high ability. It's just a reminder. Signal alone does not work as well as the other two prompts, so that it's not so common nowadays. Example: When the space is extremely limited, some advertisements could be like: "Crane: +1234567890".
- Facilitator works at high motivation and low ability. It also provides an easier path for the person, so that the problem with ability is relieved on the spot. Facilitator is the most widely used prompt for the moment, and is already widely adopted by many advertisers or UI designers. Example: some sellers promise to the customers already in the shop that they can get some discounts for the purchase.
- Spark works at low motivation and high ability. It concentrates on improving motivation on the spot. Example: Restaurants put posters at their entrance, emphasizing on their delicious food.

What's more, Fogg emphasizes that prompt can be chained. By dividing big and complicated behavior into smaller ones, the audience could be gradually guided to perform each small behavior step by step, until the moment she/he finished the whole behavior.

The human brain has an evaluation function. It evaluate the behavior's value (motivation) and the cost (the reverse of simplicity factors), and subconsciously determine if the behavior worth doing (e.g. by their ratio). According to Fogg's representation, the evaluation function takes the product of the value and the ability (see Figure 4.5).

Traditional advertisements emphasizes on motivation (our product or service is excellent), and sometimes on ability (our price is low) (but the target audience may have to wait until day time,

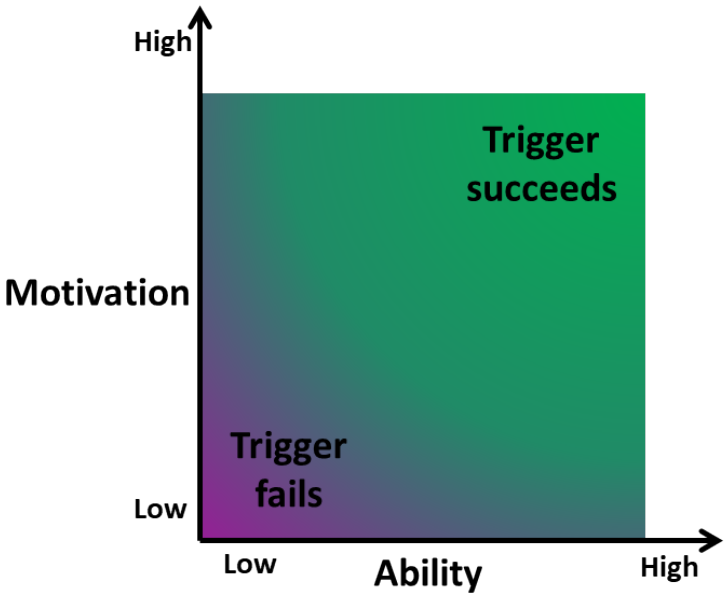

Figure 4.5. Fogg Behavior Model - motivation vs ability

and take a bus to buy). E-commerce does better: the purchase page is directly linked by the advertisement (prompt), and the purchase can be done only by simple clicks (low physical/brain effort).

It should be known that, a single advertisement can pack all of the three elements together. One may receive phone calls from the salesman saying "our product is supper effective (motivation), but it just costs you little (simplicity). If you buy it during the phone call, we will give you a discount or send it to your home (facilitator), together with some extra gifts (spark)."

4.4.3 Remorse and moral issues

Many of the game developers, especially online game developers of this decade know a lot about operant conditioning. They reward the players with virtual products for their engagement in the game (e.g. give a loot box for every fix or variable amount of in-game time or for a certain number of monsters killed). They create compulsion loop [50] that binds the reward to the next playing cycle. There are also free-to-play game where the producers earn not from purchase of the game, but from microtransactions [51] in the game. Through microtransactions, the players pay real life money for the virtual product, which may enhance the power of the player controlled character (e.g. powerups or rare equipment), or may just change the appearance of the controlled character (usually called as "skins"). These games attract newcomers with freemium. But when the player has joined in, they might find themselves unmatchable with other players who have purchased powerups. The player can play very hard, spending large amount of time, to catch up with others people, or the player can use microtransaction as a "shortcut". Due to the nature of impatience and due to the strong motivation induced by the use of operant conditioning in the game, a good number of players will choose microtransaction, to pay for quick strengthening. The

microtransaction model is expected to prevail over one-time purchase model, as a new business model that yields more revenue.

Seeing the commercial success of game industry, more people join in the field of game development. Nevertheless, some of the developers have taken game development as a money tree, without considering of what the players really need from the games. Gamers play games actually for emotional touch, aesthetic, or the joy of learning and discovery. But these game developers cares little about these, they just create games heavily based on operant conditioning and microtransactions. These games turn out only as operant conditioning chambers, which drive the internal test subjects repeating and repeating. It's similar to the gamblers not being able to quit gambling.

The sense of remorse happen when the player has spent some time or other resources on the game, with high motivation, but with a low level of enjoyment [45]. Players addicted to games are usually blamed as "He is too playful" or "He loves games too much", but it's inaccurate. For the above mentioned game, the gamers stick with it sole because of the motivation induced by the game. However, without a care for other elements, the gamers don't enjoy. People hostile to games claims that "Games are useless, it's too much addictive, and it's a waste of time". While it's an incorrect statement to be applied to all the games, it's a right claim for the games that are nothing but Skinner boxes.

The point here is that, operant conditioning and Fogg's Behavior Model can be abused, just like the case of gunpowder and nuclear reaction. The science of motivation is a powerful psychology tool, which can affect great number of audience. To be used for meaningful and significant area (e.g. psychological treatment, education, good habit formation), they are of great use. Contrarily, to be used sole for personal interest (e.g. only commercial success), without care for the potential harm to the target audience, then it's merely a mind control trick to hack on other people's volition.

4.5 Insight

Insight is a moment when the newly received information is added to prior information, making prior information meaningful. Many novel writers, film producers, dramatist, puzzle game makers and talk show speakers are adept in insight. They hide in their early chapters the clues in details that people usually overlook at first glance, and reveal the secret only in later chapter. At the moment the secret is divulged, the audience recall all the clues and details, and these information suddenly all make sense, when the audience experience a moment of epiphany, with exclamation like "Ah ha!", "I should have known this!" or "Eureka!". Spoiler jeopardizes these works greatly, because spoiler prematurely connects the refined pieces of information into an organized whole, where the moment of discovery is gone forever.

Insight is substantially an emotion brought about by learning, and especially, by building connections between the current information and pieces of prior information. Thereby, it's not hard to guess that the students can also experience the moment of insight in the process of insight. For example, the students may have learnt some formulas from physics class in high school. They don't actually understand why there are terms and coefficients put in that way, they just memorized the formulas. But at the moment they deduce these formulas with calculus, they enjoy the moment of insight. Insight is related to meaningful learning, but with one difference: meaningful learning emphasizes on providing enough meaning to the knowledge when it's being learnt, so that the students understand the topic more thoroughly; insight works when the information at the second round provides meaning to the information at the first round.

Background knowledge works in similar way. For example, in the game *Civilization 5: Brave New World*, the civilization of Huns has some unique abilities. One is that it razes enemy cities in double speed; another is that its cities take names from other (in-game) civilizations. If the players play the game first, they may not really understand why there are such strange settings. Nevertheless, when they later on learnt the history of Hun, knowing of their sweeping across Eurasia, destroying tons of cities, they will find these settings immediately make sense, and they enjoy the moment of insight. On the other hand, if they have learnt the history first, then at the moment of see the game setting, they will also immediately recall the knowledge from the courses, refreshing it in the memories. Similarly, when serious games are offered together with traditional educations, each addressing the same topic from their own perspective, the element of insight can bring great joy and enhances knowledge retention.

Chapter 5

Computer security

5.1 The Pressing security threat

Four score years ago, the Stuxnet attack shocked the world with its disguise capability and the power to extend out of cyber world. Every year, security attacks are witnessed here and there. The recent WannaCry attack has swept across the world, affecting more than 200,000 computers across 150 countries, making hundreds of million to billions dollars of loss just in 4 days. One of the largest victims of Wannacry was the National Health Service hospitals in England and Scotland, with 7000 devices affected. Were the kill-swith not found by the security expert, or the attack targeted higher critical infrastructures, it would be a much more serious issue [53].

In 2010, it's reported that 58% software are vulnerable to attacks [54]. Conficker worm was reported having infected more than 1.7 million of computers. Windows and Apple provide weekly patches in their effort to mend security risks [54].

It's reported that, the cost of cybercrime of 2017 has become 23% more than that of 2016. Organizations spend on average 11.7 million of dollars on cybercrime, and those in industries like finance or energy utilities suffers more, up to 17 million [55] . The number of successful security breach per company per year has increased from 102 to 130, by 27%. Special attention should be drawn to ransomware attack, in the sense that, its occurrence doubled, from 13% to 27%. The attack of WannaCry and Petya has even touched people and companies worldwide by the thousands. Not to mention the data breach of Equifax, jeopardizing customers of 143 million [55], or the Yahoo data theft, as all its three billion accounts were say to be compromised to the 2013 hacker attack [56] . Information loss was recorded as the most costy component, from 35% in 2015 up to 43% in 2017 [55]. Malware attack expects an average cost of 2.4 million of dollors. The average resolution time for malicious inside attacks and ransomware attacks are respectively 50 days and 23 days [55]. BakerHostetler reports that, according to 2017, it takes on average 66 days for the corporation to discover security attack after the incident happened, and 3 days after discovering for containment [57]. It's discovered by 2017 cost of cybercrime study [55], that all its studied companies witness virus, worm, and/or Trojans and malware attacks in four weeks, and 69% of the companies experience phishing and social engineering attacks.

The rise of Artificial Intelligence is also going to totally change the world. An greatly increasing amount of money and effort has been dedicated to apply Artificial Intelligence and data mining to the field of commerce, especially after Google's AlphaGo and AlphaZero has shown their unlimited strength. AI is expected to be widely used worldwide. Nevertheless, all computer systems are potentially susceptible to security attacks.

It's mentioned that, United States has many critical infrastructures highly relying on secured and available resources in computing and communication. Nevertheless, the work force that is necessary to secure these resources is far from being enough [58]. Jack Johnson from the Department of Homeland Security, speak it out that: "There is an incredibly shrinking pool of IT security professionals in government." [58] More security experts are required.

With the development of digitalized world, the popularization of smart phones and the ubiquitous Internet of Things, more software and apps are to be developed. Non security oriented programmers also need to know security. Unfortunately, many courses in computer programming, especially introductory ones, have a large variety of topics to deal with, including grammar, logic, data structure, algorithms and debugging. They talk little about software programming or computer security. Under this circumstance, the students, the future programmers, will naturally focus much on just rendering a running program, with little consideration of how resistant the codes will be against security attacks [54]. Thereby, the security training for programmers is also demanding.

Artificial intelligence based and data mining based services are becoming a new favorite pet of the modern society. These systems however, reply on the correct functioning of computer systems of course. Nevertheless, in addition to careless bugs, the systems also have to be robust enough to sustain security attacks. In September 2016, Keen security Lab has already demonstrated a remote attack on the Tesla Model S in both Parking and Driving mode [60], even without physical contact to the car. Such behavior is clearly another example that security attack in the cyberspace could harm people in the real world. When auto drive technology is being widely publicly used, the hack on the systems can cause even more harm. After all, the drivers would be confident with the driving system, relaxed and not quick enough to perform a manual correction when the system fails to detect obstacles.

Amazon and AliBaba's e-commerce has shaken the world. It has become more of a daily activity for the general public. It's reported that ecommerce has already affected 56% in-store purchase [59]. The popularization of smart phones brings people even closer to the digital world. "82% of mobile users search for a local business and 18% of local searches lead to a sale within 24 hours." [59] The whole world is going digital. Nevertheless, common folks are not yet ready for the security problems brought by this trend.

> Year after year statistics show that 60 to 70% of cyber incidents stem from low-tech human errors, also known as the Human Factor.
>
> — MAVI INTERACTIVE [61]

One of the most frequent attacks against the general public is phishing. BakerHostetler in his *2018 DATA SECURITY INCIDENT RESPONSE REPORT* [57] illustrated that, phishing is the top cause of security incidents, making up of 34%. The next is network intrusion (exploitation of system vulnerability), 19%, with 38% of which involves the use of ransomware. The third is inadvertent disclosure, taking up 17%. BakerHostetler recommend good security training of the company employees to be taken into practice, together with security technologies [57].

Besides, after the initial attack, the attacker can insert Trojan on the victim's computer, collecting more personal information. With this information, subsequent spear fishing can be conducted, which is much more effective and harder to detect. The falling of one's computer to cyber-attack affects more than just the owner himself/herself. The attacker can make use of the computer as a springboard to future attack. An employee's computer can be used as the entry point to access the company's secret data; an ordinary person's computer can be also exploited for distributed denial-of-service (DDoS) attack. Thereby, raising the awareness in computer security is actually a matter of everyone.

5.2 Security serious games on the market

In order to relieve this problem, corporations and universities start developing serious game for anti-phishing training as well as security education and training.

Circadence [62] is a corporation featuring cybersecurity education, training and assessment. From 2005 to 2008, Circadence created MVO to support mission critical communications for the US military and government. In 2010, the company begins to develop cyber range products for cyber test and evaluation. In 2016, the company created an online training platform called Project Ares. In project Ares, the players in a group are immersed in a realistic and interactive virtual environment, using real-world tools, facing numerous missions concerning authentic threats. It features a deep learning powered AI used inside, which acts as an advisor, umpire and rival. It also provides to the instructors a dashboard view to monitor students' performance. Showcase of skills, trophies and badges is also created, so that the player will be more motivated to engage the training and perform better (because of operant conditioning elaborated in section 4.4.1). It can also be integrated to the LMS through IBM BLUEMIX. It support also user-customized mission.

There are also other serious games in the topic of security. *Anti-Phishing Phil* [63] from Information Security Office of Carnegie Mellon University is a serious game that teach the audience good habits to avoid phishing attacks. The player controls a fish, which represents themselves exposed to phishing attacks. Worms appear around the fish, which represent the websites they try to visit. The players decide whether to eat or reject the worms, by deciding whether the websites related to the worms are phishing websites.

Keep an eye out [64] from Serious Game Store from Daesign is a serious game on digital security. The game mimics the daily work of the employee at the company. By interacting with the objects in the scene and making decisions like whether to use the USB, the audience learns of the security risks around them.

Info Sentinel and *Agent Surefire* are two serious game series created by MAVI Interactive [65]. They targets the human factor in the defense against cyber-attack, and cover the security in the office as well as in travel. Except for the exploration and decision making in the simulated environment, *Agent Surefire* also provide a "catch the hacker" scenario for those interest in further practice. Numerous potential security threats are there to be discovered, getting the audience more adept at finding and removing the security weakness around.

CyberCIEGE [66] made by Naval Postgraduate School is a serious game that teaches the students the knowledge about information assurance in a way similar to *SimCity*. The players take the role as an enterprise manager, who choose and decide for the company, in a way to protect the corporate assets from security attacks. This is represented in the game as purchasing and configuration of the work stations, servers, OSs, applications and network devices, as well as the training of the employees. However, the money available to be spent for these is not infinite. Therefore, the players have to correctly recognize the problem as well as the correct approaches to deal with the problem. So there is a element of resource management in the game.

Cyber Awareness Challenge [67] by Carney, Inc., Defense Information Systems Agency and SAIC is a serious game made for the authorized users of the Department of Defense and Federal information system. The players carry out a series of tasks related to the protection of sensitive information, in the form of point and click. Special attention is given to the prevention of information leakage.

CyberProtect [68] by Carney Inc, Department of Defense and SAIC, is a serious game made for the network end-user and network administrator alike. Players start with an unprotected operating system, and install security tools to add the defense for security attacks. When the installation

phase is finished, the system is test for susceptibility to security attacks. There is finite amount of budget for the players. Thereby, the players also have to deal with resource management, spending the resources on most critical area corresponding to the vulnerabilities. *Cyber Awareness Challenge*, *CyberProtect* and many other online training tools are kept at IASE (Information Assurance Support Environment) at https://iase.disa.mil/eta/Pages/online-catalog.aspx.

Cloud Defense [69] by Gronstedt Group is a serious game to help the employees to learn the concept and acquire the knowledge in cyber security. The game follows tower defense-style, and concentrates on Amazon Web Services security protocol. The players set up network equipment and set firewall architectures, in order to create a system that passes good traffic while preventing attacks.

Chapter 6

Development platform

Pedagogic object

The game is to be a serious game delivering the knowledge of computer system security to the students. Nevertheless, in addition to just "teach", the game emphasize on providing the students with personal experience, by allowing them to actively choose and use the security techniques in face of variable situations (as reference to discovery learning from the chapter Active learning).

Target audience

This game's target audience is those who have already some basic knowledge in computer networks. The knowledge of computer security is not an requirement. Nevertheless, if the audience has some knowledge in this field, it helps them to understand the terms in the game, and the game on the other hand reinforces the academic knowledge. Thereby, the game is best suited for students who are meanwhile studying computer security or network security.

Platform

The game is to be played both on mobile phones and on PCs

To achieve this, the game will be an HTML5 application developed with JavaScript. In this way, the players are able to access the game with a browser from mobile phones and PCs.

Game engine

There are authoring tools (e.g. *Construct2*, *gamesalad* and *RPG Maker*) which provides the game creators with assets and lower the requirement on programing. However, authoring tools usually targets a specific game type (e.g. RPG), which adversely makes it difficult to make games of other types (less flexibility). Also, authoring tools are more for the non-programmers, and feature mouse operations. But for those who need to frequently access code, jumping among variables and functions, authoring tools could be even counter-productive.

There are also professional game engines like Unity. Unity has great amount of game assets, and support coding with C, C++, C#, JavaScript or Boo. However, considering the author's lack of prior knowledge in computer graphics or game development, the amount of time needed to learn this engine is expected to be high.

Phaser is an open source 2D game engine for desktop and mobile HTML5 game framework. These properties of Phaser suit the platform choice well. Besides, being a real engine, it's flexible enough to implement all the game concept of the author; being a small engine, it's quicker to learn. Therefore, Phaser is finally chosen as the game engine, and HTML5 is the platform.

Modularity tool

JavaScript intrinsically does not support modularity. It's possible to split the code into multiple files. However, by doing so, the files have to be added manually and very carefully, as the order to add the files may ruin the dependency also. Therefore, the tool browserify is used. It automatically detects dependencies among files, and merges the files into a single bundle. In this way, the html file needs always and only to import the bundle file, independent of how the modules change internally.

Text editor

Notepad++ is chosen as the text editor during the development of this game. Notepad++ is a free source code editor that provides identifier and keyword highlighting for numerous languages and file formats.

Debugging tool

Google Chrome is selected as the browser for the debugging phase of the game, thanks to its developer tool.

XAMPP is selected as the server to host the game for the debugging phase.

Map editor

The map of the hall scene is made with Tiled, starting from the map created by my predecessor Antonino Aloi's work *Dropit!*.

Data gathering service

Python is chosen to write the web service to collect player's performance data in the game. Together with Tornado and SQLAlchemy, it receives player data and stores them in the database.

Chapter 7

Game design and development

7.1 Game mechanism

The game is developed as a turn-based strategic game, around fight in the cyberspace using network attack or defense techniques. The core gameplay is strategy making (how to act and react according to the current situation). Other gameplay elements include discovery (find the possible attack path and follow/block it) and resource management (whether to put on a single powerful defense, or to activate multiple less powerful defenses, and which ones). It's already discussed in previous chapter 4.2.1 that, decisions related to academic knowledge are profound and takes a relatively long time. The time needed for players who are not yet trained to apply their academic knowledge is even higher. Therefore, a turn-base strategic game has a pace much better suited than other game genres (e.g. better than Real Time Strategy and, not to say First Persona Shooting or Action games). Another merit of turn-based game is that, instead of forcing the player to adapt to the pace of the game, the game adapt to the pace of the player. The players can pause at the current round whenever they are making decisions, and they can proceed to the next round whenever they has finished the operations. The former avoids the players to be overloaded with decisions, entering a state of axiety; the latter avoids the player to have nothing to decide, falling to the state of boredom. In this way, the players are kept in the flow state, independent of their preferred pace of thinking. Besides, an turn-based strategic game is totally paused at player's round, granting the player enough time to learn or review the knowledge, from inner help, from peer discussions or even from materials exterior to the game.

The major part of the game is the cyber battle (the cyberspace scene). There, the player is going to compete with the rival in a turn-based manner. The one taking the role of intruder needs to launch security attacks (called as offensive acts) on the target server or defender's personal computer, so as to impair the defense or to deal damage to the assets under defender's protection. The defender, however, needs to defuse the attacks with defensive security techniques (called as defensive acts), while also keeping the company's server running. If the defender has protected the company assets for enough rounds, the defender wins. On the contrary, if the intruder has dealt enough damage to the company assets before the rounds ends, the intruder wins.

Both the intruder and the defender need resource to perform their acts. Intruder gets small amount of resource each time his/her round starts, whereas a better source of resource is the bonus obtained when inflicting damage to the company assets. To do this, the intruder should use one attack or a chain of attacks to achieve credential breach or data stealing. That is to say, the success in attacks on the assets push the game pace towards intruder's victory, and it grants

the intruder with more resource, which boosts the preparation for more attacks. This positive feedback creates a small compulsion loop (see section 4.4.3), like what the game developers usually create. However, to maintain a balanced game, for the majority of the cases, the intruder can only compromises the credential once in each round. Otherwise, the intruder can repeat and repeat exactly the same attack in the same round, without leaving the defender a single chance to fix the vulnerability. From the other aspect of view, rewarding the intruder only once per round is a design following the fixed interval schedule, which avoids the player's satiation to reward (see the section of operant conditioning 4.4.1).

The income of resource of the defenders comes from every client requests their server serves. Each failure to serve a client request not only means a failure to gain resource, but also results in a penalty to resource: the client is unhappy for the denial of service. This sets the players playing as defenders into an intriguing situation. Income-preferring players would improve the server's capacity to serve clients or server's resistance to DoS attacks (DoS attacks hamper the server in serving client requests), which may leave them undefended in face of attacks on their assets. Safety-preferring players would improve the defenses on assets, risking their server failing to serve clients. Such kind of choices and trade-offs is the core of decision making in entertainment games, where the players decide and choose based on their own preference, knowledge as well as the current situation. And they enjoy doing these a lot across a variety of games.

In the actual situation of cyber-attack/defense, the majority of defensive techniques are activated before the attacks really occur, instead of being invoked only during the attacks. Thereby, to mimic the situation, in this game, the defensive techniques also have to be activated in defender's round, before the intruder launches attacks his/her round. That is to say, the intruder can see what defenses are there, and he/she should use his/her knowledge to deduce which attack or chain of attacks will prevail, and which will fail. The defender, however, has to guess from which way the intruder may attack, as he/she has to set up the defense for future attack.

The game is strongly buff based. Defensive security approaches usually enforces positive buffs on the defender's side. For example, the act "CDN" gives a buff of the same name, which grants the character some resistance to DoS attacks as well as a higher server capacity. Offensive security approaches could enforces negative buffs on the defender's side, but it could fail because of the presence of some positive (defensive) buffs on the defender. For example, Sniffing attack will take effect if the defender has buff "MITM" (Man-In-The-Middle), but has no buff "Encrypted". If successful, Sniffing attack will deal damage and enforce buff "Credential compromised", which usually means that the intruder has won this round. In a word, the game features the relationship between the offensive and defensive approaches.

There is another scene before the cyber battle. That is the organization's hall. There, the player can talk with NPC's for information about the organization and about what kind of fight will be in the following cyberspace part. NPCs could also give some hints or guidance to the player.

7.2 Story setting

The story is set in 2029. At that time, the power of Internet has nearly reached every corner of the world. The rise of strong artificial intelligence has also made it common that many big companies let the AI to manage much of the basic-level and medium-level decision making.

One character called Godfried is a security expert who is hired by companies at their critical times to help defending security attacks. He is also called when they need to search for security

weak points. His creed is that the use of artificial intelligence is going to greatly enhance the intelligence of the world, and that the systems need to be protected.

The other character called Christopher is an ingenious cyber youth who have decided to join in HackIt, one of the world's biggest hacker group. HackIt usually hacks on systems where AI takes most of the control. They do this in order to warn people of the emerging threat of strong AI or even upcoming hyper AI. Their creed is that human should cautiously limit the use of AI. Otherwise, sooner or later, AI will take over the world. Christopher also dislikes the government's behavior of keeping eavesdropping and keeping secret from the public. He considers Edward Snowden as a hero for liberty. But his major concern is still about AI.

Being so much diverged in their ways. Will Godfried and Christopher one day have to confront each other in cyber world?

When controlling Godfried, the player is supposed to use the defensive techniques to defend from security attacks while maintaining the server functioning, there are also situations where it's required to discover vulnerabilities of the companies. When controlling Christopher, the player is supposed to use offensive techniques to strike down the target system as much as possible. Many security topics are covered by both characters, which allow the player to understand the techniques from both sides.

7.3 Personal notes

An entry called "personal notes" is accessible throughout the game. It's just another name for the cyclopedia for those security terms referred to in the game. The effort to put cyclopedia in the game have be found in many much "serious" games like *Civilizations*, *Romance of the Three Kingdoms 11*, *Zero Escape* and *Age of Empires*, just as discussed above in section 3.5. The accessibility to this cyclopedia both from the main menu and during the game (hall, cyberspace and review) also conforms the guideline mentioned there of making academic knowledge in the game accessible to the player. Also, according to Fogg Behavior Model (subsection 4.4.2), this effort improves the element of simplicity, which leads to great tendency for the player to read the material. A button is also added on the description window of the acts/buffs, for the player to jump directly to the entry in the personal notes. This saves the player's effort to open the personal notes, and then search for the entry for the act/buff (it's a facilitator, as reference to Fogg' model). On the other hand, calling the cyclopedia as "personal notes" (instead of just "cyclopedia") is another effort to align the game with the fiction. The "personal notes" is defined as "a personal notes made by the characters when they learnt of security knowledge from universities". As a result, when the players open personal notes, they will not feel abrupt or take the cyclopedia as something external to the story. Contrarily, they feel it quite nature and reasonable that the characters refer to their notes whenever they have forgotten something. In this way, the player will suffer little penalty switching from the fight to the pool of knowledge, so that the immersion is maintained and break of flow is kept to the minimum. Someone may feel strange why both characters, one offensive and the other defensive, share the same personal notes where both offensive and defensive techniques are inside. The explanation is quite simple: one needs to know its counterpart enough so as to prevail in the competition. Defenders need to know how the intruders sneak in, so as to organize their defenses; intruders need to know where the defenders fortify in order to scheme a plan through it. As a matter of fact, the scenarios are designed under this guideline, in the sense that, for each security topic, there will be two scenarios, one for the intruder's perspective and the other from the defender's perspective.

Description, personal notes and external links

Except the tutorial and the intro scene, there are three places where the knowledge could be delivered: the act/buff description, the personal notes and the link to external resources. It's already discussed in tangential learning section 3.5 that, the reading materials should be brief and short. Player chooses serious game over traditional materials, because of the game's better graphics, narrative and interaction. Thereby, long text is as much disliked in games as it is dislike in the textbooks. The players tend to skip long text, and if they missed something important because of this, they blame the game designer for not having told them. So, the act/buff description, as the one close to the game, is limited to one page, and it should contain only the most important information for the gameplay itself. On other word, no academic deduction, no historical events, no technique application. Some might argue that, there could be some players, who would only read these short descriptions, who would never access personal notes or external links. With no real academic knowledge in the act/buff description, they learn little from the game. Well, in order to answer this doubt, it should be known that 1. These players are not like those who are willing and interested to learn. These players play this game solely for entertainment. Any academy related stuff would drive them farer (the failure of those educational games mentioned in the beginning of tangential learning section 3.5). Therefore, there is no choice here. Haste makes waste. 2. Even if the player learnt little from these descriptions, the experience with these terms or techniques still made a difference. When the player later encounters the terms again, they can have a moment of insight 4.5. Terms mentioned in the game will turn out with special meaning, receiving special attention, not like other terms which are considered irrelevant to the players. A term with special attention will be learnt and accepted faster.

Personal notes is then the second closest source of knowledge. Even if it's accessible nearly everywhere around the game, with even links from the act/buff, the player still have to open it, and, after opening, to be patient enough to read multiple pages. It should be known that, this is already something that cannot be expected from some "impatient" players as mentioned above. For those interested, this is of course no problem. They would like to know about the academic knowledge and would feel like discovering the link between knowledge and how it's presented in the game (e.g. why the act "OTP" costs greatly, but is unbreakable). Personal notes is the major source of knowledge that is provided by the author. Still, it should be remembered that infographic is better than large, pure texts. Thereby, although many of the terms are quite abstract, the author still tried his best to add pictures to personal notes.

Link to external resources like Wikipedia is the one least closest to the game. The player has to totally switch out of the game in order to access these materials. It also happens that, sometimes these materials are written for more professional readers, which means, players with less prior knowledge may get confused reading them. The external resources are however, the most comprehensive and authentic source of knowledge, and is also acknowledged by the academia. Thereby, it's excellent for those most patient and curious players. To put external link in the game is not a common practice, but it can still been found in *Plague Inc.* (with link to the film *Dawn of the Planet of the Apes*) and in *Kerbal Space Program* (with link to NASA), and it's advocated by Extra Credits (see section 3.5).

The relationship of these three sources of knowledge is shown in Figure 7.1:

With these three tiers of structure, players of different motivation, different level of interests and different amount of prior knowledge can all find their fitting source of knowledge. Another frequently seen example of multi-tier approach is difficulty change (see section 4.2.3) (usually found as: easy, normal and hard). The only difference is that, difficulty change adjust to the player's skill level to enable a pleasant experience (flow state) for all, while here it adjust to the player's interest level to provide the most knowledge without incurring resentment.

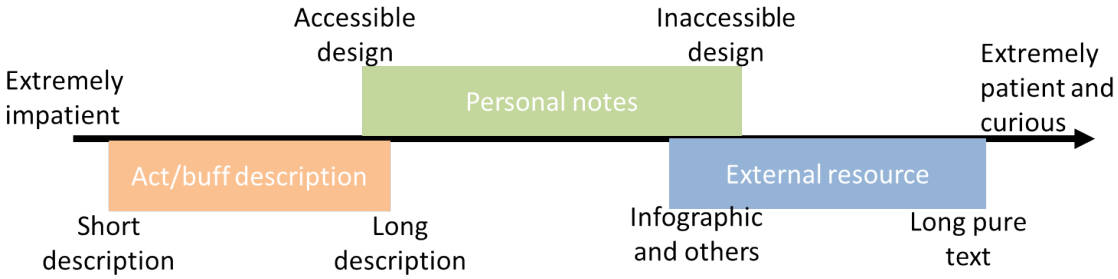

Figure 7.1. sources of knowledge vs player's patience and curiosity

7.4 Tutorials

Due to the design of the game, several tens of security terms are going to be presented to the player. As a consequence, when starting a new scenario, players who have no prior knowledge in computer security may get nervous in front of several unfamiliar terms. On the other hand, the players also have to learn to use the windows or logs, to obtain information of the offensive/defensive techniques as well as information of the current situation, which is crucial for their decision in each round. Therefore, three tutorials are created to meet these needs, following the guidelines discussed in earlier chapter 4.3:

Tutorial 1: As the very first tutorial, it teaches only the basic clicks in the hall scene. The hall scene will be, for each scenario, the place for mission briefing and general knowledge preparation. This is not a lot of control to learn, but according to the guideline "teach gradually", this tutorial should not be too ambitious and should not arrogate to talk about things of cyberspace scene. Thereby, apart from just a tutorial on control, a storyline is designed. The player is going to wander around in the company, finding clues to a security breach, and understand some guidelines in computer security. There are some instances of such designs in this game, where the player understands by seeing and experiencing, which is much exquisite approach than verbal instructions.

Tutorial 2: This is the first tutorial on cyberspace. Therefore, a lot of information is to be given. According to the guideline of teach gradually, and the guideline of avoiding big chunk of text, the intro scene of this tutorial only deliver a limited amount of information. Contrarily, much of the things are revealed to the player as the game goes, in the form of in-game dialogues, turn by turn. The acts are also unlocked one by one. The setting of this tutorial is that player's avatar is a student doing internship in a company, and there is a senior instructing him. Tutorial instructions are spoken out of the mouth of the senior as well as the avatar's thinking aloud. This made the instruction less intrusive and coherent with the game fiction. On the other hand, the rival starts with all acts locked, and it obtains more detrimental acts only gradually with time. This guarantees the tutorial to start easy, avoiding the situation that the player is easily beaten by the rival before he/she understand what is going on. The outro stage of the tutorial is also a good place to teach the player something secondly urgent. After all, the player has already experienced the game. Therefore, the explanation of the things in the game has become meaningful to them.

Tutorial 3: While the player takes the defender's role in the previous tutorial, he/she plays the intruder's role in this tutorial. This tutorial is based on the historical event of Alan Turing cracking on German encipher machine Enigma. As the second tutorial in cyberspace, there are fewer instructions on control, and the in-game dialogues tend to describe the story more. This

change allows the player to better enjoy the narrative and the game mechanism. On one hand, Turing's crack on Enigma is itself an excellent example of cryptanalysis, which suits the topic of this serious game well. On the other hand, the player, being weary of killing hundreds of Nazi in FPS and RTS games, will find this tutorial a brand new experience to fight the Nazi in another way.

Not that these tutorials are intrinsically the same as scenarios, only with slowed pace and reduced difficulty. Therefore, it's more than enough to say that the tutorials conform to the guideline of "similar to the formal game experience".

7.5 Scenario files

The implementation code is loosely coupled with the content of the scenarios. The scenarios are specified by the scenario files under the folder "scenarios". This is to provide module independence to the game, and allows progressive extension of the game whenever more scenarios need to be added. The use of JSON format also makes it possible for the players themselves to create their own customized scenarios. Entertainment games like *Warcraft III* and *Plague Inc: Evolved* benefits a lot from player-defined scenarios, and this game can expect similar things too.

Besides, a JSON file called assetsTable is also created, to gather and organize nearly all the assets used in the game, including images, sprite sheets as well as sound and music. This is for the similar purpose as those files defining scenarios. Without this file, to add more assets, one have to access the source code, find the correct line, modify, recreate bundle, clear the browser cache and refresh. With this file, one only needs to add an entry in the file, clear cache and refresh. Furthermore, those who create new scenarios will need an asset table (for original assets and newly added assets alike) to facilitate their selection of assets. This file itself is a kind of asset table. P.s. because of this, this thesis will not give an asset table or asset key, whatever it's called. Those who need it can just read assetsTable.json.

Similarly, the content of personal notes is also written in the JSON file persoanlNotes. It's recommended to created entries in personal notes for all the new acts and buffs created for the new scenarios. In this way, the player always have place to refer to whenever they find some strange names .

For the details of the player customizable JSON files, see Appendix C

7.6 Scenario design

This section talks about the scenarios already created. Most of the scenarios will follow this pattern: the scenarios are in pairs (e.g. 0 and 1, 4 and 5) and are unlocked in pairs. In each pair, the scenarios with even number will take an earlier one, starting talking about a new security issue, with the player takes the intruder's role. The scenario with odd number will then be related to the defender's role, letting the player experience from the other aspect. This choice of intruder precedes defender is made because: 1. It's historically true that speculator and law breakers precedes guardians and new laws. 2. In this game, the intruder can make decisions based on the buffs on the defender, which is visible. The defender, however, have to build up defense in preparation for future attacks, which is not clearly known. A defender's role can be more challenging for someone new to the topic.

- scenario 0: Reveal the secret
  In this scenario, the player play as intruder, trying to break the encrypted channel of the country J.

  scenario 1: You will not understand
  In this scenario, the player play as defender, and is instructed to protect the company's and client's secret transmitted over the Internet.

  The major topic of this pair of scenarios is cryptography, with a minor topic of spam email. The player is to learn of some famous key exchange algorithms, their strengths and limitations, by seeing them successfully defended attacks, or fallen to the attacks. They will also see the defense to replay attacks and spam emails.

- scenario 2: Authentication breach
  The player takes the role of intruder, trying to gain access to the war machine project of the country J.

  scenario 3: Credential guardian
  The player takes the role of defender, protecting the nuclear plant from falling to the hand of some terrorists.

  This pair of scenarios emphasizes on authentication, including password related stuff. The player will see cryptographic hash functions, as well as dictionary attack targeting the hash functions. They will also see the functionality of single sign on provided by password manager or Kerberos. Moreover, they will understand that an attacker can perform privilege escalation after the first intrusion, to more severely jeopardize the system.

- scenario 4: Wi-Fi creeper
  The player acts as intruder, sneak into the cafe bar or airport that target victim usually goes, and try to perform Wi-Fi-based attack.

  scenario 5: Is it your Cookie?
  The player acts as defender, protecting the company's assets from Wi-Fi attack.

  This pair of scenarios features Wi-Fi attack as well as cookie-based attack (connection hijacking). The player will be shocked to see so many potential attackers near them, in the cafe bars or around their home. They will surely think twice before connecting to unsolicited Wi-Fi hotspots. Moreover, new attacks like CSRF will be mentioned along with XSS, which the public should pay attention to in their everyday life.

7.7 Scoring

Scoring and achievement systems are two related elements frequently found in entertainment games these days. It's already discussed in the part of elastic challenge in subsection 4.2.3 that they contribute to a higher motivation to repeat the game and try to do better. Besides, by explicitly setting up a goal, a target, operant conditioning 4.4.1 will urge the player to pursue it. This guarantees some minimum repetition of the game before mastery. For the detail of the calculation formula for the scores, see the user manual at subsection A.9.1.

7.8 In-game scripts

In-game scripts are added to provide something dynamic during the game. Firstly, in-game scripts enable dialogues to be displayed in the game, which served as a good place for instructions and

guidance in the tutorials. As a matter of fact, the use of in-game dialogues has already surpassed the use of textual or image-text-mixed instructions before or after the game, as mentioned in the tutorial section 7.4.

Secondly, in-game scripts also dynamically unlock new acts as the time goes in a tutorial or scenario. This allows the game to start easy and simple, with little new things of new terms to learn at once. Except for its intensive use in tutorials 7.4, the dynamical unlocking of acts is also used in the formal scenarios. Besides the purpose of starting easy, it's also used to emphasize that the acts being unlocked are newer technologies, which are in general more powerful. These acts are indeed more powerful according to the game mechanism. Thereby, for the sake of a balance game alone, they are supposed to be unlocked only in later part of the game. The dynamical unlocking of acts also affects the behavior of AI, which will be elaborated in section 7.9.

```
{
    "round": 11,
    "dialogues":
    [
        {
            "name":     "Senior",
            "portrait": "youngbusinessman",
            "dialogue": "An urgent report! ^The hackers started to intercept messages
            between our server and our clients. ^Some clients have reported that their
            login passwords are uncovered. ^I will not tell them that our connections
            with them are actually unencrypted... ^You may have already noticed that some
            of our assets are already lost due to this leakage. A buff called
            \"Credential compromised\" is also created on you to indicate the data
            leakage. ^Godfried, quickly find a solution to encrypt our messages!"
        },
        {
            "name":     "Godfried",
            "portrait": "Godfried",
            "dialogue": "eh, encryption... ^My memory of cryptography has faded with
            time. I have nearly forgotten everything just one month after the exam... ^I
            need to review the knowledge of \"symmetric cryptography\" first. It's the
            prerequisite for message encryption. ^Nevertheless, after symmetric
            cryptography, I need to select a specific encryption technique and \"apply\"
            it."
        }
    ],
    "newActs":
    [
        [],
        ["Symmetric cryptography", "DES encryption","3DES encryption"]
    ],
    "shouldApply": ["3DES encryption"]
},
```

Figure 7.2. An example of one piece of in-game script

Thirdly, in-game scripts can also lock on a turn, before the player has done certain actions. This is only used in the tutorials, to enforce the players to follow guiding instructions to try the actions by themselves. These type of lock is specified inside "shouldApply" element, as shown in the figure 7.2

7.9 AI design

The game supports single player mode. In the single player mode, an AI has to take the role as player's rival. Two adept players are able to play in this game perfectly, making no single mistake and miss no chance at all. Technically speaking, an AI of the similar capability can be created,

by specifying more strict rules and perform more checks before it acts. Nevertheless, this is not necessarily the AI needed here.

For entertainment games, a game with greater depth (see skill ceiling within subsection 4.2.3) is always better, for the players keep with the game for long. The higher the game depth, the more difficult for the players to master, and for a longer time the players remain. Nevertheless, for a pedagogic game, the game is primarily supposed to be played for a limited number of times, when the players are not yet very familiar to the knowledge referred to in the game. At the moment they learnt all the academic knowledge, they should reversely be willing to quit from the game. This is clearly a point opposite to the purpose of entertainment games.

As a result, for a pedagogic serious game, a clear skill ceiling can be useful (for more information about skill ceiling, see 4.2.3), as long as it corresponds to the point when all the related academic knowledge are learnt. When reaching this skill ceiling, the player will feel bored and unchallenged, and they will not spend more of their valuable time on the game. Antagonistic games are most attractive when the player find matching opponent (because of flow). So the AI should also have a skill level not far from that of the potential players. Thereby, a perfect AI is not expected here.

There is another thing that affects the intelligence level of the AI, which is the action log. The action log notes down the history of all the acts performed by the characters, as well as whether the acts succeeded. More importantly, if an act failed, action log also notes down the reason for the failure (e.g. the presence of positive buff in the way, or the absence of a negative buff necessary for the attack). This "reason" is another place where the players learn the relationship between the acts. When the players take the role of intruder, log entries with failure will be filled at their mistakes. However, when the players take the role of defender, its rival, the AI controlled character is the one who should fail a number of times. Based on the game mechanism, if an intruder constantly fails in attacks, he/she could have problem with inadequate resources. Thereby, in order to sustain the AI despite its failures, AI gets a boost in resource.

Finally, the AI is designed as the following: in AI scripts, there are a number of action patterns. Each action pattern is potentially a chain of acts that the AI will perform in a round. The AI will randomly choose one from these patterns, based on the chances set for each pattern. However, if any of the acts in the pattern is not yet unlocked, the pattern is considered locked, and it will not participate in the selection. When a pattern is selected, the AI will first guarantee all the related acts are learnt. Then, AI will calculate if its resource is adequate for applying all these acts in a single round. If the resource is inadequate, it will wait till next turn. If resource is adequate, AI will apply these acts, in the order as specified.

Figure 7.3 is an example of AI script for the intruder written in a scenario's cyber file. The AI has, for example, 28% chance to apply "Spam email", and 25% chance to apply "MITM" followed by "Sniffing attack". If the sum of all the chances is less than 1, AI may sometimes decide to do nothing. If the sum exceeds 1, the last action pattern(s) in the list will witness decreased chance. This brings an intriguing effect, which is the chance of AI's tendency within the game. It's mentioned in section 7.8 that, the in-game scripts can unlock some of the acts previously locked. The action pattern will be unlocked if all its containing acts are unlocked. The newly unlocked action pattern joins in the selection, and could result in total chance greater than 1, thus reducing the chance AI will select last patterns. For example, for the case of figure 7.3, all acts are initially unlocked except for "Quantum computing". So the AI will select pattern 1, 2, 4, 5 with chances the same as specified for each of them. Nevertheless, when "Quantum computing" is unlocked in the game, pattern 3 is unlocked, and the total chance becomes 1.2. As a result, pattern 5, an attack pattern that is designed to be performed only in the early rounds of the game, is never selected. Pattern 4 also suffers a little decrease in chance. By this method, AI starts with child's play, and tends to perform more powerful attacks later.

```
"AI":
[
    {
        "pattern":  ["Spam email"],
        "chance":   0.28
    },
    {
        "pattern":  ["Replay attack"],
        "chance":   0.28
    },
    {
        "pattern":  ["Quantum computing", "MITM", "Sniffing attack"],
        "chance":   0.2
    },
    {
        "pattern":  ["MITM", "Sniffing attack"],
        "chance":   0.22
    },
    {
        "pattern":  ["Sniffing attack"],
        "chance":   0.22
    }
],
```

Figure 7.3. An example of AI script for the intruder

The other point is that, AI follows the preset pattern in the AI script, but its actual behavior change randomly each time. In game design, a term called "elegance" represents the property that a simple game mechanism can create multiple situations and various game experience. A game mechanism involving randomness thus tends to be elegant. Even by repeating exactly the same scenario, the player still experience different situations, for AI will act randomly and differently. There is another benefit of randomness-based AI: it fights rigid thoughts. Students would sometimes prepare for exam by matching the general shape of the question to the answers, instead of really learning the knowledge and understand the logic deduction. They do this so rigidly that they get it wrong when the professors change the exam questions by a little. Similar situation here, with exactly the same AI action, the students, especially those adept at short term memory, could remember to "do A in the first round, and do B in the second round". AI's random behavior each round is similar to professor's randomly changing his exam questions each time, which enforces the student to really think how to act or react, instead of just memorizing.

7.10 Multiplayer mode

Some single-player games adhere to the PVE style, where the players follow the paths designed by the game creators, which is usually the case for RPGs. Some other games, including CyberCraft, are in symmetric design, where the two or more characters or factions standing in similar situation competing with each other. These games adhere to the PVP style. At the absence of a real human controlled opponent(s), AI takes the role. Nevertheless, AI is capable for some brainless actions, but it's more often ineligible to be a subtle opponent. Games like Starcraft or Dota even give AI some privileges like map hacking or resource bonus so as to be matched with the player. Besides tactics in the game, a player can also play Yomi against the other players, which yields additional fun. These are the reasons why multiplayer mode in general match players with higher level of skills (see Figure 4.3). Moreover, friends playing the same game together will enjoy all kinds of interaction with each other, which is the core of social game. These human strategies and human interactions are the main reason why online games are becoming more welcomed than single-player games these years. Thereby, a multiplayer version of CyberCraft is expected, to add

more fun elements. Categorized by network topology, there are three patterns following which a turn based game can be support multi players.

Offline: Old games allow the gamers to play together by sharing the same device. When the controller role changes in the game, the device is passed to the next player. This is the most basic approach, without the involvement of a network module, and the designers don't have to worry about things like synchronization or player communication. Afterall, the players are next to each other.

Peer-to-peer: Later in the age of offline games, it becomes feasible to use network modules to support multi player for the games. With this pattern, each player can use his/her own device, each with one copy of the game installed. The players stay in the same local area network, and they connect to the game host based on the IP address. In this case, the players are still near to each other, so that they can still communicate by words. Nevertheless, the network modules and synchronization mechanism are needed in order to synchronize the data across the devices.

C/S or B/S: For the current age when online games prevail, players can just sit in their homes while playing with friend all over the world. There will be a certain number of game servers, which do the game peering, synchronization and data process for the clients. A portion of the processing can be offloaded to the client side though, but the server side has to synchronize the data among different participants in the game. With this pattern, the game is actually composed of two pieces of code, one for the client side and one for the server side. Because the players are physically separated, a communication tool is usually expected for the players to discuss their strategy. Moreover, a time-out mechanism is also needed, so as to avoid the other players waiting in despair for one player's inactivity.

For the work of CyberCraft, the main study phase is provided by single player mode. The multiplayer mode is made only to add more fun. Thereby, its multiplayer mode follows the first pattern, the simplest and quickest pattern. In the double player mode of CyberCraft, one player takes the role of intruder and the other the defender. They fight against each other by passing the device to the right person when his/her round starts.

Chapter 8

Future work

8.1 Future improvements on CyberCraft

- Better graphic. The game uses copyright-free and self-made images. The designer has tried his best to find or create the images that suits the game fiction as much as possible, but the images still cannot be compared to the main trend commercial entertainment games. After all, there are some players who are so used to fancy graphics, who get contemptuous at lower graphic games at their first glance. If there are someone interested in improving the game, providing a more fascinating graphic and effects, giving the players a more vivid feeling of "hackers", "cyber attacks", it would be appreciated.

- Picuture for acts and buffs. Most of the games, especially modern games are strongly graphic based (as compared to legacy text-based games). The skills and buffs are primarily illustrated will pictures. The text explanation comes next, only when the player's mouse hovers over the button. Unfortunately, it's infeasible for this game to hide the text of act names or buff names. However, one can still add pictures, to be displayed together with the names. As a matter of fact, in the scene of cyberspace, the space for items has already been left before the list of act names or buff names. And in the detail of the acts or buffs, a larger space can sustain better and much bigger pictures.

- More scenarios. More scenarios containing more security terms can be added to the game. Especially, if the scenarios are to be added to the tail, the scenarios can be more complex and compound, involving knowledge of different topics.

- Validator. This game decouples the code and the content of the scenarios to better support the extension of the game, potentially by non-programmers. Nevertheless, those who add new scenarios may still not have solid background knowledge in JSON, and may not know the tools helping them in case of errors. Moreover, a file following JSON format (which is called "well-formed" JSON file) alone does not guarantee a smooth running of the scenario: the code enforces more requirement one the content of the file (which is called "valid" JSON file instead). Therefore, the creation of a JSON schema going with JSON validator would greatly facilitate those who extend the game.

- Act tree. Some of the interconnections between the security terms have been depicted as dependencies among the acts in the game. E.g. Symmetric cryptography precedes DH, DDoS precedes DoS. Main trend games would usually create a graph called "skill tree" to

present the dependencies among skills, which turns out clearer and more absorbing than textual description. It's possible to display the acts and their connections through a tree structure. Moreover, it can be more than just a tree for the acts for one particular scenario. It can be a tree for all the terms in the game related to cyber security, for example, a mind map.

8.2 Some future prospects on pedagogic serious game

Along with the boom of digital game industry early in the 2000s, serious game begin to flourish, and it has been more and more widely accepted and researched [10]. Out of all the different categories, pedagogic serious game, with greatest growth rate [31], is increasingly accepted as a tool to boost learning. Nevertheless, pedagogic serious game is still at its early phase, and is not widely used for the official curriculums. This thesis is thus written, in an effort to push forward the use of serious game in education. Besides the encouragement for the future game designers, teachers, researchers engaged with pedagogic serious game, the thesis would like to also advocate more attention on tangential learning. The failure of some educational games as well as some traditional teaching methods has already demonstrated that intrusive way of knowledge delivery could even be counterproductive; whereas the success of "Hole in the wall" experiment and other researches on tangential learning has proven the worth of nurturing interest. Moreover, the majority of experiments on serious games take the games as standalone works, testing only the immediate effects, without consideration of their contribution to interest or the effects on other concomitant learning processes, which are also significant values brought by tangential learning. Thereby, this thesis would also call on more attention to be paid on serious game's boost to concomitant study.

Appendix A

User manual

A.1 Local installation

The game may not be permanently available over the Internet. A local installation on the PC will enable the users to access the game. Other PC or mobile phone users can also play the game without installation, as long as they are in the same local area network as the PC which serves the game. The source code of the game is downloadable at github at:

https://github.com/luyangshang/CyberCraft

To set up a server which serves the game, it's compulsory to have an Apache server first. You can choose the Apache Http Server itself, or you can choose other tools that incorporate apache server. Installing Apache Http Server itself in most cases starts from the source code, which means it tends to be more complex, with more dependencies to deal with. Therefore, this approach may bring problems for non-programmers. On the other hand, using a tool with apache inside would mean relatively a bigger installation, with modules not strictly necessary for the game also installed. However, it's a quicker solution, especially if you don't want any problems before starting running the game. The following subsections will introduce both ways.

A.1.1 Install Apache HTTP Server

The Apache HTTP Server Project itself provides only source codes, not binary. That means, you have to compile yourself to obtain the executable.

- To compilation and installation on UNIX or UNIX-like systems, refer to this page (The page is official and in detailed, so that it will not be repeated in this manual.):

  http://httpd.apache.org/docs/current/install.html

  You can use the "install" command of your OS and directly get apache2 (recommended). Alternatively, you can download the source file and compile to binary on your machine. Then you have to take care of the dependencies.

  N.B. For downloaded source file, Apache website also requires the user to verify the integrity of the downloaded files for security purpose. It provides both SHA256 hash value and PGP certificate. Except for the tools suggested on the website, one can also use the following command for a quick check of the hash values of a target file:

```
Windows 7 or later:
certiutil -hashfile <path to file>  SHA256
Linux:
sha256sum <path to file>
```

What inside the angular bracket should be replaced with the path of the file from the current directory of course. These commands calculate the hash values using SHA256 hash algorithms. Unfortunately, PGP signature is another thing, and cannot be verified in this way. Nevertheless, a matched SHA256 hash value alone is enough for the integrity check.

- To compile from source on Windows, refer to this page:

  http://httpd.apache.org/docs/current/platform/win_compiling.html

  You can build apache either from command line, or from within the Visual Studio IDE.

  After installation you can use Apache in this way:

  http://httpd.apache.org/docs/current/platform/windows.html

  It recommends running apache as a service, which runs in non-blocking mode without occupying the command line.

  Note that, if you have already installed XAMPP, you will find the contained Apache server in the folder called "apache". This contained Apache can be used the same as standalone Apache.

A.1.2 Install Tools containing Apache

There are many tools incorporate Apache like ApacheHans and XAMPP. Apache official website also recommends these tools for those who prefer executable. This part will take XAMPP as an example. After all, XAMPP features cross-platform, which satisfies users of all the current operating systems. Whereas, it should be known that other tools works also. XAMPP can be downloaded at https://www.apachefriends.org/download.html, which support Windows, Linux and OS X alike.

The default installation folder for XAMPP will be:

- Windows: C:\xampp
- Linux: /opt/lampp
- OS: X:/Applications/XAMPP/xamppfiles

This will be your root folder of XAMPP if you don't choose another path. Inside this folder, there is a subfolder called htdocs. This is where you have to copy the game (that is, the folder CyberCraft) into. For other tools apart from XAMPP, the place to copy the game may be different. Some could even put on no restriction on where the game should be. Therefore, refer to the tool's manual for the actual path.

Now, one can start the XAMPP and start the Apache server. The way varies for different operating systems, so refer to the FAQ here:

- Windows: https://www.apachefriends.org/faq_windows.html
- Linux: https://www.apachefriends.org/faq_linux.html

- OS X: https://www.apachefriends.org/faq_osx.html

When the Apache server is started, enter "localhost" (without quote) in the address bar of your browser and press enter, and you will be directed to the splash page. If by any means the server is still off, you will receive an error of connection refuse, and no page will be loaded.

Now, if you have successfully started the server, and the game is copied into the designated place, type in "localhost/CyberCraft" (without quote) and enter, you should see the game start loading. N.B. The game will try to adapt to window size before loading. Therefore, if you want to play with maximum size, you may need to load or reload the game with maximum window size.

A.2 Game beginning

When the game is loaded for the first time, you are asked to enter the name, surname and a casual 4-digit number. This name + surname + number will be your identifier throughout the game, with progress and score bound to it. When you have finished the game, and learning data collected, these triplets will be useful to identifier one player.

A.3 Tutorials

It's strongly suggested that the players start with the tutorials. In one hand, without the tutorials, the players may get stuck, even before the cyber battle really starts, not knowing what to do and how to do. On the other hand, the tutorials themselves are designed as equivalence of the later scenarios, with as much stories and gameplay as the formal scenarios. There are three tutorials.

Tutorial 1 focuses on the operations in the hall scene, which will be a preparation scene before the fight starts. The player can always quit from this tutorial through the gate, but it's recommended that he be patient enough, and finish the mission assigned by the boss.

Tutorial 2 teaches the fight in the cyberspace as defender. The defender need to use the resources to build up defense and protect the assets for a certain number of rounds. The defender has to also be careful about those attacks of DoS category, for such attacks will consume the server capacity, which makes you earn less resource in the next round. The defender should fix as much holes as possible before the intruder starts attacking in the next round. It should also be noted that, many fixes expire with time. If a fix has expired, and the defender has forgotten to re-apply it, the assets or the server being protected, will be again exposed to the related attack.

Tutorial 3 is about the fight in the cyberspace as intruder. The intruder earns a small amount of resource as the turn goes, but a better way to gain resource is by dealing damage to the assets under the protection of the defender. However, the intruder should choose the attack pattern carefully, as some of the vulnerabilities might be temporarily or permanently fixed by the defender. There are four types of offensive acts:

- The acts as "Finishing move", which directly deal damage to the assets. They are clearly the most liked type of attack. They deal significant damage to the assets, while also granting bounty for the damage dealt. Nevertheless, in the majority of cases, the intruder can only successfully perform these attacks once each round. Moreover,

attacks of the second category are usually needed to break the defenses, before attacks of this category to succeed.

- The acts without “Finishing move”, ”Dos category” or “Improvement”, has little or no damage. These attack are used to create or remove buffs on the rival (marked with “Rival +” and “Rival -” respectively), which open door for subsequent attacks, e.g. one of “Finishing move” category.
- Attacks marked with “DoS category” targets the server or the service, instead of the assets. These attacks force pressure on the server (displayed as the buff “Denial of service attacked”), which hamper the server from serving client requests in the next round, decreasing the defender’s resource income.
- Attacks marked with ”Improvement” are self-improvements to increase the effectiveness of other attacks. Some improvements are presented with buffs. Then the corresponding acts are marked with “Self +”.

There is no strict limitation on how many acts can be applied in each round as long as the resource is adequate. Nevertheless, players should end their turn when there is nothing that they can do in the current round.

What defenders can do before ending the round: fix more vulnerability with defensive acts, whether protecting the assets or protecting the server. Reapply defensive technique before the corresponding buff expires will refresh the buff length (if the new length is longer).

What intruders may do before ending the round: 1. Organize a “Finishing move”, potentially preceded by some acts of category 2. 2. Perform an attack of DoS category 3. Perform “Improvement”. Point 1 is the always the primary goal for each round. The intruders would in general catch every chance for that. Point 2 and point 3 are optional, intruder may decide whether it’s worthy, based on the sufficiency of his/her resource. Sometimes, if the intruder sees all the possible paths for the attacks are blocked, he/she could even directly end the turn.

A.4 Scenarios

With “Play” button you can enter the scenario selection scene (A.1). The scenarios are made in pairs, and will also be unlocked in pairs. In each pair, the first scenario is the one with even scenario number, where you play as an intruder(who is called Christopher); the second scenario is the one with odd scenario number, where you play as defender(who is named Godfried). As can be seen in the figure, “highest Score” will be noted down for each scenario completed. Successful completion of both scenarios of the previous pair will unlock the next pair. The player can repeat scenarios already completed, to get higher score or to perform better. The AI has its a preset action patterns for the specific scenario, but it will act differently each turn. Therefore, repeating the same scenario usually resulting in different situations where the player has to adapt himself/herself to it.

A.5 Hall

The company’s or HackIt’s hall is the place where the player receive his/her mission before he/she enters cyberspace to fight his/her rival. Besides mission briefing from the boss, players can also wander around, talking to other NPCs in the hall. These NPCs will provide the players with more

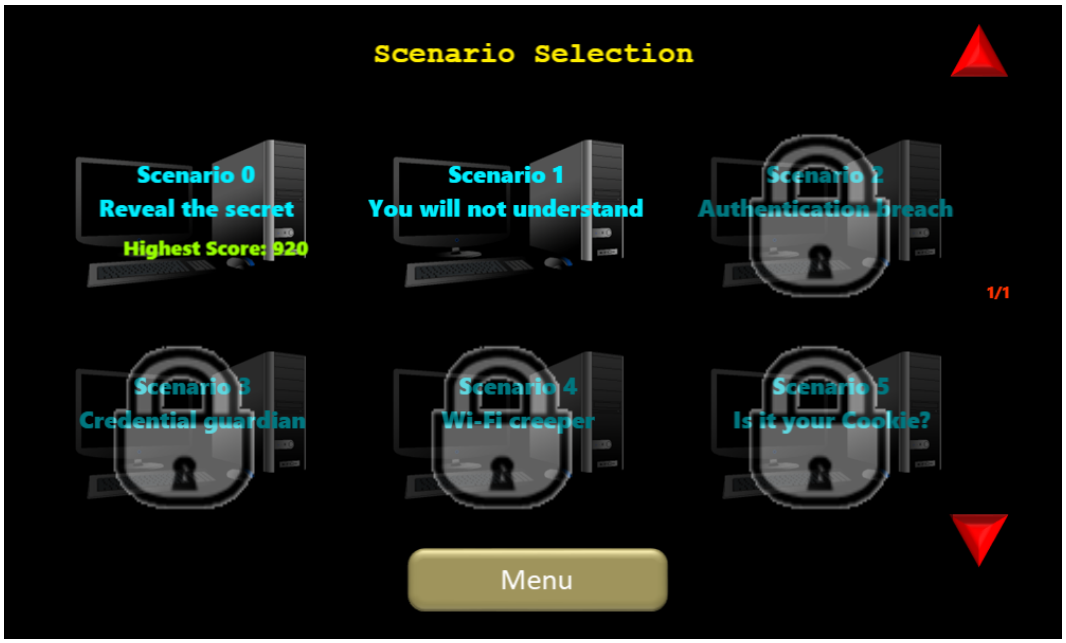

Figure A.1. Scenario Selection

information about the upcoming fight, and will occasionally give some hints and suggestions to the players. Therefore, when starting a new pair of scenarios, it's a good practice to talk to the NPCs, so as not to be totally confused when seeing the new acts.

A.6 Acts and buffs

The acts represent a security action that the character do to achieve the goal. The intruder uses cyber security attacks, denoted as offensive acts. These acts usually deals damage to the assets or impair the opponent, opening door for future attacks. The defender uses defensive techniques of cyber security, sometimes called as defensive acts, to enhance the defense of all kind, so that the service being protected will survive longer.

However, not all acts take effect immediately. Some acts could even have some lasting effects, which will remain for a certain number of rounds, or even forever. All this lasting effects are called buffs. The buffs represent that something is in use, or the state that an attack in progress. For example, a "MITM" attack performed by the intruder will, on success, give the defender a "MITM" buff. This buff, before expiration, will allow the attacker to perform a "Sniffing attack", which will deal considerable damage to the assets, which is under the protection of the defender. On the other hand, the defender can apply the defensive act called "MQV" (an authentication protocol) to obtain the buff of that name, which nullify future "MITM" attack, until the buff expires.

N.B. besides the acts enforcing buffs, there are also the acts that cleaning buffs on the character. This happens when the intruder breaks some kind of defense, or when the defender turns to alternative technique in place of the old one. Figure A.2 gives the explanation of all the possible relation between the acts and the buffs.

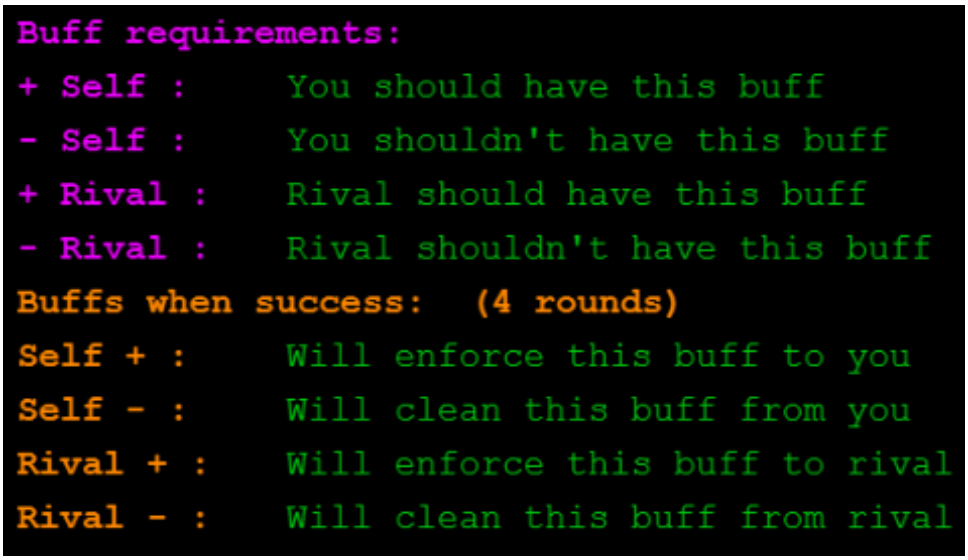

Figure A.2. Requirements on buffs and effects to the buffs that might be found on an act

A.7 Cyberspace

The fight in the cyberspace always starts with the defender's round. However, the defender need to calculate his/her resource carefully, as he/she usually does not have enough resource to strengthen on all aspects. The intruder, on the other hand, exploits these defenseless points. The main screen of cyberspace is given as figure fig:cyberspaceWithDescription.

Personal notes

If the player is a new comer, and is unfamiliar to the terms referred to in the game, he/she may need to frequently refer to personal notes. It can be opened with the book-like button on the top-left corner of the screen, or using a shortcut key N. But a better way is to use the button near the act description or buff description (shortcut key is also N). The buttons at act description or buff description will go directly to the specific entry in personal notes, which saves the player's effort to search for the term. There are also internal links where the player can navigate to other correlated terms. A chain-like button will also lead the player to external resources, to facilitate those who are interested to know more.

Except for the personal notes used for the explanation of the terms, other two things are frequently referred to. They are buffs on the character and action log.

Buffs on the character

By clicking on the portrait of the player avatar (or with a "S" key), one will be able to see the buffs on the avatar. Similarly, by clicking on the rival's portrait (or with a "R" key), the player will be able to see the buffs on the rival. Nevertheless, most of the buffs are supposed to be found on the defender. The intruder will (frequently) check the buffs on the defender, especially those positive buffs, which represent the defense that has been set up. The intruder should strike only the defenseless point or at least those that are not very well defended. On the other side, the defender will check the buffs on him/her, to know where needs more strengthening.

Action log

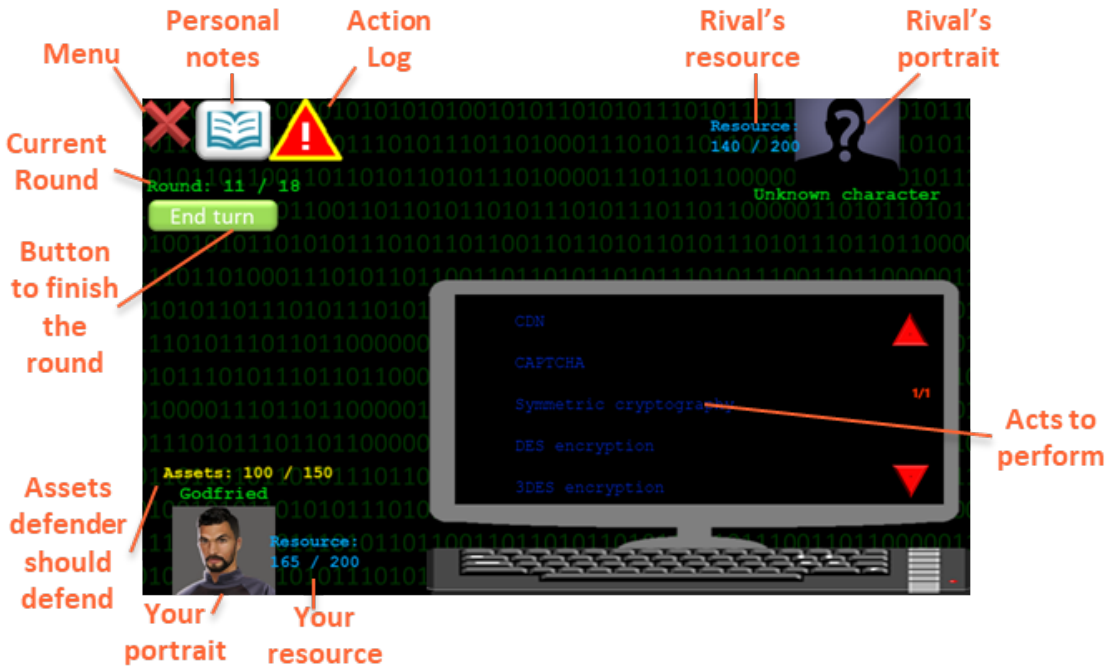

Figure A.3. The main scene of cyberspace

The other place to refer to is action log. It can be opened by clicking on the exclamatory-mark-sign on the top-left corner of the screen, just adjacent to personal notes. The action log is actually a log of the history. It notes down the acts performed by the characters, as well as whether they succeeded. This allows the player to refer to past rival acts when he/she just missed it. However, the most important value of this action log is when an act fails. The reason of why the act failed will be of great value to the player, especially the new comer who cannot remember things well. The reasons of the failure contain the presence or absence of necessary buffs, as well as the lack of luck: not all acts can be performed with 100% success rate. By clicking on one entry, the player can see the reason why the act failed. And, by clicking on the name of the act (called as Act pattern) specifically, the player can read its explanation in the personal notes.

A.8 Review

The review scene displays more or less the same as action log that the player is able to read in the cyberspace. Whereas, the review scene gives the player one more chance to revise how he/she has done during the scenario. If the scenario failed, review is an excellent place to know what is wrong, and brace oneself and prepare for one more trial.

A.9 Formulas

Advanced players and hardcore gamers are always interested in calculation formulas in the game, so that they can plan their strategy more quantitively.

A.9.1 Formula for the score

When the cyber battle ends, the outro scene will be shown, where the player's score will be calculated, mainly based on the performance noted in action log. The calculation formula is listed in figure A.4. The formulas tell that the intruder should strike fast and strike fierce. The defender, however, should concentrate on building an invincible defense, no others.

Events	Change to intruder's score
Each successful act at his round	+90
Each round defender survives (including round 1)	-50
Each damage to the assets	+5

Events	Change to defender's score
Each failed act at intruder's round	+100
Each round he survives (including round 1)	+50
Each damage to the assets	-5

Figure A.4. score calculation formulas for the intruder and the defender

A.9.2 Formula for served requests

The defender obtains resource only at the start of his turn, based on the number of client requests served and unserved. Each served (happy) client grant 10 resources, while each unserved (unhappy) client subtracting 5 resources. The following formula will be a little complex, and is intended for hardcore gamers only.

The number of served clients is calculated based on the server capacity, the number of client requests, the number of spam requests brought by the intruder, and the defender's DoS resistance. Legitimate client requests could vary each round, but only within the minimum and maximum threshold specified in the scenario's cyber file. Let's suppose the number of legitimate requests of this round is L, and the server's current capacity is C, the served requests is S, unserved requests is U. Then:

If L <= C , S = L, U = L - S = 0

If L >C, S = C, U = L - S = L - C

At the presence of intruder's spam requests, things become peculiar. Let the intruder's (total) spam request be F, defender's (total) DoS resistance be R (R = 1 - (1- R1) * (1 - R2) * (1 - R3)...). Then the effective spam request EF = F * (1 - R).

If L + EF <= C, still S = L, U = 0 — the server is powerful enough to sustain, even when attacked;

If L + EF >C, then C / (L+ EF) = serving ratio = S / L

(Effective spam request are treated the same as legitimate requests)

That is to say, S = Floor(C * L / (L+ EF))

(Round to integer is necessary. After all, there should not be "a half" client served)

There are many things that hardcore gamers can be deduced from this formula. But the most important one are:

- Increasing the capacity improves the served clients linearly when the server is destined to decline some clients.
- Continuously increasing the spam request increases the effective spam requests linearly. However, as the number of spam requests increase, boosting the spam requests results in less and less additional unserved requests. Therefore, the intruder should never try to block all legitimate requests. It just does not worth the effort.

A.10 Extra guide for the scenarios

This part is not intended for players who have not played the game.

The players are supposed to discover how to maneuver against the rival exactly. Information from NPCs, act/buff details, personal notes, action logs, previous trial of the game, or even external materials all contribute to it. Nevertheless, if the player has already retried a scenario several times, but still cannot sort out the potential paths for the intrusion or for the defense, here are some illustrations for the attack paths of the scenarios.

In the figures, red explosion indicates the offensive acts, while blue shield indicates defensive acts. Arrows starting from nowhere shows this is a possible starting point for the intruder. Shield before an explosion means that such defense can defeat the following attack, and eventually, block the attack chain. However, there are also bypasses where the intruder can still prevail despite the defense, which is illustrated as branching.

It's can be easily found that, at each round when the intruder plans attack, he should never choose a chain on which some defense is built, except there is also a bypass. On the other hand, the defender always tries to fortify on every path, and try to maintain this state for as long as possible. N.B. only clear blockings are listed in the figures. There are also other acts which increase or decrease success rate, which are not shown in the figures.

Scenario 0, 1:

The two scenarios share the same pattern. The player only plays in different roles.

From figure A.5, one can see that, there are 5 possible attack chains. As a matter of fact, the first three ends with "Sniffing attack", and deals a significant damage. "DH" is one kind of defense on these attack chains, but MITM can be used to bypass it. "MQV" and "RSA key exchange" avoid "MITM", but Quantum computing defeat them all. Then "Quantum criptography" becomes an ultimate defense on these attacks. The fourth chain, "replay attack" goes another way and deals normal damage. This path is thus taken as a second choice when the former attacks are impossible. The fifth path actually deals no damage, but it creates an effect of DoS on the rival. This effect should not be overlooked through, for the defender gets scarce resource, and may not be able to put up a fight.

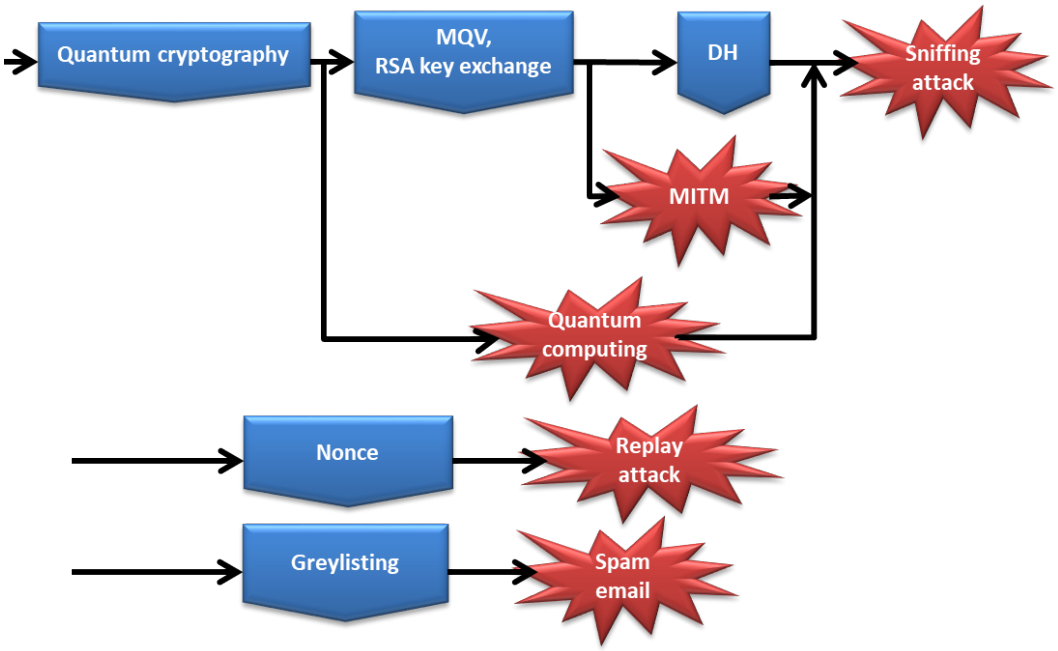

Figure A.5. Intrusion patterns and their defenses for scenario 0 and scenario 1

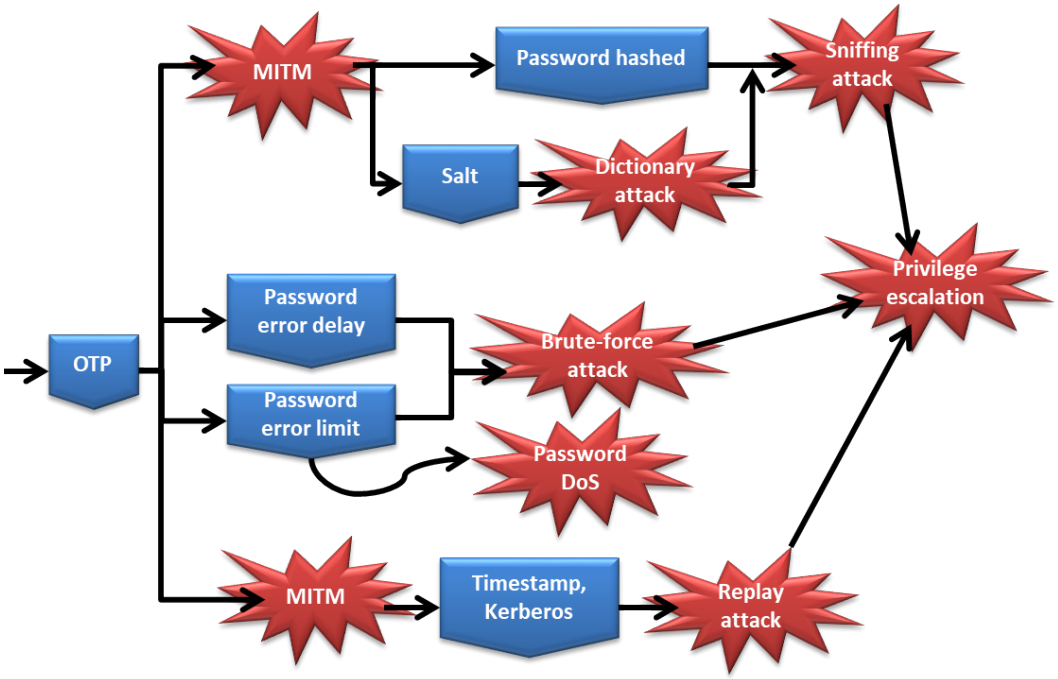

Figure A.6. Intrusion patterns and their defenses for scenario 2 and scenario 3

Scenario 2, 3:

The two scenarios share the same pattern. The player only plays in different roles.

Figure A.6 illustrates that, "dictionary attack" provides a bypass through "password hashed", but it still has its nature enemy. "Password error limit" fights "Brute-force attack, but it adversely opened door for "Password DoS" attack.

It's interesting to see that, "OTP" is a panacea to keep away from any attacks in these scenarios. Senior gamers would immediately guess that "OTP" will be greatly costly. You guess it right. It's usually used in the last rounds to guarantee you rest easy.

An unique offensive act unlocked during these scenarios called "Privilege escalation" greatly enhances the strength of the intruder. It's used only after a successful credential compromise, dealing an extra great damage to the assets.

One think not illustrated in the figure is that, you gain additional resource in this scenario by providing the functionality of single sign-on. It can be provided by a "Password manager" or "Kerberos".

Scenario 4, 5:

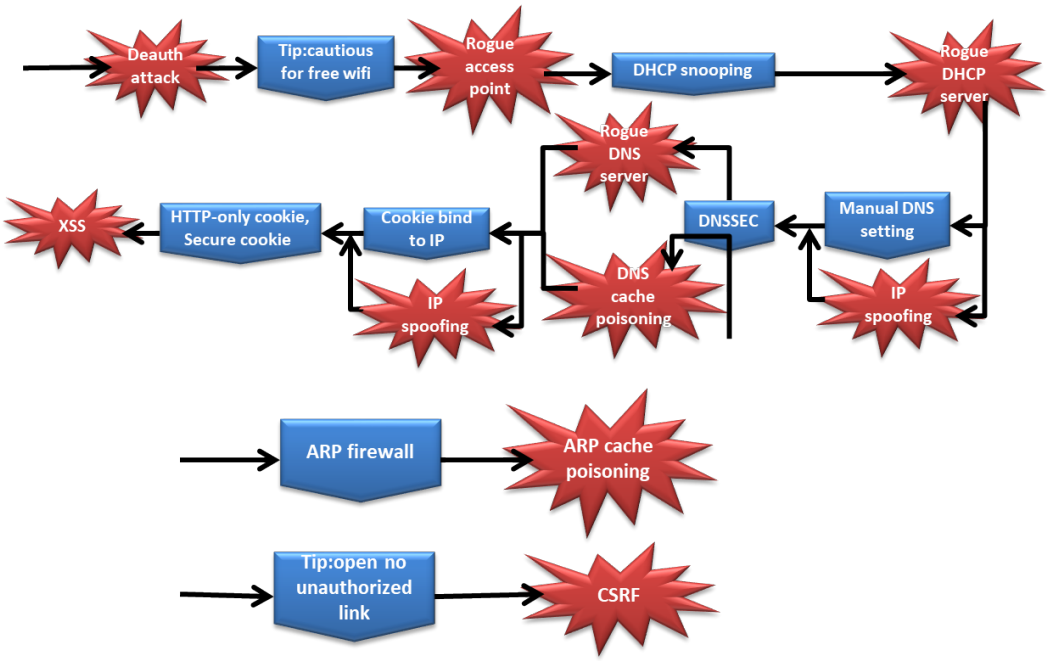

Figure A.7. Intrusion patterns and their defenses for scenario 4 and scenario 5

The two scenarios share the same pattern. The player only plays in different roles.

It's demonstrated in figure A.7 that, some attack chains have prolonged length. But don't worry, the intruder also gets partial success on the "milestones" in the attack chain (e.g. "DHCP hijacked", "DNS hijacked"). Nevertheless, for quick and fierce attack, successful "XSS" attack and "CSRF" attack are the really targets. "IP spoofing" is found here as bypass through two defensive acts, which are intrinsically IP-based techniques. "ARP cache poisoning" has its own attack chain here, but it does not come with great damage or great bounty for the intruder, so

don't overestimate it. There are four potential attack chains in this pair of scenarios. The first one starts with "Deauth attack", but the second one starts half way, at "DNS cache poisoning".

A unique act for this pair of scenarios is "(Wi-Fi) Deauth attack". It's not defendable, and it kicks the defender offline. This means the defender can do nothing in the next round. If some defensive approach expires in that round, he cannot renew it. Therefore, the defender may have to renew buffs before the expiration date, if he suspects the intruder is preparing this attack. From the intruder's perspective, "Deauth attack" is obviously a good supporting technique for overwhelming assault.

Appendix B

Programmer's manual

This manual talks about the structure of the CyberCraft folder as well as the JS code of it. For installation of the game, please refer to the first section of User manual. For information about non-code part, especially if you are non-programmers, please refer to Successor's manual in the next chapter.

B.1 Game scenes

The game majorly has 12 scenes (see figure B.1).

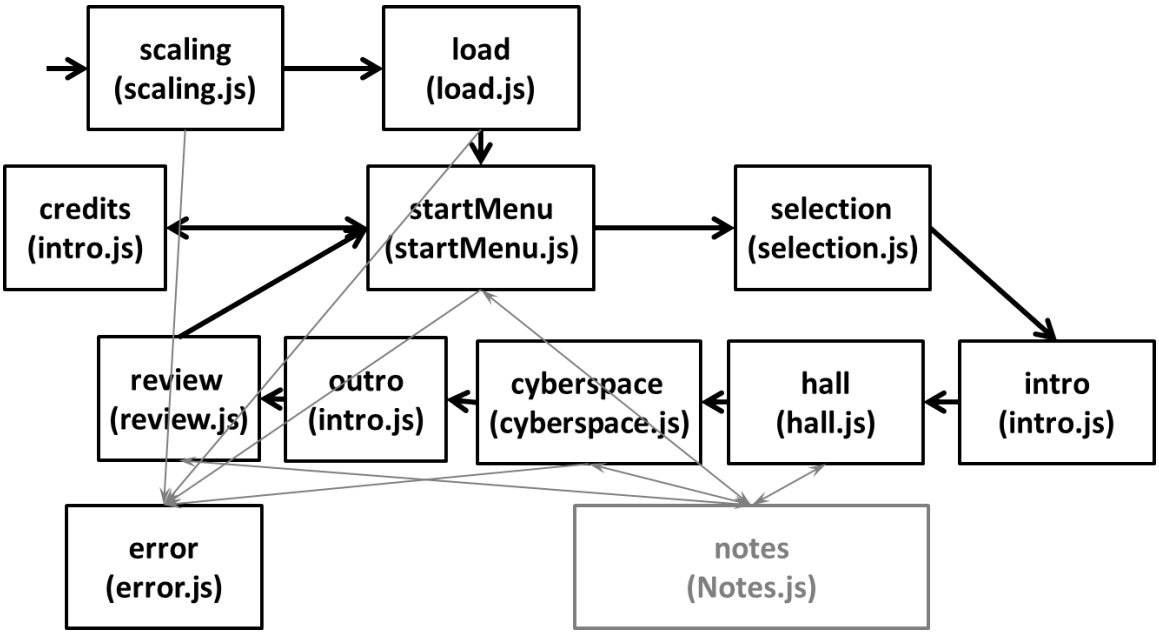

Figure B.1. The scenes of the game

The game starts with scaling (adapt to screen size and prepare loading), pass through to load, and then reach the main menu called startMenu. For a normal game procedure, the player starts at intro (general briefing), goes to hall (specific mission briefing and hints by talking to NPCs), and enter cyberspace, where the cyber fight takes place. After the fight, it goes to outro (consequence briefing), and then review (revise which action or reaction was correct and which one was wrong), and return to startMenu. In the presence of fatal errors at scaling, load or startMenu scene, it will switch to error scene. A fatal error may be caused by one or more corrupted data files. However, for non-critical errors, including the errors with data files, an error message will pop out, but the game can still go on, without going to the error scene. All the scenes except for notes are game states, and they are defined under js/states (but the three scenes intro, outro and credits share the single file intro.js). The notes scene is not a state, because the players cannot afford the state change when they want to refer to notes during the cyber fight (in cyberspace scene). Thereby, notes.js is actually an part of modules (under js/modules).

N.B. It's possible that some scenarios have no intro or outro scene. The logic flow just passes to the next scene (hall and review respectively). Besides, tutorial 1 has only intro and hall, tutorial 2 and tutorial 3 has no hall scene.

B.2 Structure of the game folder

The game folder of CyberCraft contains the following things:

assets: This folder contains all the assets used in the game, involving images, sprite sheets, sound and music.

- **images:** All the images, whether in the form of a single image or a sprite sheet.
  - **notes:** Some pictures to be used in personal notes
  - **portraits:** The portrait pictures for the characters in the hall or in the cyberspace
  - **sprites:** The sprite sheets for the characters in the hall
- **music:** long audio files used as the BGM of the game
- **sound:** short audio files used for the sound effect of the game

css: The css files related to the tools of bootstrap and alertify used in the game

jquery: The jquery code of bootstrap and alertify used in the game

js: It contains all the JavaScript codes for the game

- **modules:** The module files in charge of a certain functionality. They are used by the game states.
- **states:** The state files each is a game state.
- **bundle.js:** this is a bundle generated from all other Javscript files. index.html actually uses this file, instead of the original js files.
- **main.js:** The "root" JavaScript files. It incorporates Phaser and the state files and builds global variables.
- **parse.bat:** The BAT file (Windows) to facilitate the rebuild of bundle file from the source files. The bundle is built with the JavaScript tool browserify. Any change in the source file will not be able to affect the game until the source files are packed into the bundle file.

phaser.min.js: The simplified version of Phaser, the game engine used for this game.

scenarios: The JSON and TXT files to customize each scenarios, assets and personal notes. The content of this folder is designed as player customizable, to facilitate future extension by the players. For detailed description of the files in this folder, see Appendix C

index.html: This file is the main entrance of the game, though vast majority of the work is done by JavaScript located in js folder.

B.3 States

The Phaser game engine divides a game into different states. This provides modularity to the game development. Moreover, Phaser provides lifecycle management and sprites creation or destruction for the states. A state will be a scene from the player's aspect of view. This game is divided into 9 state files, for 11 states, all placed in the folder js/states:

scaling: The first state of the game. It scales the game canvas to the window size, and prepare (a little) asserts for the incoming load state, as well as potential error state.

error: This is the state for fatal errors when loading necessary files.

load: This state load all the game assets, as well as the JSON files and TXT files. It also reads saved game from browser's local storage.

starMenu: This is the first scene to be presented to the player, starting from where the BGM plays. The player can also read the personal notes here, without even starting a scenario.

selection: This is the scene for the player to choose a tutorial or a scenario to play. For scenario selection, the scenarios are organized in pairs, and can be unlocked in pairs, when the previous pair is completed.

intro: The file intro.js is made to manage three states at the same time: intro, outro and credits. Intro is the first scene for each scenario (except that some scenarios can have no intro). Intro tells the general information about the story setting, like when, where and who. Outro is the scene after the cyber battle, to depict the result of the cyber battle. A score will also be given to evaluate how the player has performed. Credits is the general information about the game resources. The content of intro, outro and credits are all specified in JSON and TXT files under scenarios folder.

hall: Hall is the scene after intro scene. There the player will get a better briefing of the task, in the form of a dialogue. The player will also get more information about the incoming fight, even some hints from other NPCs. The content in the hall is specified by scenarioX_NPCs.json (where X is the scenario number).

cyberspace: This is where the cyber fight commences. It is configured by scenarioX_cyber.json.

review: This is the last scene of a scenario, after outro scene. In review, the player revises his/her performance in the last fight, recognize where he/she did right and where wrongly.

B.4 Modules

Some functionality are shared between different states, some functionalities are cross referenced by other functionalities. To facilitate reuse and enhance modularity, each functionality is made into one JS file under js/modules folder, majorly in object oriented model (they are classes), whose names starts with capital letters.

Act: A value class for the Act. An act is one operation that the character can perform in order to achieve victory. The definition of an act involves many properties, some are compulsory and some has default values.

ActManager: Being one of the "Manager", actManager is the one that creates, stores, converts acts, as well as managing the learning or application of acts. Those effects from the acts like taking damage, act failure, enforcing or cleaning buffs are all managed by the ActManager. One instance is created at the start of cyberspace.

AIManager: It managers all the AI's operations like learning or applying of acts, based on the AI script written in the scenario's cyber file. AI will randomly choose an action pattern, which can be a chain of multiple acts. When it's done, AIManager will also end AI's turn. One instance is created at the start of cyberspace

AjaxFileReader: The module that exploits Ajax to load the customizable files. It's used by load state for the loading of all the JSON and TXT files except assetsTable.json (it's loaded without ajax, before load scene).

AudioManager: The manager for all the sound effect and BGMs across the scenes. It's created during load state, and alive for the whole process of the game.

BuffManager: It manages the buff database as well as the buffs on the characters. Note that the influence of the buffs on the application of an act is managed by ActManager. One instance is created at the start of cyberspace.

dynamic_text: It's a module that provides the typing-like animation for the text of the dialogues as well as that of intro and outro and some of credits.

EffectManager: It manages the major animations in cyberspace like attacking, defending, Popup words, unhappy face. One instance is created at the start of cyberspace.

GameManager: It manages a more general things of the game in cyberspace, like income, resources, server assets, round initialization. One instance is created at the start of cyberspace.

HintBox: It provides the hint box across the states, to display a hint when the players mouse hover over a clickable button. One instance is created for each state that need it.
Used by: startMenu, hall, cyberspace, review.

learning_data send: This module sends the player's learning data to the data collection server, so that information about the game's influence on the player can be obtained.

loadSave: A module managing the loading and saving of game data into the local storage of the browser.

LogEntry: A value class for one piece of log for the characters' performing an act, how it goes, and, if failed, why it failed.

LogViewer: It manages the display of action logs by cyberspace as well as review state. One instance is created at the start of either state.

Messager: A class managing the customized alert messages as well as the message queueing. Used by: cyberspace: action log, review.

MultimediaText: It manages the creation of pure text, text with typing-like animation, dialogue with typing-like animation and image-text mixed page. This class is reference by multiple states and classes.
Used by: notes, credits, intro, hall: dialogues, cyberspace: in-game dialogues, outro.

Notes: It build up the personal notes interface for the player to read. Note that although the personal notes is opened full screen, it is not designed as a state. It' only an upper layer over the original layer, so that it only "pause" the current state. In this way, the player can easily return to where they were before opening the personal notes. Otherwise, the player may think twice before opening personal notes, for fear that he/she would loss the current progress. One instance is created for the startMenu, hall, cyberspace and review, to make the personal notes as accessible as possible.
Used by: startMenu, hall, cyberspace, review.

NPCManager: It manages the dialogues of the NPCs in the hall. The NPCs has their states, which indicate which dialogue the NPC will say when he/she is clicked. The dialogue has also dependencies. e.g. NPC1 will not say dialogue 2 before NPC2 has said dialogue1. This functionality is suitable for designing a story like tutorial 1. The configuration of the dialogues is written in scenarioX_NPCs.json One instance of NPCManager is created at the start of hall state.

PersonalNotes: The value class for personal notes. One instance is created by loading, and remains for the whole game.

RecordEntry: A value class that stores the player record. A record is created only when the player has succeeded in the scenario, and it contains the whole bunch of action logs (see LogEntry) as well as other things. This class also calculates and stores the scores for the player(s). A record for double player mode will not last long, but a record for single player mode in a scenario will be stored in browser's local storage, which will be the basis for player data sent to the data gathering service.

ScriptManager: It manages the script as written in "scripts" of the scenario's cyber file. A piece of script will be active at the start of a certain round, displaying dialogue and unlocking new acts for the characters. It's an excellent tool for the tutorial or the first scenarios, in the sense that the player will see the acts unlocked progressively, and he/she will get in-game hints and guidance. Both these two properties conform to the guidelines for game tutorials.

ScrollButtons: It provides the scroll up and scroll down buttons and the indicator for the current page among the all available pages. Nevertheless, the displaying of the actual content of the page is managed by the caller function. After all, all the multipage interfaces use this class, but the way to display the content of the page diverges a lot.
Used by: notes, credits, selection, intro, cyberspace: acts on the main panel, cyberspace: buffs, cyberspace: action log, outro, review.

tile_functions: This module is used when creating the scene for the hall, from the tiled map file hallMapIntruder.json and hallMapDefender.json.

Appendix C

Successor's manual

The game supports easy extension of the scenarios, even by non-programmers. The successor's manual concentrates on the content of these extendable files as well as the data format that should be complied when creating new scenario files.

The scenario files are in TXT and JSON format. TXT files can be edited with any text editor. The JSON files can be added with text editors also, but if a tool with more support like word coloring, bracket pairing, things can be easier. One of the possible tools is notepad++, which support a large variety of data format and programming languages alike. Alternatively, online JSON editors are also a good choice for the work of JSON files specifically. One of the online JSON editors locates at http://jsoneditoronline.org/.

C.1 File structure

To add new scenarios, one needs to add files to the scenarios folder. The folder contains the following files (most of the files are bind to a particular scenario. Therefore, the scenario number in the filename will be represented as an X here):

- README.txt: It's a text file describing the content of this folder, similar to what is written here. Not expected to be changed.
- hallMapIntruder.json and hallMapDefender.json: A json file made with Tiled to illustrate the 2D map of the hall. If you want to create a better background for the hall scene, e.g. adding more decorations, you can do it here. However, if you want to add any moving objects or animations, you have to touch the code in js/states/hall.js. If you want to add new assets to the hall, you have to add one mapping to assetsTable.json and add one statement to js/states/hall.js (you should recreate the bundle before any change to the JS files takes effect).
- personalNotes.json: This file stores the entries related to the personal notes. Most importantly, all the acts and buffs activated in the game will have a link to the corresponding entry in the personal notes with exactly the same name. The personal notes will then discuss the security terms more deeply. It's strongly suggested that, whenever a new act or a new buff is added to a scenario, an entry of it to be added to personal notes also.

- credits.txt: What will be shown at the credits scene. Add credit to you for your contribution to the game.
- assetsTable.json: An table for all the images, sprite sheets and audio (sound and music), to be loaded at load scene, which covers nearly all the assets of the game. images contains the path of the image and the key assigned to it. spritesheets need also the specification of frameWidth and frameHeight. For audios, urls takes the value of a single string or an array of strings. In this game, all urls are given in single strings though.
- common_acts.json: The common acts file defines the acts and buffs used by cyberspace scene that are frequently shared across scenarios. However, for a certain scenario, not all of the acts or buffs are used. Also, the scenario can define its own acts or buffs in addition to those defined here. The scenario can even redefine some acts or buffs. All these settings for the scenario are done by the files called scenarioX_cyber.json
- scenarioX_cyber.json: The cyber file specifies the data used in the cyberspace scene for scenario X exclusively. Examples of the containing data include the player's role in the fight, the defender's assets, as well as the acts and buffs used in the cyberspace. Note that, this file takes some acts or buffs from common_acts.json, but it also define its own acts or buffs.
- scenarioX_intro.txt: (optional for each scenario) The intro file illustrates what will be shown at the introduction scene of a scenario. It involves texts as well as pictures. With it the players are supposed to see a short introduction of the story of this scenario before the game formally starts (e.g. a short briefing of time, place and characters).
- scenarioX_NPCs.json: The NPC file stores for the scenario the information of the NPCs that will be found in the hall. Most important of all, it contains the dialogues of the NPCs, as well as how the NPCs change their dialogues (states) with the player's interaction with them. The hall is supposed to be a place where the player can freely gather information related to the incoming fight. The NPCs are to give hints and recommendations for the player.
- scenarioX_outro.txt: (optional for each scenario) The outro file is similar to the intro file, only that it's put at the end of the level, showing the result of the what the player has done in the game. It also support texts and pictures. It's a good place for the revision or extension of the knowledge delivered in the game.
- tutorial2_cyber.json and tutorial3_cyber.json: The cyber file for the two tutorials teaching things about cyberspace. Tutorial2_cyber teaches how to play as defender, and tutorial3_cyber teaches how to play as intruder.
- tutorialX_intro.txt: The tutorial equivalence of introduction file.
- tutorial1_NPCs.json: The NPC file for tutorial 1 exclusively. Only this tutorial focuses on hall scene, so only tutorial 1 has NPC file.
- tutorialX_outro.txt: The outro file for the three tutorials.

Therefore, to add a new scenario, one needs to add one cyber file and one NPC file, and optionally an intro file and one outro file. The scenario number is not in any way bound to the content of the scenario. Thereby, you can renumber scenarios as you want. By doing so, you can even adding new scenarios in between old ones. However, the scenario numbers always have to be consecutive. This is because the loader stops searching at the first missing scenario. It's also recommended to add entries in personalNotes.json for every new acts and buffs you have created.

C.2 File format details

1. personalNotes.json: the note have a first entry named "Personal notes", which tells something in general about the note itself. The root element "desc" here is just that "something about itself".
   "acts" contains "offDesc" and "defDesc", which are the descriptions of offensive and defensive acts respectively. "offDesc" and "defDesc" are followed by "offensive" and "defensive", which defines the entries of offensive acts or defensive acts. Similarly, parallel to "acts", there is also "buffs" with "buffDesc", and under the other "buffs" element of it, the entries for the buffs are defined. The definition for offensive acts, defensive acts or buffs follows this format:

   name : A name of the security term (the entry). Acts and buffs of exactly the same name will obtain a link directly to this entry.

   desc : the description of it. The description can contain multiple pages, delimited by the caret character (' ^ ').

   `e.g. Page1.^Page2 sentence1. Page 2 sentence2.^Page3.`

   Each page can be a pure text page or an image-text-mixed page. The pattern is quite similar to that of the intro file described below, except that the pure text will not be displayed with typing animation.

   sees : (optional)An array of links to other entries in the personal notes. It's recommended to try to add some links to related security terms, so that a knowledge graph is built.

   url1 and [url2]: (optional) Link to external websites where the security term is further explained. The link to external resources in addition to the personal notes itself, serves as a more professional resource for the players with more motivation and more curiosity.

   It's recommended to create an entry for each new acts or buffs defined in common_acts.json and scenarioX_cyber.json (with exactly the same name), so that the players can always have a place to refer to when they get confused with the term.

2. scenarioX_intro.txt/scenarioX_outro.txt/tutorialX_intro.txt/tutorialX_outro.txt/credits.txt: these files follow the same pattern: It involves pages of pure text or image-text-mixed pages. The pure text page will be displayed with typing animation, while the image-text-mixed page will be displayed immediately. The pages will be delimited by a caret character (' ^ ').

   A pure text page contains only pure text of course. Note that the caret character (' ^ ') is reserved character, which is not supposed to be anywhere inside the page. Character ' # ' is also not supposed to be placed at the start of the pure text page.

   An image-text-mixed page is characterized by a sharp character (' # ') at the start of the page. As a matter of fact, this kind of page is composed of images and texts, each described after a ' # ' character. Here, newline as \r, \n or \r\n will be ignored by the program.

   To create an image, follow this pattern:

   `#image$<x coordinate>$<y coordinate>$<path to the image file>$<width>$<height>`

   width and height are optional. Assigning both values will shrunk or enlarge the image as you want. However, if you specify only width, height will take the same value, resulting in a square image.

   To create a text in the image-text-mixed page, follow this pattern:

```
#text$<x coordinate>$<y coordinate>$<text to create>
```

The text can also have one more argument:

```
#text$<x coordinate>$<y coordinate>\$<text to create>\$<word wrap width>
```

Note here that, all those within angular brackets should be replaced with actual values. Also, if the readers know about Phaser, they will realize that the arguments in this pattern are just aligned with the arguments to define image sprite and text sprite.

3. scenarioX_NPCs.json:
The file contains an array called NPCs, whose elements are description of each NPC in the scenario.

   name : the name of the NPC. One can also add the title and the profession can.

   sprite : the path to the picture of the sprite of the NPC

   portrait : the path to the picture of the portrait of the NPC

   x : the x coordinate to put the NPC sprite in the game

   y : the y coordinate to put the NPC sprite in the game

   speeches : the array of the NPC's speeches. A speech is a chain of sentences (potentially multi page) that the NPC will say without stopping. Though not specified here, when the game runs each NPC will have a state. E.g. at state 0, when talked to, the NPC will use the first piece of speech; at state 1, the second piece of speech

   Each piece of speech contains prerequisites (optional) and speech. speech is of course what the NPC will say at the particular state. Long speech need to be divided into multiple pages. The character ' ^ ' will be used to separate pages.
   prerequisites is the condition on other NPCs' states that need to be firstly meet, before this NPC will come to this state. This functionality is useful to build an experience of discovery in the hall scene. e.g. NPC1 asks the player to collect information by asking NPC2. NPC1 will repeat the same sentence if the player just keeps clicking on NPC1. When the player has talked to NPC2, having found the information, setting NPC2's state to the appropriate value, NPC1 will change to the next state to congratulate the player for the finding. To see a really definition, if the prerequisites is the following:

```
"prerequisites":
[
 {"npc": "npc2",
  "state": 2},
 {"npc": "npc3",
  "state": 3}
]
```

   That means npc3 has to reach state 2 (the third speech defined in speeches), and npc3 has to reach state 3, before this npc will come to this particular state.

   There is one special speech (state), which is the speech asking if the player is done with the talk in the hall, and is ready to go into the cyberspace to challenge the opponent. This particular speech is necessary for every scenario (in order to process from hall scene), and it's indicated by a grave accent (on the tilde key adjacent to number 1 in America keyboard)

at the first page of a speech. When an NPC when talked to, reaches this speech, a question like "Are you ready for the cyber battle?", together with "Yes" button and "No" button will be shown. The real question can be configured by writing a single page text after the grave accent. Extra pages of text after the grave accent will be ignored by the program. N.B. don't add more speeches (states) after this final question. If you do so, if the player clicked on the "No" button, the NPC will never ask the question again, trapping the player forever in the hall. Final point: the first NPC will come to talk to the player when the player enters the hall, even if the player has not clicked on anyone. It's quite reasonable for this NPC to be the manager or the desk clerk of this place

4. common_acts.json:

   (a) acts: it consists of two arrays. The first array defines acts for the intruder, and the second array defines acts for the defender.

   name : (required)the name of the act

   prerequists : an array of acts that need to be unlocked before unlocking this act

   learningCost : the cost to unlock this act

   desc : short, single page description. It's shown in non-scrollable popup window, so don't make it long. To put long explanation, use personal notes

   needSelfBuffs : the player has to have this buff in order to perform the act

   needRivalBuffs : the rival has to have these buffs for the player to perform the act successfully

   noSelfBuffs : the player should not have these buffs in order to perform the act

   noRivalBuffs : the rival should not have these buffs for the player to perform the act successfully

   cost : (required) the cost of performing this act. Zero-cost act cannot be performed. It will be used as prerequisite for other acts.

   successRate : the initial success rate of the act. It can be modified in game by other acts. By default, it takes 1.

   selfBuffs : the buff enforced to the player when the act succeeds

   rivalBuffs : the buff enforced to the rival when the act succeeds

   cleanSelfBuffs : the buff cleaned on the player when the act succeeds

   cleanRivalBuffs : the buff cleaned on the rival when the act succeeds

   buffLength : for how many rounds the buff will remain on the player/rival. Setting buffLength as -1 means the buff is has infinite length; 0 is erroneous; 1 means the buff expires at the end of your round; 2 at the end of rival's round. For majority of the cases, a positive even number is expected. Default to -1.

   bonus : the bounty for the intruder when the act succeeds. It's also the damage to the assets of the defender

   spamRequests : the spamRequests generated from the buffs of this act (it's normally supposed to be an offensive act enforcing one single buff to the defender). Default to 0.

   modifier : a string which modifies some properties of other acts.
   modifier format: The string can contain multiple modifications, each delimitered by the semicolon (' ; '). For each modification, it contain five parts delimitered by colon(' : ')
   <role>:<act name>:<property>:<operant>:<amount>

role==0 means it modifies for the offensive acts, while role ==1 means it modifies for the defensive acts. property takes values like successRate or spamRequests. operant takes one of the five values: '+', '-', '*', '/', '='. They represent assignment operation '+=', '-=', '*=', '/=', '=' respectively.

learnt : if the act is initially learnt at the start of the game. Default false.

unlocked : if the act is already unlocked at the start of the game. Default to true. Acts Initially locked is supposed to be unlocked by scripts.

(b) buffs: it's an array of buff definitions. The definition contains the following values:

name : (required)the name of the buff. It's possible to use the same name as the act which enforces the buff.

desc : (required)the buff description. It's also a single-page short description within a non-scrollable popup window, so don't make it long. To put long explanation, use personal notes.

capacity : the extra server capacity provided by this buff. Default to 0.

upkeep : the amount of resource lost each defender's round, due to the presence of the buff. Default to 0.

dosResistance : the resistance to DoS attacks provided by each buffs. Default to 0.

5. scenarioX_cyber.json and tutorialX_cyber.json:

(a) name: the name of the scenario

(b) serverCapacity: the initial server capacity of server to serve incomming requests

(c) serverAccessValley: the minimum amount of requests to the server each defender's round

(d) serverAccessPeak: the maximum amount of requests to the server each defender's round

(e) initialResource: how much resource the intruder and the defender have (before round 1)

(f) constantIncome: the resource obtained each time the character starts his round. This guarantees a minimum income. However, it's more suitable to let the defender gain resource from responding to the client requests rather than from this.

(g) maxResource: the players can't keep more resource than this value.

(h) maxRounds: the number of rounds of this scenario. If the defender can sustain until this time, the defender wins.

(i) assets: the initial "HP" of the defender. The defender will be penalized on assets at each credential breach. If the assets dropped to zero or less than zero before the maxRounds reaches, the intruder wins.

(j) defensive: if true, the player plays as the intruder; if false, the player plays as the defender. The AI written in the latter part of this file should be consistent with the value here: the AI plays the opposite role.

(k) doublePlayer: true: the scenario supports double player mode; false (default): the scenario will be omitted by double player mode. It's useless to set this property for a tutorial.

(l) characterName: the name to be displayed as the name of the intruder and the defender

(m) portrait: key value of the portrait pictures of the intruder and the defender

(n) commonActs/commonBuffs: the name of the acts/buffs activated in this scenario, which is defined in common_acts.json. If this scenario will modify any parameters of an act/buff, the act/buff should not be listed here.

(o) acts/buffs: the new definition and redefinition of acts/buffs for this scenario, compared to those in common_acts.json. For detailed format of definition, refer to common_acts.json above.

(p) initialBuffs: describes the buffs that the intruder or the defender already have at the start of the game. name: the name of the buff. length: for how many rounds the buff will last. length == -1 means the buff will be there from the start till the end; length >0 means the buff will be there for the first rounds; length == 0 is erroneous.

(q) AI: an array of action patterns that AI will follow.

pattern : an array of acts characterizes this action pattern. When the pattern is chosen, AI will firstly guarantee the acts are all learnt (could take multiple rounds). Then, if the resource is enough, AI will apply these acts in sequence in one round. If the resource is insufficient for all the acts, AI will wait, until the necessary resource is obtained. The acts in the pattern are usually supposed to be a combo.

chance : the chance that AI will choose this pattern out of all others. Note that if any of the acts in the pattern is not unlocked yet, the pattern is considered lock, and AI will ignore the pattern.

The sum of the percentages of all the action patterns can exceed 1. This is because for each turn, AI will only consider the unlocked patterns. The sum of unlocked action patterns can also exceeds 1. In this case, the patterns will consume the chances in sequence, meaning that the last patterns may suffer their chance decreased. This is understandable and even useful, in the sense that one can make use of this effect to reduce AI's likely hood to perform some acts that are designed for the earlier stages.

(r) scripts: It's an array of scripts, each following this format:

round : the script will be activated at the start of this round

dialogues : the dialogues to display:

- name: the name to be displayed for the speaker of the dialogue
- portrait: the portrait of the speaker
- dialogue: the (multi-page)dialogue to display. Use ' ^ ' to delimit pages.

newActs : arrays of acts to be unlocked. First array for the intruder and second array for the defender

shouldApply : (should be specified only for player's rounds) it's an array of act names. It enforces the player to apply those acts in the designated round. If the player has not applied all the acts in the round, the "End turn" button will be locked. This functionality is useful for the tutorial section, to force the player try out newly taught acts. Nevertheless, this functionality reduces player's freedom, so it's deprecated for the formal scenarios. Also be aware, when adding this constraint, one should guarantee those acts are already unlocked, and the resources are adequate. Otherwise, the player will fall into a deadlock.

About numbering of scenarios: one can create as many new scenarios as he/she want. If new scenario is to be put after the old ones, just number them increasingly. To add scenarios in the middle, however, one has to renumber those old scenarios, leaving a gap for the new one. In either case, remember that the program will stop searching for more scenarios at the first absent number, so, do number them continuously.

C.3 Error messages

If you modify the files in the scenario folder, you may encounter some error messages related to the inconsistence or wrong format. Here are some of the error messages (XX, YY or ZZ are going to be replaced with the values based on your actual situations):

- **Cannot find scenarios/assetsTable.json or the file is corrupted!**
  Maybe assertsTable.json is missing; maybe the file does not conform with JSON format. One common mistake is missing or having too much commas in an array.

- **Error! The act "XXX" activated for this scenario (scenario Y) is not defined in common_acts.json!**
  In scenarioY_cyber.json, under the element of "commonActs", you have written XXX. This means the scenario will take the act definition for XXX from common_acts.json. However, it's not found. Maybe it's because of wrong spelling (the name is case sensitive and should match exactly). Maybe you have got a wrong role: intruder's acts are always in the first array, while defender's acts always in the second array.

- **Error! The prerequisite "ZZZ" of act "YYY" activated for this scenario (scenario Z) is wrong!**
  In the definition of act called YYY from scenarioZ_cyber.json, the property of prerequisites is specified. However, at least one of the prerequisites (should be an act name) is not an act defined for this scenario.

- **Error! An act name is missing Recheck scenarioX_cyber.json**
  In the definition of an act, the name is compulsory

- **Error! The act "YYY" is missing cost Recheck scenarioX_cyber.json or common_acts.json**
  In the definition of an act, the cost is compulsory. No act should cost zero resource. Zero-cost act on the other hand means the act cannot be used. Then the act is only put in the game as a prerequisite for other acts.

- **Error! The buff "XXX" related to an act in scenario Y is not found!**
  This scenario will enforce/clean buff on the character. Nevertheless, the definition of the buff is not found.

- **Error! The buff "XXX" selected for this scenario (scenario Y) is not defined in common_acts.json**
  In scenarioY_cyber.json under commonBuffs, a buff called XXX is selected. Nevertheless, it's not defined in common_acts.json. You should add the buff definition in common_acts.json, or define the buff in scenarioY_cyber.json instead, and delete this reference in commBuffs.

- **Modifier format wrong!**
  It's not suggested to use modifier if you have other alternatives. But if you do want to use it, the modifier should be written as a string, delimited by semicolon (;) into substrings, each following this regular expression:

  ```
  ^[01]:[^:]*:[^:]*:["+""-""*""/""="]:[0-9]+\.?[0-9]*$
  ```

  For detailed description, see "File format details" above.

- **Error! The act "XXX" specified in the AI pattern for this scenario (scenario Y) is not defined!**
  Under the element pattern of AI, you have given a name that is not seen as an act name.
- **Sorry. This entry is not found in personal notes!**
  It's recommended that when you add new acts or buffs to the game, you add entries to the personal notes also. In this way, the player will have a in-game help about the new security term. Also, personal notes serves as an in-game source of knowledge for those interested to learn.
- **Error! in the file scenarioX_NPCs.json, in the prerequisites of the speeches, a reference to a npc name is not found!**
  The dependency of dialogues is based on the name of the NPC. Probably you have changed the name of the NPC, but have not changed the references to it. Remember that it's case sensitive.
- **Error! Inside scenarioX_cyber.json, under the element of "script", property "round" takes a non-positive value: "YY"**
  The script will run at the start of round YY. YY has to be a positive number of course.
- **The game just stop at loading or during the game**
  It's quite possible that there are other runtime errors that can be viewed only by checking the console. The way to open console could be different from browser to browser, but the console is usually one part of Developer tools (opened with F12). For messages at console, see below.

There are also some other console messages that can be seen only by opening developer tools (e.g. with F12 key):

- **GET http://localhost/CyberCraft/scenarios/scenarioX_cyber.json 404 (Not Found)**
  This message always comes. The game gets to know about the number of scenarios only at the first cyber file not found. That is to say, the game has found X-1 scenarios. Therefore, there is no problem here.
- **scenarioX_intro.txt not defined/scenarioX_outro.txt not defined**
  You have created a scenarioX_cyber.json, which indicates a new scenario is added. However, the intro or outro file for the scenario is not created with it. There is no problem if you don't have intro or outro files. The game will just skip them.
- **scenarioX_NPCs.json not defined**
  The NPC file is always needed. The absence of NPC file will result in an error sooner or later. If you don't know what to be written in the NPC file, just put a single person, with the only dialogue leading to the cyberspace (a single page dialogue starting with the grave accent character).
- **Error loading asset from URL assets/XXX/XXX.png**
  (This is a warning) You have added new reference to assets in assetsTable.json. However, the path is wrong.
- **Uncaught SyntaxError: Unexpected token , in JSON at position XXX**
  One of the JSON files has syntax error. It could be the newly added JSON file. A debugger may offer you a link directly to the place of the error.

Bibliography

[1] Wikipedia - Pinrciples of Learning, https://en.wikipedia.org/wiki/Principles_of_learning

[2] Wikipedia - Positive Psychology-11- Tal Ben-Shahar, https://www.youtube.com/watch?v=GDvS7WLuJnU

[3] http://www.sohu.com/a/131960875_631279

[4] Xu Liang, Bai Wen-fei. Theoretical Reviews on Mechanism of the Formation of Habit, 2005

[5] Csikszentmihalyi M, Csikszentmihalyi IS, editors. Optimal experience: Psychological studies of flow in consciousness. Cambridge university press; 1992 Jul 31.
https://books.google.it/books?id=lNt6bdfoyxQC&pg=PA18&lpg=PA18&dq=Mihaly+Csikszentmihalyi+bits+per+second&source=bl&ots=kaJxiuj1za&sig=BYHyjng4lXVqs5I7GUd2kT9U_Pw&hl=it&sa=X&ved=0ahUKEwjXjfimjdDaAhUEyKQKHYrnC_kQ6AEIZTAJ#v=onepage&q=Mihaly%20Csikszentmihalyi%20bits%20per%20second&f=false

[6] Flow, the secret to happiness, https://www.ted.com/talks/mihaly_csikszentmihalyi_on_flow/transcript

[7] Wikipedia - Active Learning, https://en.wikipedia.org/wiki/Active_learning

[8] By Kokcharov - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=44917532

[9] https://www.simplypsychology.org/bruner.html

[10] Deterding S, Dixon D, Khaled R, Nacke L. From game design elements to gamefulness: defining gamification. InProceedings of the 15th international academic MindTrek conference: Envisioning future media environments 2011 Sep 28 (pp. 9-15). ACM.
DOI 10.1145/2181037.2181040

[11] Le Compte A, Elizondo D, Watson T. A renewed approach to serious games for cyber security. InCyber conflict: Architectures in cyberspace (CyCon), 2015 7th international conference on 2015 May 26 (pp. 203-216). IEEE.
DOI 10.1109/cycon.2015.7158478

[12] Aljilani Y, Kadobayashi R. Games and Their Overlooked Potential: Facilitating Learning Using Entertainment Games. InAdvanced Applied Informatics (IIAI-AAI), 2015 IIAI 4th International Congress on 2015 Jul 12 (pp. 291-296). IEEE.
DOI 10.1109/IIAI-AAI.2015.262

[13] Csikszentmihalyi, Mihaly. Flow: The Psychology of Optimal Experience, 1990

[14] Mozelius P, Fagerström A, Söderquist M. Motivating factors and tangential learning for knowledge acquisition in educational games. Electronic Journal of e-Learning. 2017;15(4):343-54.
http://www.ejel.org/ or http://www.diva-portal.org/smash/get/diva2:1135214/FULLTEXT01.pdf

[15] Brown T, Li H, Nguyen A, Rivera C, Wu A. Development of tangential learning in video games. Department of CIS, University of Pennsylvania, PA. 2014. http://www.seas.upenn.edu/~cse400/CSE400_2013_2014/reports/07_report.pdf
[16] Floyd, D., & Portnow, J. (2012, May 10). Tangential learning: How games can teach us while we play. Extra Credits. https://www.youtube.com/watch?v=rlQrTHrwyxQ
[17] Wikipedia - Task switching, https://en.wikipedia.org/wiki/Task_switching_(psychology)
[18] Extra Credits - Tangential Learning - How Games Can Teach Us While We Play https://ed.ted.com/on/iun03qDc#watch
[19] Wikiquote - Arthur C.Clarke, quoted by Sugata Mitra, https://en.wikiquote.org/wiki/Arthur_C._Clarke
[20] Portnow & Floyd's tangential learning concept for learning contents in videogames. http://biblioteca.ucm.es/revcul/e-learning-innova/5/art387.pdf
[21] Learning Theory - Constructivist Approach, http://education.stateuniversity.com/pages/2174/Learning-Theory-CONSTRUCTIVIST-APPROACH.html
[22] Wikipedia - Duolingo, https://en.wikipedia.org/wiki/Duolingo
[23] Wikipedia - Darfur is Dying, https://en.wikipedia.org/wiki/Darfur_is_Dying
[24] Wikipedia - Flow (psychology), https://en.wikipedia.org/wiki/Flow_(psychology)
[25] Positive Psychology-3- Tal Ben-Shahar, https://www.youtube.com/watch?v=2sEyWefn0Cs&index=4&t=448s&list=PL28D16304BA57DD7E at 46'00"
[26] Tutorials 101 - How to Design a Good Game Tutorial - Extra Credits. https://www.youtube.com/watch?v=BCPcn-Q5nKE
[27] Federation of American Scientists. Summit on educational games: Harnessing the power of video games for learning. 2006
[28] Derryberry A. Serious games: online games for learning. 2007
[29] Breuer J, Bente G. Why so serious? On the relation of serious games and learning. Journal for Computer Game Culture. 2010;4:7-24.
[30] Eliane Alhadeff - SERIOUS GAME MARKET, https://www.elianealhadeff.com
[31] Adkins S. The 2016-2021 Global Game-based Learning Market. In Serious Play Conference 2016.
[32] BrainyQuote, https://www.brainyquote.com/quotes/marshall_mcluhan_391513
[33] Wikipedia - America's Army, https://en.wikipedia.org/wiki/America%27s_Army
[34] Johnson WL, Vilhjalmsson H, Samtani P. The Tactical Language Training System. Interactive Events. 2005 Jul:11.
[35] gamelearn, https://www.game-learn.com
[36] serious game store, https://www.seriousgamestore.com
[37] Breakaway, http://www.breakawaygames.com
[38] pulse!! - SGS&C (Serious Games Showcase & Challenge http://sgschallenge.com/pulse/
[39] Perfect world - SGS&C (Serious Games Showcase & Challenge http://sgschallenge.com/perfect-world/
[40] Vision - SGS&C (Serious Games Showcase & Challenge http://sgschallenge.com/vision/
[41] Aflac Trivia - SGS&C (Serious Games Showcase & Challenge http://sgschallenge.com/aflac-trivia/
[42] Ijsfontein, https://www.ijsfontein.nl/en/
[43] Pixofun, [http://www.pixofun.com/
[44] Filament Games, https://www.filamentlearning.com/

[45] Sylvester T. Designing games: A guide to engineering experiences. " O'Reilly Media, Inc."; 2013 Feb 15.
[46] Wikipedia - Operant conditioning, https://en.wikipedia.org/wiki/Operant_conditioning
[47] By Curtis Neveu - I used Adobe illustrator, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=25543294
[48] BJ Fogg's Behavior Model, http://www.behaviormodel.org/
[49] Wikipedia - Behavioural change theories, https://en.wikipedia.org/wiki/Behavioural_change_theories, Fogg Behavior Model part
[50] Wikipedia - Compulsion loop, https://en.wikipedia.org/wiki/Compulsion_loop
[51] Wikipedia - microtransaction, https://en.wikipedia.org/wiki/Microtransaction
[52] http://www.mobileui.cn/persuasive-design-behavior-model.html
[53] Wikipedia - WannaCry ransomware attack, https://en.wikipedia.org/wiki/WannaCry_ransomware_attack
[54] Adamo-Villani N, Haley-Hermiz T, Cutler R. Using a Serious Game Approach to Teach'Operator Precedence'to Introductory Programming Students. InInformation Visualisation (IV), 2013 17th International Conference 2013 Jul 16 (pp. 523-526). IEEE. DOI 10.1109/IV.2013.70
[55] Ponemon Institue LL. Cost of Cyber Crime Study: insights on the security investments that makes a difference. 2017.
[56] Yahoo says all three billion accounts hacked in 2013 data theft https://www.reuters.com/article/us-yahoo-cyber/yahoo-says-all-three-billion-accounts-hack
[57] BakerHostetler. 2018 DATA SECURITY INCIDENT RESPONSE REPORT. Building Cyber Resilience: Compromise Response Intelligence in Action. 2018
[58] Underwood M, Tesone D, D'Amico A, Doris K. The jagged joystick: Game technology for net-centric training simulations. Proc. of the SISO Spring 2008. 2008.
[59] 10 ECOMMERCE TRENDS FOR 2018, 10ecommercetrend.com
[60] Nie S, Liu L, Du Y. Free-Fall: Hacking Tesla from Wireless to Can Bus. Briefing, Black Hat USA. 2017. https://www.blackhat.com/docs/us-17/thursday/us-17-Nie-Free-Fall-Hacking-Tesla-From-Wirel pdf
[61] MAVI interactive, http://www.maviinteractive.com/immersive_cyber_security_training.asp
[62] CIRCADENCE, https://www.circadence.com
[63] Anti-Phishing Phil, https://www.cmu.edu/iso/aware/phil/index.html. Alternatively a free demo at https://oit.byuh.edu/help/anti-phishing
[64] Keep An Eye Out, https://www.seriousgamestore.com/en/serious-game/keep-eye-out-2
[65] Mavi Interactive, http://www.maviinteractive.com/
[66] CyberCIEGE, https://my.nps.edu/web/c3o/cyberciege
[67] Cyber Awareness Challenge, https://iatraining.disa.mil/eta/cyber-awareness-challenge/launchPage.htm
[68] play CyberProtect at: https://iatraining.disa.mil/eta/cyber-protect/launchcontent.html
[69] Cloud Defense, http://gronstedtgroup.com/project/intuit-cyber-security/