NTNU
Norwegian University of
Science and Technology

Data-driven Security Game

STIX and Stones

Dag Erik Homdrum Løvgren

Master of Science in Computer Science
Submission date: June 2018
Supervisor: Jingyue Li, IDI
Co-supervisor: Tosin Oyetoyan, SINTEF

Norwegian University of Science and Technology
Department of Computer Science

Problem Description

Name	Dag Erik Homdrum Løvgren
Faculty	Faculty of Information Technology and Electrical Engineering
Department	Department of Computer Science
Study program	Computer Science (Master, five years)
Specialization	Interaction Design and Game Technology
Start date	Monday, 8th of January, 2018
Due date	Monday, 4th of June, 2018
Supervisor	Associate Professor Jingyue Li

Title

Data-driven Security Game.

Task Description

Gamification is seen to be useful in Agile development and has been applied to security. Examples of popular security games are Protection Poker and Elevation of Privilege.

Field results, however, show that security expertise and experience are needed for useful deliverables when these games are used. Besides, they are manual and requires more than one person to play.

In this project, we want to combine expertise and experience into a computer board game by using available data. The goal of this project is to develop a game that combines both threats and mitigations as an underlying knowledge base.

The game will be extensible and allow a player, i.e. a developer/architect/tester/etc. to play a security strategy fun game for their project or for new features they want to develop. The game can also be used to teach students information security.

Abstract

In this day and age, the field of information security is becoming more and more important. Developers need to know how to defend against new threats to their system. Unfortunately, the field of information security is not the most popular among computer science students. To address this is issue, attempts at using gamification and computer games as learning tools have been attempted. Big games like Protection Poker and Elevation of Privilege have attempted to engage and interest students, but have so far not been tremendous successes.

This thesis attempts to identify the requirements for creating such a game, and to implement it as a fun and engaging game that provides its users with up-to-date, real-world knowledge of information security. To ensure that the needs of the users are in focus, they are involved in the process from the very start, and they help shape the development of the game. The game is implemented as a data-driven computer game, meaning it continuously fetches data from online sources to stay up to date with any new information. This is completely autonomous, and requires no manual labour.

To test how well this game is able to solve the problem, a user-testing session is performed. Results showed that the game had great promise. The users liked the concept and perceived that they improved their knowledge of information security after they played it. Though the concept was well received, several issues like information overload and missing features were also discovered.

The game implemented in this thesis is not the next great educational game, but it proves the concept is valid, and that with more development and polish, it could become a success. This thesis outlines what changes could be done to this version of the game to improve it, and outlines suggestions for what features could be implemented in the future, to take the game to the next level.

Sammendrag

Nå om dagen blir feltet informasjonssikkerhet viktigere og viktigere for utviklere. De trenger å vite hvordan de skal beskytte systemet sitt mot nye trusler som stadig dukker opp. Dessverre er det slik at det ikke er det mest populære emnet for data- og informatikk-studenter å velge. For å løse dette har det blitt gjort flere forsøk med å bruke "gamification" og dataspill til å engasjere brukerne. Store spill som Protection Poker og Elevation of Privilege har forsøkt på dette, men har så langt ikke lykkes i stor skala.

Denne masteroppgaven forsøker å identifisere og kartlegge kravene for å lage et slikt spill som kan lykkes. Spillet må kunne servere brukerne reell og oppdatert data om informasjonssikkerhet. For å forsikre at brukernes krav og behov er i fokus blir brukerne involvert fra start til slutt av utviklingen av spillet. Spillet blir implementert som et datadrevet dataspill, altså et spill som alltid sørger for å hente ny informasjonssikkerhet-data fra oppdaterte kilder på internett. Dette systemet er helt automatisk, og krever ikke noe manuelt arbeid.

For å teste om spillet har klart å løse problemet ble brukerne involvert i en testrunde av spillet. Resultatene viste at spillet hadde stort potensial. Brukerne likte konseptet og opplevde at de økte sin kunnskap om informasjonssikkerhet etter å ha spilt det. Selv om konseptet ble godt mottatt, ble flere feil og mangler med spillet oppdaget.

Spillet implementert gjennom denne masteroppgaven er ikke det neste store dataspillet om informasjonssikkerhet, men det viser at prinsippet er solid og at det med mer tid og utvikling vil kunne bli en suksess. Masteroppgaven avslutter med å vise endringene og forbedringene som kan bli gjort for å forbedre spillet, og gir forslag til ny funksjonalitet som kan bli implementert i fremtiden for ta spillet til neste nivå.

Preface

This thesis is the culmination of my five years as a computer science major student at NTNU, studying interaction design and game technology. With this thesis I am able to do something I have been dreaming of for a long time: to develop a full-fledged video game, entirely from scratch. This thesis was written during my last semester at NTNU, from January to June 2018.

The idea behind the game is to entice more people's interest in information security. These days, this is a hot topic, and it has truly been fun being a part of the spearhead piercing these new and exciting waters.

Writing this thesis has been a great learning experience, and at times frustrating and stressful. Fortunately, I have had good help from my supervisor Jingyue Li, and co-supervisor Tosin Oyetoyan.

Firstly, I would like to thank my supervisors for excellent guidance, many good pieces of advice, and valuable discussions about the problem and my proposed solution. Secondly, I wish to thank the many people who responded to my surveys and participated in user-testing the game, without whom there would not be any thesis. I would also like to thank the people who have taken the time out of their busy day to read through this not-so-small wall of text: your feedback was invaluable. Lastly, I would like to thank my good friends and classmates at NTNU, whom I have continuously drawn upon for motivation and support throughout the semester.

I hope you enjoy your reading, and should you feel the urge or need to look at or try the game produced during this thesis, you are more than welcome to download it at https://github.com/dagerikhl/ddsg-docs/tree/master/Data.

Dag Erik Homdrum Løvgren

Monday, 4th of June, 2018

Trondheim, Norway

Table of Contents

List of Tables

List of Figures

Abbreviations

API Application Programming Interface. 34, 43

CAPEC Common Attack Pattern Enumeration Classification. 13, 17, 33, 34, 41, 53, 62, 70, 71, 81
CIA Confidentiality Integrity Availability. 34
CWE Common Weakness Enumeration. 13, 88

DdSG Data-driven Security Game. 2, 3, 12, 13, 16, 17, 42, 45, 57, 58, 60, 62, 64, 65, 69, 70, 72, 73, 77–79, 81, 82, 85
DotA Defence of the Ancients. 15
DSRM Design Science Research Methodology. 2, 3

EIR External Interface Requirement. 32, 33
EoP Elevation of Privilege. 1, 7, 22, 25, 30, 71

FR Functional Requirement. 32–39, 70, 71

GUI Graphical User Interface. 62

HD Hero Defence. 13, 16
HUD Head-up Display. 53, 83

IEEE The Institute of Electrical and Electronics Engineers. 32

JSON JavaScript Object Notation. 34, 42, 43

NPC Non-player Character. 14
NTNU Norwegian University of Science and Technology. 5, 20, 21
NVD National Vulnerability Database. 13, 88

OS Operating System. 33
OWASP Open Web Application Security Project. 7, 34

PP Protection Poker. 1, 7, 8, 22, 25, 30

RPG Role-playing Game. 13, 14, 16, 28, 29, 31

SQL Structured Query Language. 88
SRS Software Requirements Specification. 3, 32, 45, 57, 70
STIX Structured Threat Information eXpression. 11, 34, 42, 60
SUS System Usability Scale. 58, 63–66, 77, 78

TD Tower Defence. 11, 13, 15–17, 24, 26–31, 45, 54, 62, 71, 72, 75, 79, 87, 89

UI User Interface. 17, 26, 31, 32, 35–38, 67, 74, 77, 83, 85, 86
UML Unified Modeling Language. 42

Chapter 1

Introduction

1.1 Research Motivation

The field of information security is a narrow one, and it has a fairly high threshold of knowledge to get into. As of now it requires a lot of work, and very few choose to make this their main focus of study or their career. For many software developers, designing and implementing information security in their projects is simply something they do because they *have to* do it. They neither understand nor want to understand the knowledge behind it. Often, this aversion to information security starts when they are students and are first introduced to the concepts.

Gamification has on many occasions proven itself as a useful tool for teaching. It has also been tried within the domain of information security knowledge. Examples of popular games in this domain are Protection Poker (PP) and Elevation of Privilege (EoP). But field testing these products have uncovered some problems: they require information security expertise and experience when being used; they are manual tabletop games; they require more than one person to play. All of this severely restricts who, when, and where these games can be played. Additionally, any new information about information security in the world takes a *long* time to reach the users of the product, as board games takes a long time to update and deliver.

This thesis aims to solve these problems with an educational game about information security and formulates this research goal to that end:

> *"To elicit the core concepts and ideas for developing intuitive and fun data-driven information security games that could positively impact information security education for amateurs and beginners in information security."*

1.2 Research Questions

Research Question One	What are the requirements of a computer game for teaching information security?
Research Question Two	How can I make this game data-driven so no human labour is needed to update or add new information?

Research Question Three How can I best implement a game to be fun and provide educational benefits to the users?

1.3 Main Contributions

The main contributions of this thesis will be to identify what the requirements for teaching information security through computer games are, and to design and implement a computer game for that purpose.

There are many factors which I think will set Data-driven Security Game (DdSG) apart and could give it an edge compared to other games that currently exists, and that try to solve the same problem.

Firstly, it is going to be a computer game. This will enable the application to be easily updated and delivered to the users. As fast as the domain of information security evolves, I believe this will prove essential, and help ensure that relevant knowledge is taught.

Secondly, the idea is to create a game driven by the data, hence the name. What this means is getting the data straight from current sources of information security knowledge and using this data in the game, dynamically. This ensures that the information presented to the users are up to date with what the real-world threats are at the time.

Thirdly, I want to avoid any hard restraints on when, where, and with whom you can play this game. Earlier games often require information security experts and multiple players. I want DdSG to be available to anyone with a little bit of spare time and any slight interest in information security.

Finally, to do all this it is paramount to deliver an intuitive, fun, and engaging experience that draws the users in and lets them have fun while learning about information security. It is my, perhaps ambitious, hope that this game will contribute to drawing more people into information security; people who have an interest in information security, but lack an engaging and fun way to learn about it.

1.4 Thesis Outline

This thesis will employ the Design Science Research Methodology (DSRM), originally described by Hevner et al. (2004). This methodology describes six activities for developing an artifact for information security (Peffers et al., 2007, pp. 52-56):

1. Problem identification and motivation.
2. Definition of the objectives for a solution.
3. Design and development.

4. Demonstration.
5. Evaluation.
6. Communication.

This methodology is chosen because of its well-defined structure and research that supports it as a good choice for developing information security projects, as shown by Peffers et al. (2007), who case-tested and evaluated the method on four different cases. Peffers et al. states that:

> *"[DSRM] is consistent with prior literature, it provides a nominal process model for doing DS research, and it provides a mental model for presenting and evaluating DS research in information security."*
>
> *— Peffers et al., 2007, p. 46*

Furthermore, it is also stated that:

> *"[DSRM] effectively satisfies the three objectives and has the potential to help aid the acceptance of DS research in the information security discipline."*
>
> *— Peffers et al., 2007, p. 46*

This thesis is structured to conform to the six activities defined by DSRM.

Chapter 2 and Chapter 3 seeks to discover what current solutions exist and what benefits and improvements this thesis can provide compared to them, i.e. how this solution can be designed to be better than previous attempts at solving this problem. After this, Chapter 4 outlines the general idea for the game.

To develop a computer game, or any software project, it is necessary to first identify and document its requirements. In Chapter 5 the core requirements for this game are identified: who the users are, and what the users' needs and dispositions towards the game are. At the end of the section a Software Requirements Specification (SRS) for DdSG is produced.

The idea behind DdSG is to create a data-driven game that is fun and educational. Chapter 6 shows how this is done in this project. It shows both the implementation of the data-driven back-end server, and the implementation of the front-end game client. Here, both the high-level architecture and the detailed design for each part of the system is described.

Chapter 7 documents a usability testing session. This session aims to demonstrate the solution to the users, and receive feedback from them about what the good aspects are, and what the bad aspects are. The results of this will then be used to evaluate the success, or lack thereof, of this proposed solution.

In Chapter 8, I will be discussing the research conducted in earlier parts of the thesis in light of the research questions.

To communicate the results and their impact for this project and future work, Chapter 9 concludes the thesis with a summary of the research questions based on the results from the evaluation and

the other previous parts of the thesis. It also discusses the shortcomings of DdSG and tries to propose concrete ideas to how these could be improved in the future, based on the results of user-testing the game.

Chapter 2

Previous Work

This master thesis is a continuation of my specialization project (Løvgren, 2017) done as part of my education at Norwegian University of Science and Technology (NTNU) autumn 2017. The specialization project has sections detailing related work, discussion about game-type, game design and concept, system architecture and design, technologies to use, and a reference to this thesis in its future work.

This thesis builds upon the research done in the specialization project, and this is especially true for the related work and game-type sections. The specialization project report is extensive, and to include it in its entirety would make this thesis unnecessarily verbose and cumbersome to read. Therefore, I will try to condense and compact the important parts from the specialization project report and include them in this thesis, citing the report where more extensive data can be of interest.

Chapter 3

Related Work

3.1 Prestudy

Much of the previous specialization project consists of an extensive related work research section, which will be the basis for this section. I have removed parts not of the greatest interest, and summarized the parts I deem important for answering the research questions posed in this thesis.

The complete section about related work can be found in my prestudy report (Løvgren, 2017, pp. 7-22).

3.2 Existing Games

3.2.1 List of Games

There are a number of existing games for teaching information security knowledge out there, but none that have really become a grand success as of yet. Among these are:

- PP.
- Control-Alt-Hack.
- Cyber Threat Defender.
- EoP: the Threat Modeling Game.
- Open Web Application Security Project (OWASP) Cornucopia.

In my thesis, I will limit myself to looking at two of the more popular games of this type: PP and EoP.

3.2.2 Protection Poker

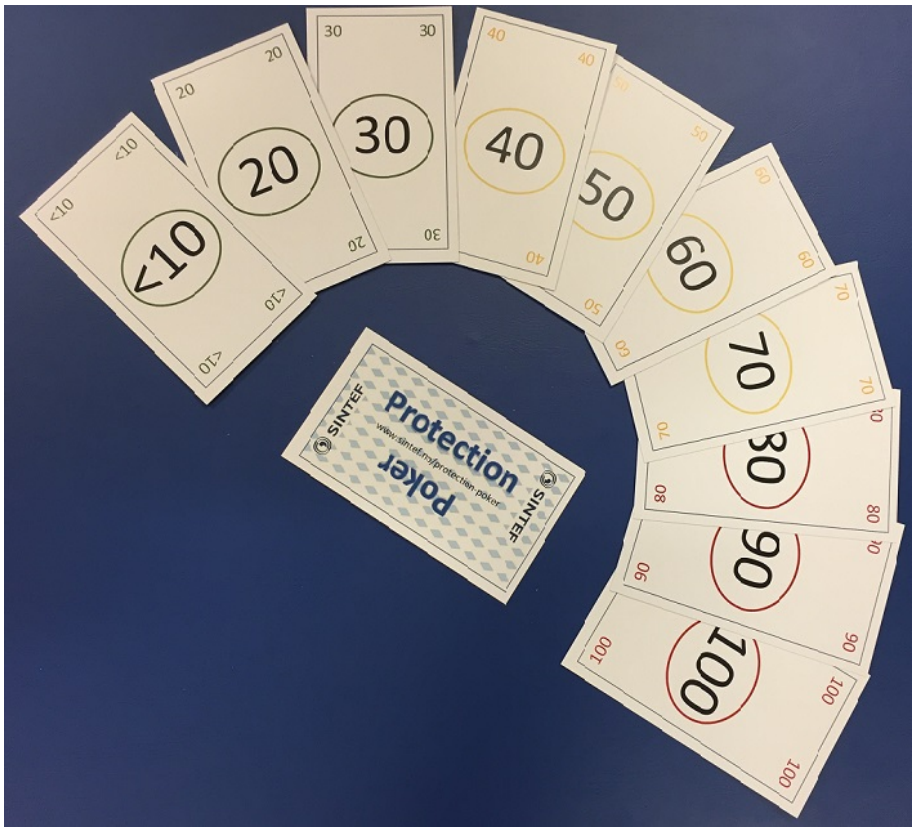


Figure 3.1: *Protection Poker game.*

PP is a game for security risk assessment created by professor Laurie Williams at North Carolina State University (Williams, Meneely, and Shipley, 2010). The game resembles the popular planning game Planning Poker for agile teams.

The purpose of this game is not as much to teach people information security knowledge, as it is to help teams already developing projects to reflect and plan their risks.

The game is intended to be played by a software development team during some form of iteration planning meeting. Gameplay consists of letting players discuss some potential security risk in their system, or to discuss a new feature they are planning to implement and its potential risks. After the discussion, all players vote with a certain amount of points on two things: ease of attack (eoa) and value of affected asset (voaa). The security risk is then found by this formula:

$$risk = average(eoa) \times average(voaa)$$

When this has been done for all features and/or potential risks to the system, the team prioritizes what they should plan for the next iteration according to what is the most significant risk to their system.

We can elicit that this type of game requires several players, and everyone must at least be somewhat familiar with information security concepts to be able to discuss the risks. Often an expert is also present as a moderator/facilitator.

However, it is also quite simple; the rules are basic and the artifacts require to play are simple and easily home-made.

3.2.3 Elevation of Privilege

Figure 3.2: *Elevation of Privilege game.*

This game is a game created by the giant Microsoft to teach and reason about threat modeling (Shostack, 2014). The game is intended to be played with a real or imaginary system with potential threats and mitigations in mind.

They utilize a familiar concept by basing the game roughly on a standard deck of cards. The deck is altered to have six different schools of threats, following the STRIDE[1] principle (Xu et al., 2012, p. 527), instead of suits, and each number/face has one threat within that school described. The game is played by players taking turns playing one card. After a card is played, discussion about whether or not this threat is mitigated in the system they are considering ensues. Throughout the game, points are awarded for playing threats, playing better threats than others, and other factors.

[1]Spoofing, tampering, repudiation, information disclosure, denial of service, and elevation of privilege.

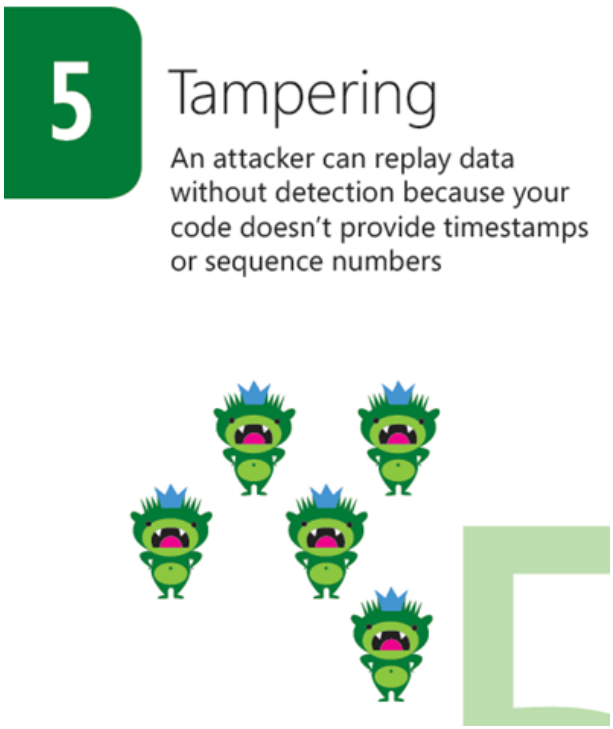


Figure 3.3: *Elevation of Privilege example card — The 5 of Tampering.*

The game requires information security experts to be present, either as players themselves, or as facilitators. The game is also played with several people, and can not be played alone.

The theory behind the game is well thought out and based on many well-defined and sound principles. Unfortunately, this does not save the game from several problems:

- Unnecessary complex rule set.
- Too many complex game artifacts.
- Boring gameplay.
- Requires a lot of information security expertise or experience to play.

While the game may work well for teams consisting of people familiar with or experts in information security that wants to threat model their system, it does not work very well for newcomers trying to get into and learn about information security.

Chapter 4

Preliminary Design of the Game

To achieve the goal of this thesis and create a game to help teach information security knowledge, my idea is to create a Tower Defence (TD) driven by data from well known sources of information security knowledge. These data sources are discussed in Section 4.1. The rationale for the TD game-type is detailed in Section 4.2. The game is supposed to be single-player, and be available anywhere and anytime, even offline.

So the idea is to create a two-part system: one front-end game client for the players, and one back-end server for the data sources. There will be many game clients, all communicating with a singular server application. This adheres strictly to the client-server architectural pattern (K. Davis, Turner, and Yocom, 2004). The back-end server will be hidden from the users, they will only see and be made aware of their game client.

The server will fetch data from the sources at a regular interval, and process this data to create usable game entities for the game client, such as attacks, mitigations, and assets. The server will make these entities available on a network for the game clients to fetch.

As to the game client, I will create the game using the Unity game engine (Unity Technologies, 2018a). The game will present the information security knowledge data to the player as game entities based on the Structured Threat Information eXpression (STIX) concept (S. Barnum, 2014, p. 5), with some additions. Specifically, these game entities will be presented to the users:

- Attack patterns.
- Course of actions, as ways to mitigate an attack.
- Assets.

Table 4.1 shows what real-world information security concept these entities represent.

Additionally, Table 4.2 shows some more real-world concepts represented in the game, but are not one of the core game entities.

Table 4.1: *Translation of real-world information security concepts to game entities.*

Real-world information security knowledge concepts		Game entity
Attack pattern Attack Threat	⟷	Attack pattern
Course of action	⟷	Course of action
Mitigation Solution	⟷	Implemented course of action
Asset System asset Activation zone	⟷	Asset

Table 4.2: *Translation of real-world concepts to game concepts.*

Real-world concepts		Game concept
System	⟷	Board
Injection vector Remote access point External interface Periphery systems System access point Interface	⟷	Simplified to one of: Client, Network, or Server

The game will be structured so that the player will have to defend his or her assets from incoming attacks based on attack patterns. To avoid his or her assets being compromised, the player has to take certain courses of action to deflect and defend against these attacks.

Figure 4.1: *STIX game entities.*

Throughout the game, the player will learn what course of actions are usable against what attacks, and what assets are compromised if the attacks are successful.

These game entities form the basis of the educational content that DdSG will serve its users. In the future, it should be possible to add other entities. Examples of other types of entities that can be added are vulnerabilities, weaknesses, or threat actors.

Adding other entities like this would likely require using additional data sources, and this is discussed in the next section.

4.1 Data Sources

To keep the data in the game up-to-date with real-world information security knowledge it is important to choose some data sources that are kept up to date, and reflect the current information. For this purpose I have chosen the MITRE source Common Attack Pattern Enumeration Classification (CAPEC). For a detailed discussion of why I chose this source, see Løvgren (2017, pp. 54-57).

In the report, Common Weakness Enumeration (CWE) and National Vulnerability Database (NVD) are also mentioned as potential sources. I have chosen to not use CWE because it adds a whole new game entity, and thus a lot of complexity, while providing little value to the game. CAPEC has so much data already, and adding weaknesses from CWE in addition will not provide any revolutionary new features to the game. In a very simplified model, one could view the mitigations connected to the attack patterns in CAPEC as the absence of weaknesses. That is why the added value of CWE is limited.

I have chosen not to use NVD because the structure of the data from NVD is so fundamentally different from the data from MITRE, making it much more difficult to implement fetching from both sources.

This becomes especially difficulty because of the data-driven nature of DdSG, where all the parsing of entities from the data sources has to be autonomous and require no manual labour.

In short, CAPEC provides information about the attack patterns that are used for attacks, as well as information about possible mitigations of these attacks, and assets they intend to compromise.

4.2 Game-type

A more in-depth discussion about the different game-types can be found in my prestudy report (Løvgren, 2017, pp. 23-32).

4.2.1 Game-types Considered

To create this game, a couple of different types of games were considered: Role-playing Games (RPGs), TDs, and Hero Defences (HDs). The choice of these three game-types was based on what games are popular at the time of writing this thesis, what game-types have been consistently popular throughout the history of games, and what game-types are most appropriate for abstracting information security knowledge onto.

RPGs have been around a long time. Early examples include the very popular game Dungeons & Dragons, which has helped shape many of the current games today, such as Baldur's Gate,

depicted in Figure 4.2.

In an RPG, the player controls the actions of one or more characters that they role-play as in one universe or another. This is often an imagined universe. RPGs are often connected to a rich lore and a greater narrative of the universe it takes place in. Popular themes within the genre is typically fantasy and science fiction.

Figure 4.2: *Baldur's Gate Enhanced Edition showing the player with a six-person party engaged in combat with computer controlled Non-player Characters (NPCs).*

Source: Personal screen-capture 2017-10-01.

Some of the key aspects of RPGs are character development, narrative, quests, and NPC interaction. All of these are used as tools to further immerse the player in the game, and are paid special attention to when developing RPGs. Character development allows the player to evolve their character, becoming better and more powerful as they go along. The narrative is used to create an intriguing story the player can get invested in by appealing to their curiosity. Quests are specific tasks in the game which the player must complete to further the narrative. Completing small or large quests gives the player a sense of achievement and boosts their self-esteem. NPC interaction allows the player to even further immerse themselves in the world because the fantasy becomes more real with simulated and realistic people in it.

Figure 4.3: *Tower Defence King gameplay, where several Structures have been erected to defend against the advancing monsters.*

Source: http://mobims.ru/wp-content/uploads/2017/04/6-1.jpg [Accessed 2017-10-25].

The main idea of TDs is letting the player defend their base against enemies with some form of defences they can place on the board. The base usually has a set amount of integrity, that is gradually compromised when enemies are able to get past all the player's defences and enter or attack the base. The entire game takes place on one or more similar and simple boards, as can be seen in Figure 4.3. In TDs, players might not be able to beat a certain level the first time they play it. TDs usually offer the players the option to play again, and they can employ a different strategy to beat the level. This encourages learning and adapting.

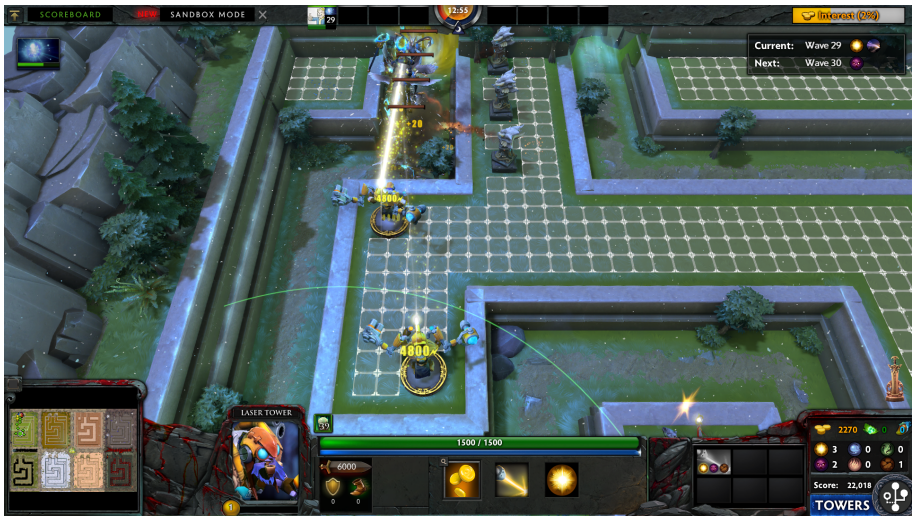


Figure 4.4: *The Defence of the Ancients (DotA) 2 adaptation of Element TD.*

Source: http://www.hernantas.com/wp-content/uploads/2016/03/dota2-2016-03-01-20-07-04-90.bmp [Accessed 2017-10-25].

While the core elements of TDs are quite simple, many TDs implement some unique feature in their game to set it apart from all the others. An example of this is Element TD, which can be

seen in Figure 4.4. In the lower right corner, six different elements can be seen. In Element TD, these elements can be gained from killing difficult bosses. Each element gained gives the player new options for defences in the form of new towers or new upgrades to existing towers that were previously unavailable.

Figure 4.5: *Haunted Island HD game.*

Source: https://static.taigame.org/image/screenshot/201601/hero-defense-haunted-island-12.jpg [Accessed 2018-03-21].

HDs are similar to TDs in many ways. They are based on the same idea of a base that has to be defended and enemies that are attacking it. The difference is that instead of defensive structures of some kind, such as towers, the player has to defend their base with a hero, i.e. a character they play in the game. Usually the player only has control over this player, and not a multitude of defences as in TDs. Because the players control a character that they can develop across the course of the game, we can look at HDs as a kind of mix between RPGs and TDs, or as TDs with the concept of role-playing a character borrowed from RPGs.

4.2.2 Choice and Rationale

RPGs were considered because of their popularity and ability to completely immerse the player in the story. They have been around as long as we have had computer games, and have consistently proved to be a popular choice. However, they also require a lot of effort and creativity to get right. The success of the game is completely dependent on the designers' ability to write and implement an engaging story. This is not something I, as a game designer and programmer, could guarantee any success with, and the success of DdSG would have been left largely up to chance.

HDs have their merits and are similar to TDs in many ways. The main difference is that the level of immersion is heightened in a HD, where the RPG element of role-playing a character, a

hero, is introduced. While this can be very fun in many games, and provides a fun and similar alternative to TDs, it does not map as well with the concepts in information security knowledge.

TDs however, match quite perfectly with the concepts in information security knowledge as is. TDs' defences, bases, and enemies are easily compared to the concepts we have in information security knowledge of mitigations, assets, and attacks. The TD game-type is also fairly simple, has well-defined mechanics, and does not require the designer to come up with or plan completely new features and mechanics unless they want to. All of this enables a game to be implemented as a TD with relatively little effort, compared to many other game-types, and since the game-type is well established and has well-defined game mechanics it should make quantifying what the faults and successes of the game are in testing.

4.3 Preliminary Game Design Sketches

The basic idea of DdSG can be seen here. The sketches show the game board, and does not take into account the User Interface (UI), only the game board and entities.

The sketch shows how the board could look like when course of actions have been taken, defences have been placed, and attacks have entered the board but have not yet reached any critical parts of the system. The example attacks, mitigations, and assets are arbitrarily chosen, but they are connected according by the data provided by CAPEC.

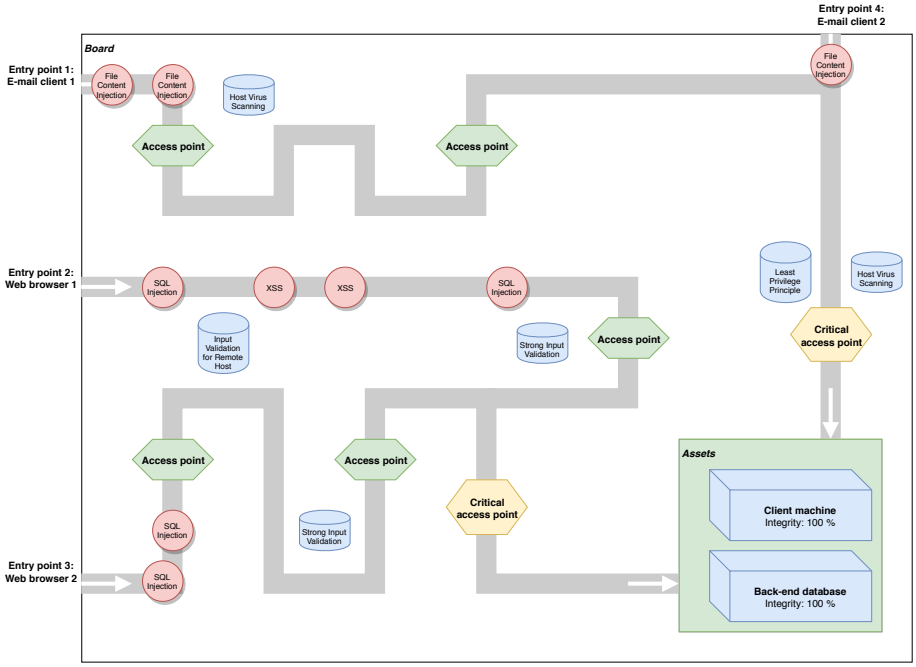


Figure 4.6: *Concept sketch including a XSS attack.*

This is a very early sketch of the idea, and the finished design does not look exactly like this, and the main concepts like access points takes on a different look and function. This will be further

explored and improved upon in Section 6.2.1.

Chapter 5

Identifying Detailed Requirements of the Game

5.1 Target Group

Following the research goal, the purpose of this thesis is to create a computer game to positively impact security education, in the classroom as well as among professionals. This alone indicates a fairly large and diverse target group. I find it necessary to narrow down this scope and focus on a smaller group for the initial development, to reduce complexity and development time. It should still be possible to expand the game and enable it for a larger target group at a later time.

Because of the nature of this problem, and the fact that its intention is to improve security education, I assume that all users within the possible target groups have at least some knowledge of computers. I assume this because they otherwise would have no interest in information security knowledge. I also assume many, but not all, of the users will have at least some experience with computer games. I assume this because it is fairly common for users with an interest in computers to have experience with games, even more so computer science students.

There are three main levels of relevant users for this game:

Amateurs Users with no knowledge of information security knowledge. E.g.: Computer science students who *have not* taken any information security course; or users with an above average interest in information security knowledge.

Beginners Users with some or moderate knowledge of information security knowledge. E.g.: Computer science students who *are taking*, or *have taken* one or more information security courses; or computer-savvy users with an above average interest and knowledge of information security knowledge.

Experts Users who are well-versed in and familiar with information security knowledge. E.g.: Information security course professors; or information security experts/practitioners that deal with information security knowledge for a living.

For the reasons stated above the second group, *the beginners*, will be the target group of this game. This way, I can make the assumption that all users of the system have some interest in or knowledge of information security knowledge. It also makes it easier to find users for testing

and evaluation, as there are information security courses with a multitude of students who might be interested in testing this game at NTNU, where this thesis is being written.

The users in this group will have the following characteristics:

Age	Primarily between 18 and 30. Outside the range is not a problem.
Occupation	Primarily computer science students.
Information security knowledge	An above average interest in information security knowledge, and/or at least some knowledge of information security knowledge.
Computer knowledge	At least familiar with computers.
Motivation	Currently looking to increase their own information security knowledge, either out of personal interest or for information security course they are currently taking.

These parameters include all students currently enrolled in the course TDT4237 - Software Security at NTNU. By viewing the statistics from the course's last semester, it can be seen that the number of students enrolled varies between 120–200 students. In addition, there may be more potential users that fits all the parameters of the target group, who are not enrolled in the mentioned course. They might have taken it before, or simply have a personal interest in information security knowledge.

I therefore estimate the target group to consist of at least 200 users, only counting potential members of the target group that are at or near NTNU in Trondheim, Norway. These are all potential users I can use and involve in the process of development to create the best possible solution for them.

5.2 User Survey

5.2.1 Purpose

In any information system development project, it is vital to involve the users early in the process, and *keep them involved* throughout the development process. Otherwise, there is a significant risk of diverging from the actual requirements and needs of the users. Involving users in the development process is in tune with one of the core motivations for agile development: to avoid diverging from the actual needs of the users.

This is why this user survey is performed at this early stage, before any detailed design or implementation has begun. Performing the survey at a later stage could lead to a necessary redesign of the system, doubling the work.

The purpose of this user survey is to collect information about what the target group wants and wishes for in a solution. I also want to collect feedback on the proposed concept to see if there are better ways to go about it, or other improvements to be made.

In addition, background information about the users' profile and characteristics will be collected, so they can be taken into account during the evaluation. They can be possible points of errors or discrepancies if they are not properly documented. This area will also collect information about the current state of information security knowledge and other relevant knowledge the users may possess.

5.2.2 Group Size

Because a survey is a very *passive* form of data collection, it requires less effort on the users' part, as well as on the data collector's part, i.e. my part. It is therefore easier to engage large groups of users at this stage, than it will be with the usability session for evaluation that will be performed later.

For this user survey, I will be trying to receive as much feedback and reach as many users as possible. The survey was therefore sent out to all users I can think of within the target group, including students taking the Software Security course at NTNU. They will not be required to answer the survey, but by sending the survey to so many users there is a very good chance that a significant number of users choose to answers it, giving enough data to base further development on. In addition to this, I will send the survey to people I know that fit the target group. I will keep these responses separate to discern what answers are given by the people I do not know from the answers given people I do know.

I estimate the survey was sent to around 50–100 users, but do not expect to receive near as many answers. Optimistically 5–10 % will answer, giving me around 10–20 data sets to help improve the concept and proceed development from.

5.2.3 Possible Error Sources and Biases

To motivate the students in the Software Security course to answer the survey, I let the participants enter a prize lottery where one student would win a gift card for use at the local cinema worth NOK 250. This could possibly motivate some students to answer the survey more than one time, providing random or untrue answers.

There is a possible bias in the second group the survey was sent to. This is due to the fact that I know these people, and they may be averse to hurting my feelings or giving too critical feedback. For this reason, this group of responses is kept separate from the others, so that this can be accounted for and reasoned about.

5.2.4 Questions

The survey will for the most part consist of quantitative research, asking simple questions that everyone can answer with little effort.

I will also open the survey for more qualitative research with some open-ended questions, if some of the users should wish to provide more detailed feedback. This type of feedback is more informative for the developer, i.e. me, but requires a lot of time to collect and analyze for such a large group. Therefore, these qualitative questions are made optional.

The questionnaire is divided into three main areas, with subareas:

1. Background: Information characterizing the user and their current knowledge.
   (a) Profile: Background information, e.g. age, gender, etc.
   (b) Level of information security knowledge: What level of information security knowledge the user currently possesses.
   (c) Game experience: How much experience the user has with computer games, and what types.
2. Quantitative: Many simple questions for feedback on the problem and the current solutions, and the proposed solution.
   (a) State of current solutions. Here, two of the more popular games for teaching information security knowledge, PP (Williams, Meneely, and Shipley, 2010) and EoP (Shostack, 2014), are considered, as the author has most experience with these two.
   (b) Proposed solution: How positive or negative the users are to the proposed solution, and what aspects are positive or negative.
3. Qualitative: A few optional open-ended questions that the user can choose to answer should they feel they have more feedback they want to contribute with.
   (a) Concept: More descriptive feedback on the proposed solution.
   (b) Their ideal solution.

The entire form and all the questions can be seen in Appendix A.1.

5.2.5 Data Collection

The user survey is shortly presented and made available to the users Tuesday, 13th of February, 2018. Data is collected by the end of Monday, 19th of February, 2018. This should give all the users that have intentions of answering the survey, enough time to answer it.

5.3 User Survey Results

5.3.1 Summary

A summary of all the results from the user survey can be seen in Appendix A.2 and Appendix A.3.

In this section I will be accounting for the results gathered from all the questions posed in the survey, in order. Where possible biases become apparent as differences between the two groups of respondents, I will shortly address what could cause this difference.

The survey was answered by 22 people from the Software Security course, and by 9 people from the group of people I know personally. This totals 31 respondents from the survey. Although not required to complete the survey, quite a few left helpful feedback and opinions in the optional qualitative questions.

5.3.2 Explanation

All questions where respondents were asked to rate their answer on a number scale, the scale goes from 0, not at all, to 5, very.

I will be referencing the group from the Software Security course as the first group, and the group of people that I know personally as the second group.

5.3.3 Background

Age The age-range of the respondents ranged from 20 to 27, with over half of them, 51.6 %, being roughly normally distributed around the ages of 23–24. The average age of the Software Security group was 23.3, and 24.0 for the other group, which is slightly higher.

Gender The males represent 83.9 % of the respondents, while the remaining 16.1 % are female.

Occupation All the respondents from the Software Security course are students. Together with the other group the majority of respondents are still students, with 87.1 % of all respondents. A few work with something else or directly with information security.

Familiarity with Information Security Knowledge When asked to rate their own familiarity with information security knowledge most rated themselves in the 2–4 range, 90.3 %, with one

respondent considering themselves not familiar, and none considered themselves experts on the subject.

How They Have Learned Information Security Knowledge While 90.9 % of the respondents from the Software Security course answered that they had learned what they know about information security knowledge in lectures, only 55.9 % of the respondents from the other group chose lectures. This indicates that maybe only half of the second group has ever taken a course in Software Security. The internet ran a close second with 87.1 % of the respondents having learned information security knowledge from it. It is also worth noting that not one respondent answered that they have learned information security knowledge from educational games.

Familiarity with Programming All of the respondents were somewhat familiar with programming, but more so in the second group. Most were at least somewhat familiar, and the bulk of the respondents, 67.7 %, answered in the range of 3–5, meaning fairly familiar.

Use of Information Security Knowledge In Programming Most respondents seldom used information security knowledge when programming information systems, mainly ranging from monthly to a couple of times a year, but it was somewhat more common among the second group than the first. Out of all the respondents 22.6 % answered they use it monthly, and 32.3 % answered they use it a couple of times a year.

How Often They Play The second group is more into gaming than the first, as no one in the second group responded that they play computer games less than a couple of times a month, and only one answered a couple of times a month. The rest of group two responded they play games at least every week, and more often. The first group is more spread out when it comes to gaming frequency, but only one responded that they never play computer games, the rest do. Among all the respondents, 61.3 % responded that they play computer games every week, or more often.

Types of Computer Games Played The types of games the respondents play are very varied, and most play more than one type of game. Among all the types of games, the three most popular types are: first-person shooters, role-playing games, and adventure games. These were checked by 51.6 %, 51.6 %, and 48.4 %, respectively.

It is also worth noting that no more than 12.9 % answered that they usually play TDs.

Familiarity with TDs Most respondents seem to be familiar with TDs, only 9.7 % said they were not familiar with them at all, and the bulk of the respondents, 77.4 %, answered that they were fairly familiar with TDs, giving their familiarity a rating in the range of 3–5.

Attitude Towards TDs The overall attitude towards TDs varies quite a bit, but does have more neutral votes in the second group, indicating they neither like or dislike TDs in general. Out of all respondents: 12.9 % answered in the 0–1 range, disliking the game-type; 48.4 % answered in the 2–3 range, staying neutral; and 38.7 % answered in the 4–5 range, actively liking the game-type.

5.3.4 Quantitative Questions

Familiarity with Existing Games for Teaching Information Security Knowledge When asked about the two existing games about information security knowledge, PP and EoP, 19.4 % of the respondents responded that they had heard of both games. Only 6.5 % had personally tried EoP, none of these among the first group, while 16.1 % said they had personally tried PP.

One respondent also answered that they had tried other information security knowledge games supplied by the OverTheWire community (Van Acker and "morla", 2018). The respondent did not state exactly which games from OverTheWire, so I assume multiple ones, as they are all closely knit.

Protection Poker When asked about how well they feel PP taught them information security knowledge on a scale of 0–5, the average answer was 2.0. Only one respondent rated this as high as 4.

When asked about how fun they found PP to be, the ratings were not much higher. The average answer was 1.3, with no answers higher than 2.

Elevation of Privilege Fewer respondents had tried EoP than PP, which is somewhat surprising, as EoP is a much larger project, and created by Microsoft.

Of the three respondents that had tried EoP, one answered they did not feel it taught them any information security knowledge. The average rating was 1.3.

When asked how fun they thought EoP was, the average answer was 1.0, and none answered higher than 2.

Other Games There is an error in one of the responses received here. I was informed by one of the respondents in the second group that they accidentally ticked 0 when asked about the information security knowledge value and fun factor of other games, and were unable to remove the answers. These two results should thus be ignored.

The one respondent that had tried another information security knowledge game, OverTheWire games, gave its ability to teach information security a fairly low rating of 2, but rated it fairly high on how fun it was, with a rating of 4.

Aspects of an Educational Game When asked what aspects they consider most important for an educational game, the aspects listed as the most important, in order, were:

1. Fun factor.
2. Ease of use.
3. Educational content.
4. Performance.
5. Documentation.
6. Graphics.

It is worth noting here that the first group rated educational content and ease of use higher than fun factor, while the second group had a much higher rating for fun factor and performance.

TD as the Game-type When asked about whether or not they thought a TD is a good choice of game-type for this game, 61.3 % responded that they thought it was a good idea, while 9.7 % stated they did not think it was a good idea, and 32.3 % were undecided or unfamiliar with TDs.

Will to Try Educational Games for Information Security Knowledge Many of the respondents stated that they were willing to try an educational game for learning information security knowledge. In fact, not one of the respondents from the first group stated that they would not be willing to try such a game. In total, 83.4 % stated they would be willing to try out such a game, 12.9 % were undecided, and 3.2 % would not be willing to try it.

Wanted Features When asked what aspects the respondents most wanted to see in an educational game about information security knowledge, close relations to real-world information security was clearly the most popular one, with 51.6 % of the respondents choosing it. The other three choices that received a fair amount of votes were: links to the underlying information security knowledge with 16.1 % votes, easy-to-use UI with 12.9 %, and up-to-date data with 9.7 %. This was also reflected in the question where respondents were asked to tick *all* aspects they would like to see in such a game.

The aspects that decidedly received fewest votes were cool graphics and availability on several platforms, which were only chosen as a wanted aspect *at all* by 25.8 % of respondents each, and was not chosen as the most wanted aspect by any respondents.

Two respondents provided additional answers. One suggested doing the application as a web application with support for mobile devices for availability. A second one suggested that early access to factual information about the different enemies, attacks, and defences should be a priority. And a third one pointed out the importance of making a *fun* game, and that no one cares about boring games.

Unwanted Features When asked about what features would most deter them from trying a game such as this, the topmost chosen answer was a UI that is hard to use, being chosen by 41.9 % of the respondents. The second most unwanted feature was boring game entities with 22.6 %, closely followed by bugs with 16.1 %.

It is also worth noting that not one respondent chose boring graphics or high system requirements as their most unwanted feature.

5.3.5 Qualitative Questions

Thoughts on Teaching Information Security Knowledge Through Games Many respondents voiced their support of the concept and said positive things about using a game to teach information security knowledge. Here, a larger portion of the second group voiced their support without any other feedback, which could be because of their familiarity with me as the author.

There were also some skeptics that voiced their concern that they would spend more time learning the game than studying. There were also some concerns about the game becoming too much educational content, and not fun enough. Especially one answer described this in good detail:

"Most educational games I have tried force too much knowledge on the player. The most important thing is to make the game fun. If the game is fun and contains some educational material the player might learn something. If the game is not fun, people won't play it and won't learn anything. Gameplay needs to come first and educational aspects second."

Faults with Preliminary Game Design When asked about the faults they perceived with the preliminary design, many pointed out that the learning aspect and the learning curve can easily become too difficult and too steep for some. And that I need to consider everybody that could possibly want to play this game. One student also expressed doubts that a TD is well suited for learning.

Some comments were also made on specific aspects of the preliminary design sketches they were presented with in the survey, which may reflect on my inability to accurately explain what these sketches represented and what level of detail in them was to be considered part of the core design.

Some comments were also posed as good suggestions or as very constructive questions about aspects of the preliminary design, such as:

> *"I think the attacking entities should vary in seriousness based on how dangerous a security threat they are."*

And:

> *"How you go about visualizing the different security challenges, take stack smashing, how will you teach this concept accurately? What about side-channel attacks?"*

One response also mentioned that older people might not be as familiar with games as younger people, and that this is worth keeping in mind.

Positive Aspects of Preliminary Design Many comments pointed out how a TD could be a good format for making a simple and fun game that is easily learned and accessible by many. One such comment stated:

> *"A tower defense is a good choice for an educational game as it is a proven game mechanic. Players generally find it fun regardless of theme or topic."*

Other comments also stated that this design sounded like something they could find fun.

Suggested Improvements When asked about more concrete comments on how the respondents would improve the design many good ideas were suggested:

> *"Split attacks into groups based on seriousness. Maybe the player loses if just one if the really serious attacks get through? Seriousness could be based on whether a security threat would affect a single user or the whole system."*

> *"Add in an RPG element of a movable player (think of TRON)."*

> *"gamefy it even more. this is whats gone make people use it."*

And especially one comment had very detailed suggestions:

> *"From the pictures i see you've thought about stuff like this, but since you are asking about relating the game information to real world stuff, maybe add 'defences'/'towers' in the form of things are directly related to real world applications, like 'login pages', and you could maybe upgrade the login page with better password hashing, salt, form validation etc. Things that make sense. I havent played a lot of TD games but usually the fun part is being able to upgrade and buy cool things. Basically as the game goes on you become powerful and you kill a lot of enemies, and you can buy cool one time use items. Game design is important."*

These are suggestions that are good to keep in mind to avoid straying too far from real-world information security knowledge.

Others had suggestions that would entail changing the game-type or adding very large features that would alter the core of the game. Good suggestions, but they would mean increasing the scope of the project substantially.

Extra Focus Suggested areas of extra focus lists examples like:

- Making it understandable.
- Easy at the start, harder as the game progresses.
- Making it fun.
- Having intuitive and easy to use controls and game mechanics.
- Let entertainment supersede education.
- Focus on making it a unique TD, as there are many of these around.

Users' Ideal Game Here many different creative answers were given, e.g. making the game as a web application with WebGL graphics or letting the players write actual code to protect their assets.

Some comments also mentioned incorporating some form of an RPG component in the game, like a story or plot, to immerse the player further. One comment in particular described this in good detail:

> *"[...] either with or without a basic sub-plot such as being hired as a security consultant for a web/tech company. Though game mechanics are the most important part of a game, I feel a sort of story or plot would help making the learning aspect of this game interesting, as an arcade style game based around only the game mechanics would not supplement the learning experience in a meaningful way."*

Other Comments Not much relevant information or comments were given here.

5.4 User Survey Discussion

5.4.1 Background

The results of the background questions coincided quite well with what was the designated target group with regards to age, gender, and occupation; though perhaps more for the first than the second group.

Many had learned what they knew about information security knowledge through lectures, but quite a large part of the respondents had also used the internet to teach themselves about information security knowledge. This indicates that there is a will to learn and teach themselves in addition to listening to a professor talk about information security knowledge.

Somewhat surprisingly, only about two thirds of the respondents rated themselves as fairly familiar with programming. This is something that might influence how easily the players take to programming concepts, and should be considered thoroughly. To assume that all players are familiar with programming could put the player off the game if they are not. And as they say, "assumptions is the mother of all [problems]".

There was no surprise that most of the respondents were computer gamers to some extent. Many were very frequent gamers. This makes it easier to create a game for the target group, because many of them are already familiar with many of the concepts and mechanics of gaming.

Surprisingly enough, not many responded that they usually play TDs, which is contrasted by the fact that over three fourths of the respondents said they were familiar with TDs, and almost half of them liked TDs as a game-type. This indicates that maybe something special has to be done to make the game stand out among TDs, and not just become "one of the flock" that people forget quickly.

5.4.2 Questions

When it came to existing games for teaching information security knowledge, the respondents were asked about PP and EoP. Only about one in five had heard about these two games, which strongly indicates that teaching information security knowledge through games is not a widespread approach. A lot more of the respondents had tried PP than had tried EoP. Overall, people did not find these games fun, and they did not feel it provided any good value as a learning tool. Only one respondent rated PP as a fun game. This all indicates that there is a serious problem with existing games, both in providing entertainment and teaching information security knowledge. My hypothesis, which is one of the things I am exploring in this thesis, is that the latter is a product of the former: when a serious game is not entertaining, i.e. *fun*, it is not going to be engaging enough to teach.

One of the respondents had tried another game for learning information security knowledge, which indicates that for some of the target group, there is a will and motivation to try out alternative ways of learning information security knowledge.

According to the answers, the most important aspects for such a game is how fun it is, how easy it is to use, and what educational content it delivers, with fun factor and ease of use well ahead of educational content. This is something I will take due note of and focus on when developing the game, making these are not lost to other concerns. It is also shown here, as in other answers given, that the respondents care little for the graphics, it is a secondary concern.

The response to TD as the game-type was positive, and lends support to my decision of creating such a game. It serves to strengthen my belief that this is a good and engaging choice for the users in the target group.

Many of the respondents said they would be willing to try an educational game for learning information security knowledge. This indicates a lack of fun and engaging ways of learning information security knowledge as of now, and may indicate that this is keeping people from learning more about information security knowledge; that the problem is not that "information security knowledge is boring", but rather that there are no good tools for learning information security knowledge in an engaging way.

When asked about the most wanted features for such a game, it was clear that the content of the game needs to bear close relations to the state of information security knowledge in the real world. It can not diverge from this by prioritizing fantastic or imaginative threats, attacks, assets, and so on. Easy to use UI was again listed in high demand, indicating that people may have experiences with playing games that have a complex and less user-friendly UI that have frustrated them. The least wanted features were again graphics, as well as availability on several platforms. This indicates that people do not care much for having the game available on every platform they own, and may be indicative of the fact that the most significant use of this game would be in a classroom where most have a computer available.

The most *actively* unwanted feature, by a good margin, was hard to use UI. This, in addition to the earlier answers prioritizing fun and UI over educational content, indicates to me that the answers rating close relations to real-world information security knowledge as the top wanted feature may be skewed slightly by too few alternatives that the respondents could choose for that question.

Many of the qualitative answers served to emphasize the fact that the game has to be intuitive, easy to use, and easy to learn. They also provided many good suggestions and things to keep in mind when designing the game, and maybe even implement some of these into the game, if the time and scope allows.

One concept that was particularly interesting was the addition of an RPG element to the game: adding some plot or story. This was suggested to increase the immersion the users experience. And this is something I recognize from my own experience with games. Adding a plot is a heavily debated subject when it comes to games, see Jenkins (2004). Regarding this, it is all about balancing scripted and emergent gameplay (Sweetser and Wiles, 2005). Too much scripting, i.e. plot, can feel constricting for the players. A good rule of thumb is to guide the players with scripted gameplay, but always allow for emergent gameplay.

5.4.3 Consequences

This user survey has provided valuable insight into the demographic and profile of the target group. It has also provided valuable suggestions of focus areas, improvements, and features for designing a game of this type.

The responses has solidified my belief that a TD is the right choice of game-type for this kind of game.

As far as implementing features suggested by the respondents go, one particular suggestion of improvement caught my immediate attention: adding a story and plot to the game. I believe this will add entertainment-value to the game for many players, but it has to be done right, and not in a restrictive way. This is something I will strive toward, as far as the time and scope of this project allows. But adding a narrative to a game is no trivial task.

Also, there are a number of focus areas and concerns that the responses have emphasized and indicated is important to them. These focus areas will be at the top of my list of concerns to address when I am designing the game. I will here list the top three focus areas that I elicit as the most important for the target group, in order:

1. Fun: The game needs to first and foremost be fun and engaging. It has to provide entertainment-value to the users, meaning it should not only work as a serious game, but as a game.
2. Intuitive gameplay: The game needs to be quickly and easily understood, and the UI needs to be intuitive and easy to use for all users in the target group.
3. Real-world educational content: The educational content, i.e. the learning value, provided by the game needs to reflect the real-world state of information security knowledge. And it needs to be superseded by the entertainment value the game provides:

   "Pedagogy must, however, be subordinate to story—the entertainment component comes first. Once it's worked out, the pedagogy follows."

   — Zyda, 2005, p. 26.

5.5 Software Requirements Specification

5.5.1 Background and Structure

The SRS will be based on what the research goal for this project is, and what the idea is trying to achieve compared to existing projects in the same domain. The results from the users in the user survey will help shape and prioritize the requirements to what the end-users wish for in this project.

Furthermore, I will adapting a similar layout to that specified by The Institute of Electrical and Electronics Engineers (IEEE) for writing an SRS document (Institute of Electrical and Electronic Engineers, 1984). However, due to the small size of this project and the focus on developing a system and not operating it, I will trim the layout of the SRS to some select categories of requirements. Specifically, I will include these parts in my SRS:

1. External Interface Requirements (EIRs).
2. Functional Requirements (FRs).
3. Software system attributes, such as:
   (a) Reliability.
   (b) Availability.
   (c) Security.
   (d) Maintainability.
   (e) Portability.

The FRs are given a priority rating based on the value they provide to the system. A higher priority rating means that features supporting that requirement should be implemented in the system first. Priority ratings are given as low (L), medium (M), high (H), or critical (C).

5.5.2 External Interface Requirements

Table 5.1: *External Interface Requirements of the server.*

Id	Requirement
EIRS01	Requires a computer with minimum 1 GHz CPU and 1 GB RAM.
EIRS02	Requires a computer with a screen and keyboard.
EIRS03	Requires a computer running either Windows or any Linux Operating System (OS).
EIRS04	Requires Node.js v. 8.9.0+ and NPM v. 5.6.0+ to be installed.
EIRS05	Requires a command-line terminal for the administrator to interact with.
EIRS06	Requires an internet connection with minimum 10 MB bandwidth both up and down to serve data to game clients.

Table 5.2: *External Interface Requirements of the game client.*

Id	Requirement
EIRG01	Requires a computer with minimum 1.5 GHz CPU and 4 GB RAM.
EIRG02	Requires a computer with a screen, a keyboard, a computer mouse, a “Graphics card with DX10 (shader model 4.0) capabilities [...] CPU: with SSE2 instruction set support” (Unity Technologies, 2018b).
EIRG03	Shall display properly on any standard computer screen of at least 13.3" with any standardized aspect ratio.
EIRG04	Requires a computer running OS “Windows Vista SP1+, Mac OS X 10.9+, Ubuntu 12.04+, [...]” (Unity Technologies, 2018b).
EIRG05	Requires an internet connection to stay up to date with data from the server.
EIRG06	Does not require an internet connection to be played.

5.5.3 Functional Requirements

Table 5.3: *Functional Requirements of the server.*

Id	Artifact	Requirement	Priority
FRS01	The server	Shall fetch the latest version of data from its registered data sources, i.e. CAPEC, every 24 hours.	H

Continued: *Functional Requirements of the server.*

Id	Artifact	Requirement	Priority
FRS02	The server	Shall filter the data from data sources based on provided filters, e.g. only include attack patterns related to the OWASP Top 10 list.	M
FRS03	The server	Shall generate these game entities from the filtered data: • Attack patterns. • Course of actions. • Assets.	C
FRS04	The server	Shall generate relationships between the game entities.	C
FRS05	The server	Shall categorize course of actions to a smaller set of more general categories based on their descriptions.	H
FRS06	The server	Shall categorize assets to a smaller set of more general categories based on their descriptions.	H
FRS07	The server	Shall make a best possible attempt, but not guaranteed successful, at parsing these fields from CAPEC to put into the attack pattern entities: • Summary. • Attack Steps. • Attack Prerequisites. • Typical Severity. • Typical Likelihood of Exploit. • Examples-Instances. • Probing Techniques. • Indicators-Warnings of Attack. • Solutions and Mitigations. • Attack Motivation-Consequences. • Injection Vector. • Payload. • Activation Zone. • Confidentiality Integrity Availability (CIA) Impact.	H
FRS08	The server	Shall persist game entities as a JavaScript Object Notation (JSON) file (Crockford, 2006) on the server's file system, or as a JSON payload in memory.	M
FRS09	The server	Shall log errors and malfunctions to log files persisted on the server's file system: `combined.log` for all logged information, and `error.log` for errors only.	H
FRS10	The server	Shall test the most critical parts of the server's functionality: the generation of STIX entities, and the serving of these entities on an Application Programming Interface (API) endpoint.	M
FRS11	The server	Shall provide a *read me* file for easy setup and use of the server application.	M
FRS12	The server	Shall provide game entities as a JSON payload through an API-endpoint for game clients on a publicly accessible URL.	C

Table 5.4: *Functional Requirements of the game client.*

Id	Artifact	Requirement	Priority
Menus			
FRG01	The game	Shall provide a play configuration menu.	H
FRG02	The game	Shall provide a options menu with customizable options for the game.	M
FRG03	The game	Shall provide an about menu with information about the game, its background, and its author.	L
FRG04	The game	Shall provide a highscore menu with information about previously played matches.	L
FRG05	The game	Shall provide a main menu that links to the other menus: level select, options, highscore, and about.	C
FRG06	The game	Shall provide a pause menu while in the game.	H
FRG07	All menus	Except the main menu, shall provide a "back" button to return to the previous menu.	H
FRG08	The play configuration menu	Shall let the user choose relevant options for the match they are about to start: • Game difficulty. • Game speed.	H
FRG09	The options menu	The options menu shall provide options for enabling and disabling music and sound, and for adjusting their volume.	M
FRG10	The options menu	The options menu shall provide options for graphical quality.	L
FRG11	The main menu	Shall provide a button to exit the game.	H
FRG12	The pause menu	Shall provide buttons for: • Resuming the game. • Restarting the level. • Opening the options menu. • Going to the main menu. • Exiting the game.	H
FRG13	The pause menu	Shall stop all progress of the game view when open.	H
FRG14	The pause menu	Shall remember the state of the game after the user has opened the options menu.	M
All UI			
FRG15	The UI	Shall only contain elements with a clear purpose that do not clutter the UI.	H
FRG16	The UI	Shall not hide information, have inconsistencies, misdirect, distract, or breach with common practices for game UIs.	H
FRG17	The UI	Shall be usable by users in the target group without any training.	H

Continued: *Functional Requirements of the game client.*

Id	Artifact	Requirement	Priority
FRG18	Buttons	Shall be labeled with either: a universal unambiguous symbol for what the button represents, or a text label indicating its purpose and effect.	H
FRG19	Buttons	That are hovered over shall indicate that they are interactable, with some visual effect.	M
FRG20	Buttons	That are clicked shall indicate that they have been interacted with, with some visual effect.	M
FRG21	Buttons	That are disabled shall clearly indicate that they can not be interacted with.	M
FRG22	Buttons	That are disabled shall give feedback to the user about *why* they are disabled.	L
Game view			
FRG23	The game view	Shall show a game UI relative to the screen space at all times, i.e. attached to the edges of the screen and *not* the in-game world space.	C
FRG24	The game view	Shall contain a playing board that constitutes the boundary of play for all game entities and player interaction.	C
FRG25	The game view	Shall populate the board with assets, entry points, and paths when started.	C
FRG26	The game view	Shall allow the player to pause and unpause the game to inspect entities and the game board.	L
Gameplay			
FRG27	The player	Shall win the level when all the attacks have been destroyed/mitigated.	C
FRG28	The player	Shall lose the game when all his/her assets are destroyed.	C
Attacks			
FRG29	Attacks	Shall be generated from the attack pattern game entities produced by the server.	C
FRG30	Attacks	Shall be spawned in waves, i.e. as a set amount of attacks that spawn and must be destroyed before a new wave can spawn, at the entry points with 100 % integrity.	H
FRG31	Attacks	Shall be selected randomly at the start of each wave.	C
FRG32	Attacks	Shall be limited to 2 different types during each wave.	M
FRG33	Attacks	Shall make their way along the paths to the assets to destroy them.	C
FRG34	Attacks	Shall have these attributes: integrity, and speed. They may also have additional special attributes, e.g. immunity to area of effect attacks or immunity to damage over time attacks.	H
FRG35	Attacks	Shall be destroyed when their integrity reaches 0 %.	H

Continued: *Functional Requirements of the game client.*

Id	Artifact	Requirement	Priority
FRG36	Attacks	Shall be removed from the board when they are destroyed.	M
FRG37	Attacks	Shall be selectable by left clicking.	H
FRG38	Attacks	That are selected shall show extended information and possible actions in the UI.	M
FRG39	Attacks	That are hovered over shall show extended information about them	M
FRG40	Attacks	That are right clicked shall open a web page in the player's default browser with all their external references.	H
Mitigations			
FRG41	Course of actions	Shall be generated from the course of action game entities produced by the server.	C
FRG42	Course of actions	Shall indicate whether or not a mitigation can be implemented at the current position of the mouse on the board.	M
FRG43	Mitigations	Shall have these attributes: damage, attack speed, range, and mitigated attacks. They may also have additional special attributes, e.g. area of effect attacks or damage over time attacks.	H
FRG44	Mitigations	Shall attempt to destroy attacks that they mitigate when they come in range.	H
FRG45	Mitigations	Shall not attempt to destroy attacks they do not mitigate.	H
FRG46	Mitigations	Shall be sellable for 75 % of their implementation cost.	L
FRG47	Mitigations	Shall be removed from the board when they are sold.	L
FRG48	Mitigations	Shall be selectable by left clicking.	H
FRG49	Mitigations	That are selected shall show extended information and possible actions in the UI.	M
FRG50	Mitigations	That are hovered over shall show extended information about them	M
FRG51	Mitigations	That are right clicked shall open a web page in the player's default browser with all their external references.	H
Assets			
FRG52	Assets	Shall be generated from the assets game entities produced by the server.	C
FRG53	Assets	Shall have a certain integrity that starts at 100 % when the match starts.	H
FRG54	Assets	Shall show a bar over themselves indicating their integrity.	M
FRG55	Assets	Shall lose integrity when reached by an attack.	H
FRG56	Assets	Shall be destroyed when their integrity reaches 0 %.	H
FRG57	Assets	Shall be selectable by left clicking.	H
FRG58	Assets	That are selected shall show extended information and possible actions in the UI.	M

Continued: *Functional Requirements of the game client.*

Id	Artifact	Requirement	Priority
FRG59	Assets	That are hovered over shall show extended information about them	M
FRG60	Assets	That are right clicked shall open a web page in the player's default browser with all their external references.	H
Game UI			
FRG61	The game UI	Shall show buttons for pausing the game and opening up the pause menu.	C
FRG62	The game UI	Shall show the time elapsed during the match.	L
FRG63	The game UI	Shall show information about waves: • Current wave number. • Time elapsed in current wave. • Countdown to next wave between waves.	M
FRG64	The game UI	Shall show current integrity of assets.	H
FRG65	The game UI	Shall show the player's current worth.	H
FRG66	The game UI	Shall show current score.	M
FRG67	The game UI	Shall show information about what potential attacks can be expected in the next wave.	M
FRG68	The game UI	Shall contain buttons for placing taking course of actions that implement mitigations.	C
FRG69	Course of action buttons	Shall show the cost of implementing its mitigation.	H
FRG70	Course of action buttons	That have a cost higher than the player's current wealth shall be disabled.	H
FRG71	Course of action buttons	That are hovered over shall show extended information about the mitigation.	H
FRG72	Course of action buttons	That are right clicked shall open a web page in the player's default browser with all their external references.	H
FRG73	Potential attack indicators	That are hovered over shall show extended information about the attack.	H

Continued: *Functional Requirements of the game client.*

Id	Artifact	Requirement	Priority
FRG74	Potential attack indicators	That are right clicked shall open a web page in the player's default browser with all their external references.	M

5.5.4 Software System Attributes

The server should be maintainable by its author, i.e. me, or any new author at any later point. It should require the administrator to provide credentials in the form of a username and a password to interact with it. It should be stable when operating, and restart itself should it crash or shut down for any reason.

The game client should be reliable to not crash once it is started, and should not interact with any of the other systems or applications running on the user's computer. It should be runnable as a single executable, and not require any setup or install.

Chapter 6

Implementation

6.1 Data-driven Back-end Server

6.1.1 Architecture

The back-end server is built as an Express Node.js application (Hahn, 2016). This framework lets me write a fairly simple server application that is easy to deploy to almost any online cloud-service without much trouble.

For my server application, I am deploying it to an online service called Heroku[2]. This way, the server is available to anyone that has the server URL, which in this case is the game clients.

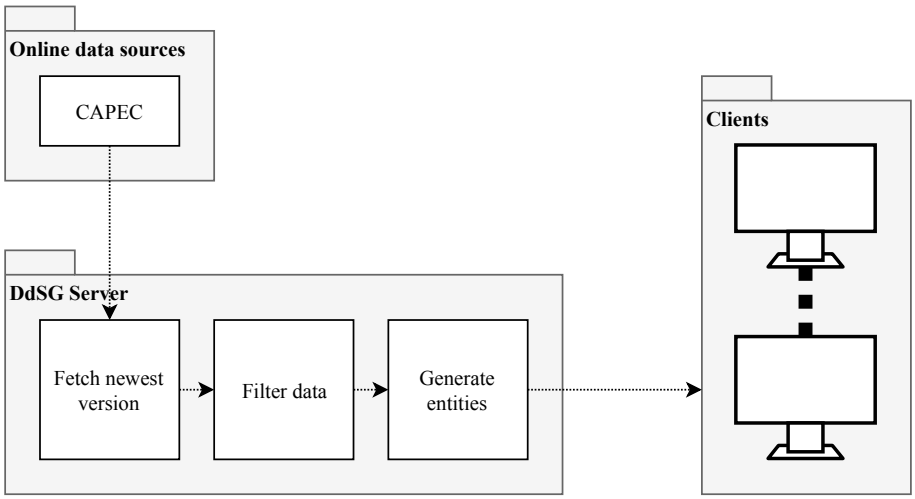


Figure 6.1: *Server architecture.*

The server's architecture is built as a sort of data pipe. The data starts out in the online sources, i.e. CAPEC, and it is the server's job to fetch the newest version of them, filter the data, and then generate entities from the data that are made available to any client connecting to the server. An overview of this architecture can be seen in Figure 6.1.

A more detailed description of how this data pipe structured can be found in Section 6.1.2.

[2] https://www.heroku.com/

The server architecture has been purposefully made more general and modular than strictly necessary. This is so that future developers of the server easily should be able to add new sources for data, add new filters, or generate different types of entities.

As of writing this, the server serves its entities as a JSON payload at https://ddsg-server.herokuapp.com/entities.

6.1.2 Detailed Design

The entire Unified Modeling Language (UML) diagram for the server application, only including the most central modules, can be seen in Figure 6.2.

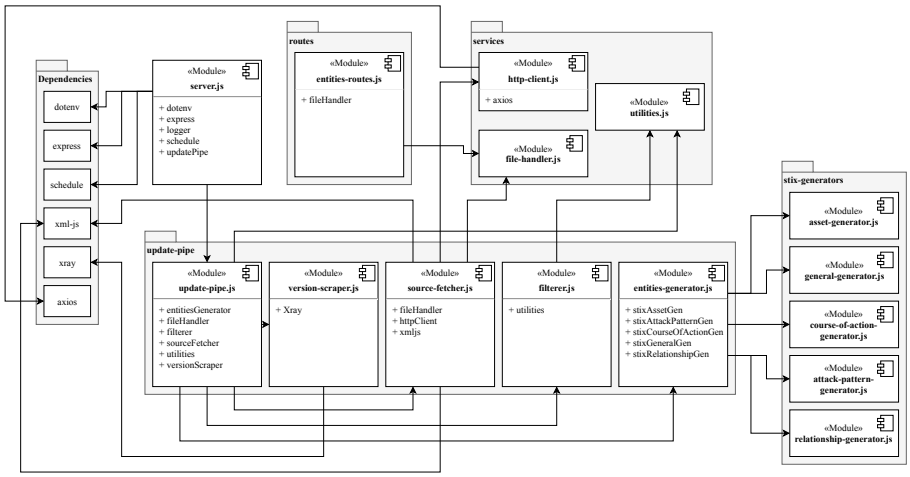


Figure 6.2: *Server detailed class diagram for important modules.*

As stated earlier, the server runs as a Node.js Express server. The program starts in the `server.js` file. This file is responsible for configuring and starting a server that listens for connections on a specified port. Furthermore, it starts the process of fetching and updating new data from the online sources. This is handled by an update pipe that includes modules for:

- Scraping what the newest version of the source is.
- Fetching the newest data from the source.
- Filter this data based on some conditions.
- Generate STIX-like entities from the filtered data.
- Store these entities. The entities can be stored in memory for use with simple hosting services, or as a file for more robust persistence. This is configurable by the administrator.

Figure 6.3 shows how the data travels through, and is manipulated by the system, all the way from the online sources to storing them for use by the clients. The update pipe module acts as controller for the other modules, responsible for using them and calling their methods when necessary.

This pipe is run on a regular basis, making sure that new entities are generated from information that is up to date automatically with no need for human interaction. This is what makes DdSG data-driven.

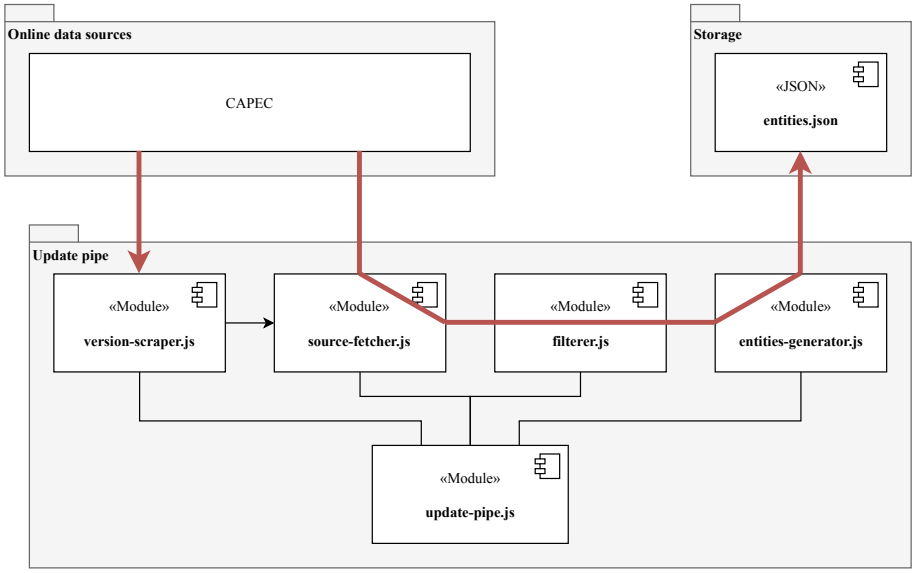


Figure 6.3: *Server data pipe. The data travels through the system along the red lines.*

Finally, the server exposes an API endpoint in another file, which can serve the entities generated to clients, such as this[3]:

```
1  app.get('/entities', (req, res) => {
2      logger.info(`Serving: ${req.method} ${req.headers.host || ''}${req.url}`)
       ↪ ;
3
4      res.json(entities);
5  });
```

This means that wherever the server application is hosted, the clients can connect to the server at the endpoint `/entities`, and receive all the entities generated by the server as a JSON payload.

To ensure that all of this works as intended, I have also been using unit tests and a couple of integration tests to test the logic of the server application. Some of the tests are simple unit tests to ensure that each module's exposed interface works as intended, such as this test for the `utilities.js`'s `timestamp()` method. The purpose of this method is to return an ISO 8601 formatted string for the present time. Here are both the function itself, and the tests covering its functionality[4]:

```
1  // From `utilities.js`
2  function timestamp() {
3      return `${new Date().toISOString()}`;
```

[3]Simplified for brevity.
[4]Simplified for brevity.

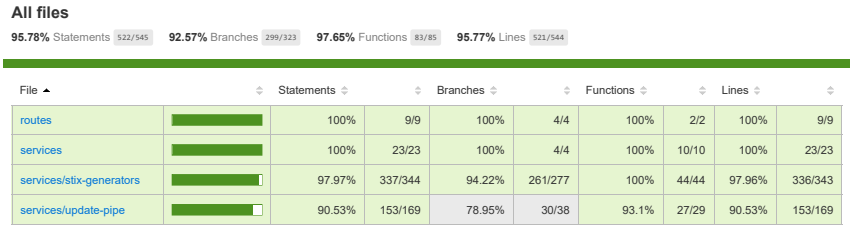
```
4  }
5
6  // From `utilities.spec.js` (i.e. the test file for `utilities.js`)
7  it('should return an ISO 8601 formatted date-string', function () {
8      const isoTimeRegex = /^\d{4}-[01]\d-[0-3]\dT[0-2]\d:[0-5]\d:[0-5]\d\.\d
       ↪ +([+-][0-2]\d:[0-5]\d|Z)$/;
9
10     timestamp.should.match(isoTimeRegex);
11 });
12
13 it('should return a time within 10 seconds of being called', function () {
14     const lowerRange = new Date(timeNow.getTime() - 1000*10);
15     const upperRange = new Date(timeNow.getTime() + 1000*10);
16
17     let interpretedTime = new Date(timestamp);
18
19     interpretedTime.should.be.within(lowerRange, upperRange);
20 });
```

These tests both tests that the resulting string is formatted correctly, and that the resulting time is within acceptable parameters.

But in this application, the most important tests that are implemented are the integration tests. They are testing the update pipe module's `fetchUpdatedDataFromSources()` method, which is responsible for enacting and completing the sequence for generating entities from the newest data in the online sources. They also makes sure to try the method with and without using the file system, so that the administrator can be sure that both storing entities in memory, and storing them in files, works as they should.

Also, as an additional integrity measure, I have been using a code coverage tool for JavaScript called Istanbul[5] to make sure that the tests I use actually cover the functionality of the server. Figure 6.4 shows how much of the server's code is covered by the unit and integration tests. It is important to note that the code coverage does not automatically mean that the lines covered are correct and without fail. This is the responsibility of the tests, and how they are written. What we gain from keeping track of code coverage, is to see that not any substantial or critical parts of the code is missed by the tests. That could indicate that there needs to be more tests, or better tests.

[5] https://github.com/gotwarlost/istanbul.



File	Statements		Branches		Functions		Lines	
routes	100%	9/9	100%	4/4	100%	2/2	100%	9/9
services	100%	23/23	100%	4/4	100%	10/10	100%	23/23
services/stix-generators	97.97%	337/344	94.22%	261/277	100%	44/44	97.96%	336/343
services/update-pipe	90.53%	153/169	78.95%	30/38	93.1%	27/29	90.53%	153/169

Figure 6.4: *Code coverage of tests on the server.*

As we can see in Figure 6.4, in this server application, Istanbul reports that 95.8 % of all the statements in the code is covered by tests. This is a good number, and indicates that the tests are testing a necessary amount of code.

6.2 Front-end Game Client

6.2.1 Game Design

The results of the user survey in Section 5.2 reaffirmed my belief in TD as the game-type. The game will be based on the template that is the TD game-type, but with modifications and additions. As explained earlier in Section 4.2, what really makes a TD successful is some fun and intriguing unique features. I will therefore strive to design and include some unique features to DdSG that, in addition to being one of the few educational games for teaching information security knowledge, makes it stand out among other TDs. This will be reflected in the SRS.

Put shortly, DdSG will be a single-player educational TD.

Having improved and concretized the idea of how this game should look, I have also created some sketches to illustrate the game in three different states.

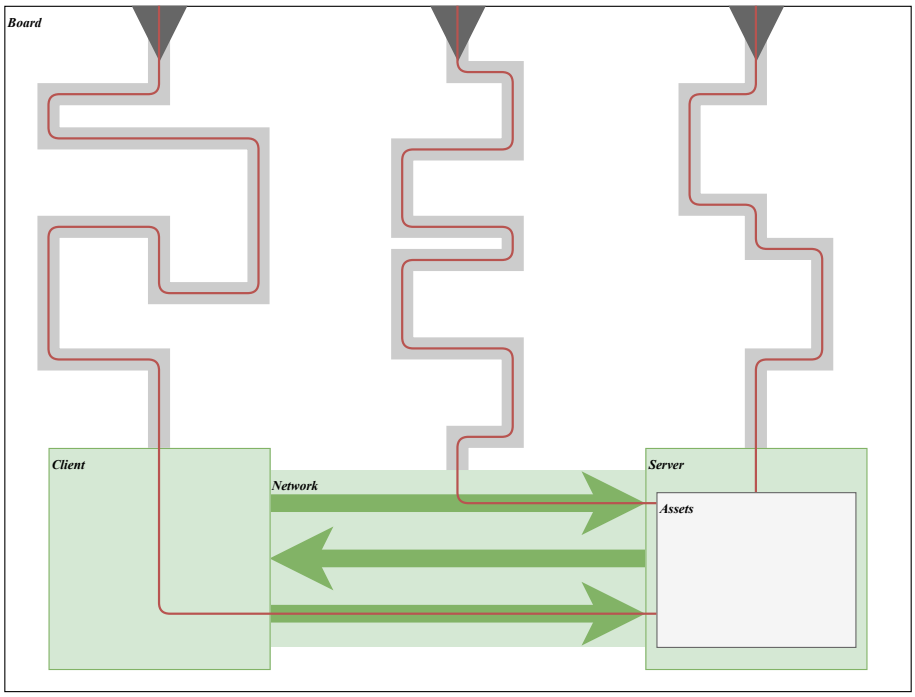


Figure 6.5: *Sketch of game design.*

The initial bare game state, as illustrated in Figure 6.5. This shows how the initial board will look like with all its static content. The red line shows the path the attacks will follow towards the assets.

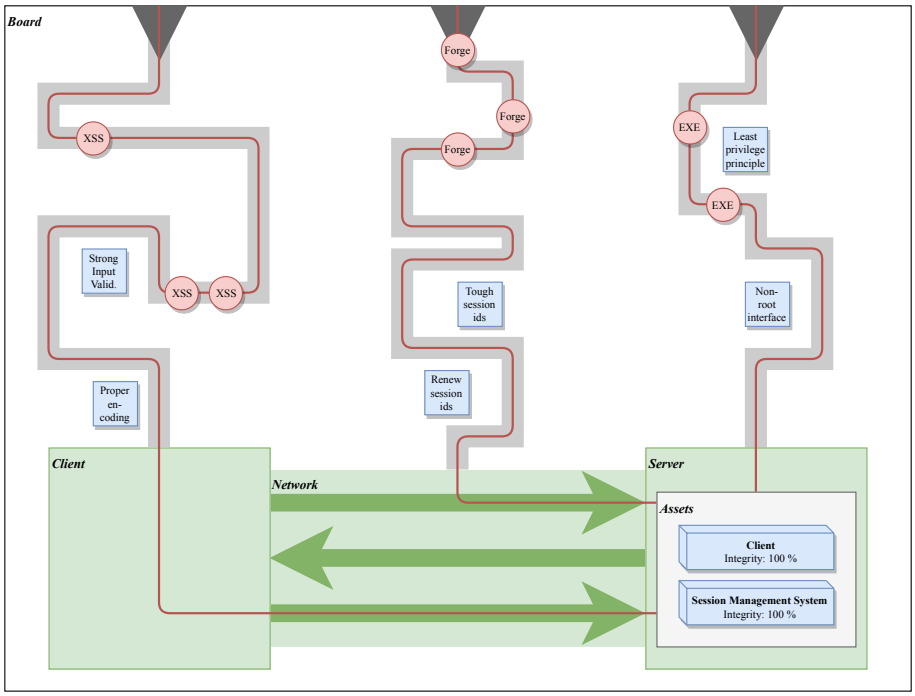


Figure 6.6: *Sketch of game design with entities.*

A game-in-progress state without interaction between the different entities, as illustrated in Figure 6.6. Here, the game has chosen some assets for the player to defend, spawned some

attacks that are making their way towards the assets, and the player has placed some mitigations along the paths to try to mitigate the incoming attacks.

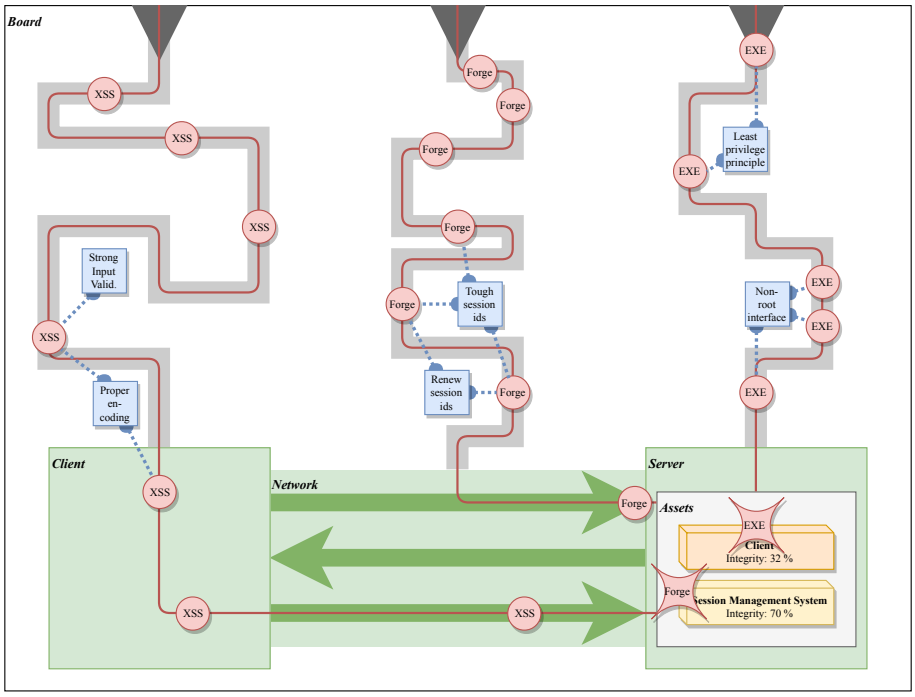


Figure 6.7: *Sketch of game design with game in progress.*

A full on in-game-progress with interaction between entities, as illustrated in Figure 6.7. Here, the mitigations have started engaging the incoming attacks, trying to destroy them before they reach the assets. Some attacks have been able to reach the assets and damage their integrity.

6.2.2 Architecture

As I am building this game in the Unity game engine (Unity Technologies, 2018a), most of the software architecture is hidden from me. Unity is a closed source game engine, and I therefore do not have insights into the internal workings of it, e.g. how its physics engine interacts with its scene renderer/graphics engine (Eberly, 2004). What I, as the game designer, have control over is the scripts used to control gameplay and game objects.

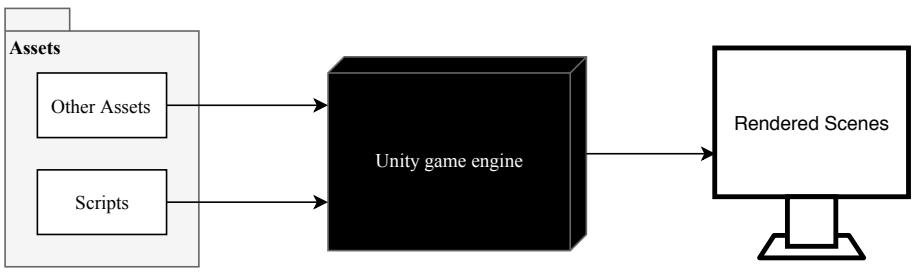


Figure 6.8: *Architecture of a Unity game.*

To create a game with Unity, I need to write scripts to dictate the behaviour of game objects in the scenes. These scripts are combined with other assets and resources such as images, 3D models, music files, sprites, icons, and others to produce the final rendered scenes that are presented to the user.

Nearly all scripts will interact with the Unity game engine as well, but since I can not know how that works internally, I will abstract it away and think of it as a black box (Bunge, 1963). We can think of it this way: the Unity game engine takes input from my scripts and other assets, and produces output such as rendered scenes. This is illustrated in Figure 6.8.

6.2.3 Detailed Design

As explained in Section 6.2.2, the design of the front-end game client is mostly determined by the Unity game engine. Thus, showing a detailed layout of the codebase here would provide little value. Instead, this section will focus on showing the different views the game presents to the user, as well as an explanation of the functionalities they provide.

Figure 6.9: *Loading screen.*

The game, titled "*STIX and Stones*" as can be seen in the loading screen in Figure 6.9, consists of six menus and one game view. The menus are there to support the user, create a better user experience, and guide the user into the game view with a customizable experience.

The six menus in the game are:

- Loading screen. Seen in Figure 6.9.
- Main menu.
- Options menu.
- Highscore menu.
- About menu.
- Play menu.

The loading screen provides little in the form of visual value, its purpose is to mask the important things that are happening in the background. While the user sees the loading screen, the game is fetching updated entities from the server to use in the game. There are also some checks during this phase. The game will always fetch the entities it gets from the server, and will not fetch new entities if the cache is newer than seven days. Additionally, it will use an old backup of the entities if it for some reason can not connect to the server.

Figure 6.10: *Main menu.*

After the loading screen is presented, the users are taken to the main menu. This menu acts as a routing hub where they can choose which other view they want to go to. They have the option of going straight to the play menu, adjust options, view highscores, check out the about menu, or exit the game completely.

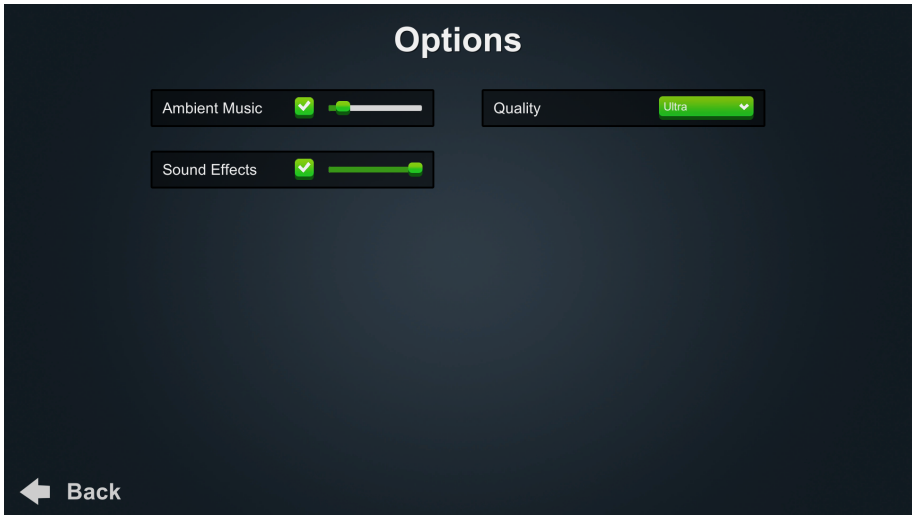


Figure 6.11: *Options menu.*

In the options menu, the users are given options to adjust the ambient music, sound effects, and the graphics quality of the game. This is to allow the users some feeling of control over the

input/output of the game, and to allow people with low-end computers to play the game just as well as people with high-end computers.

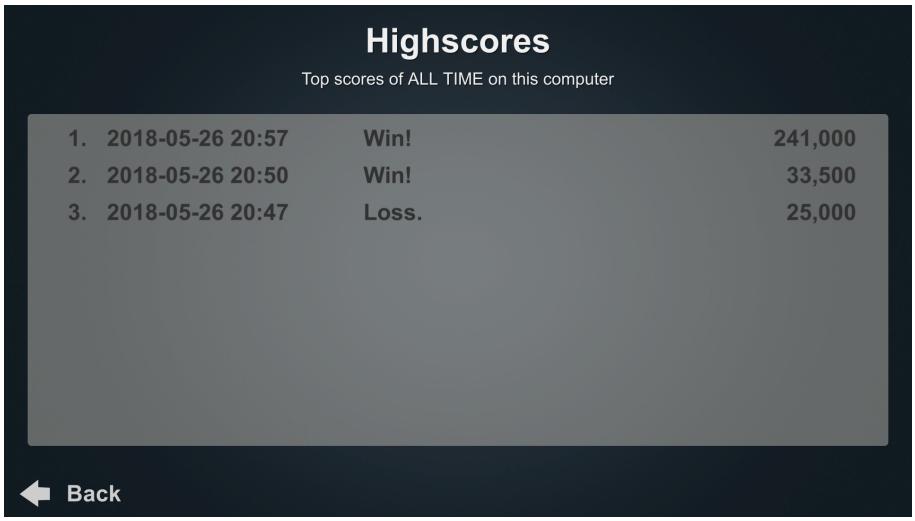


Figure 6.12: *Highscore menu.*

The highscore menu does exactly what it sounds like it does: it shows all the highscores that have been recorded on the user's computer, sorted by score. Since this is a single-player game, there is no name attached to each score, and no online scores.

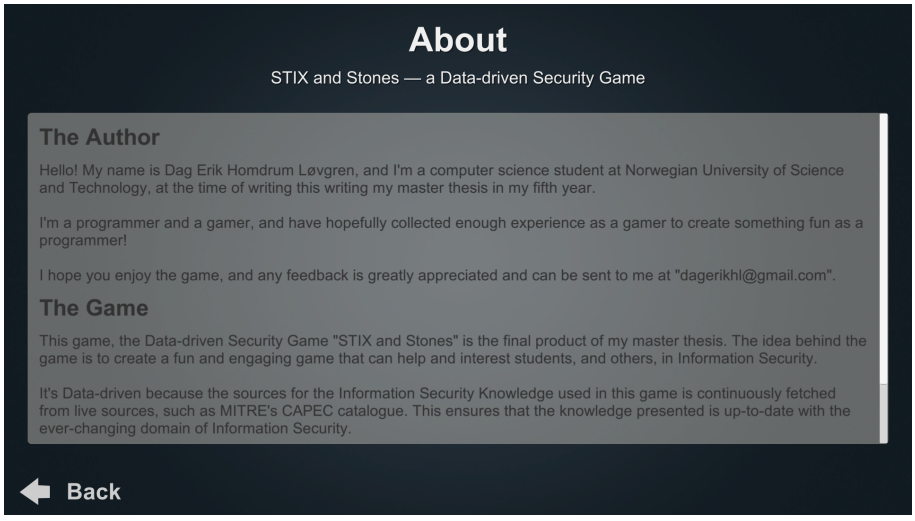


Figure 6.13: *About menu.*

The about menu shows the curious user some information about the game itself, and about the author of the game.

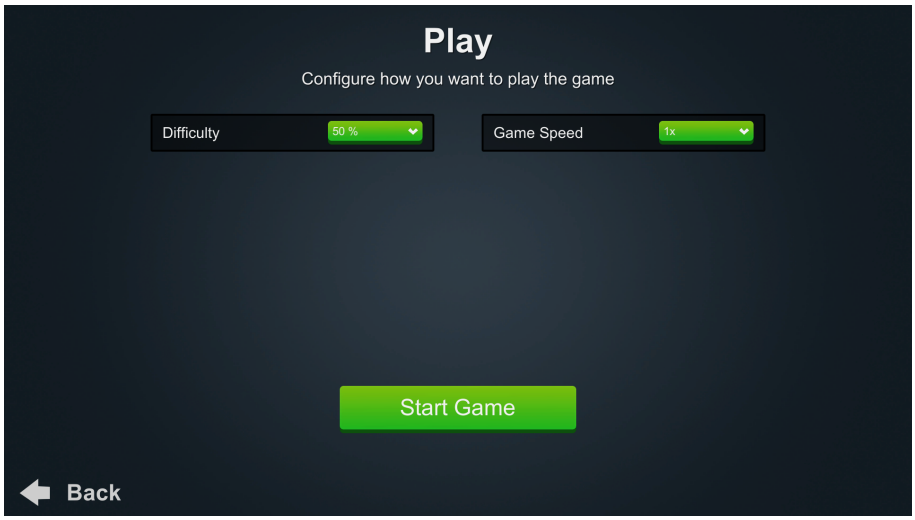


Figure 6.14: *Play menu.*

The play menu that the users are taken to after pressing “Play” in the main menu presents them with some options for configuring the gameplay for the match they are about to play. They are offered options for difficulty and game speed. Game speed affects how fast *everything* in the game happens, from how fast attacks spawn, how fast they move, to how fast their mitigations are able to destroy the incoming attacks. The difficulty affects how much integrity incoming attacks will have, and some other small things that change how difficulty a match will be.

The main part of the game consists of the game view itself. This is where all the gameplay happens. It also has three minor menus:

- The in-game pause menu.
- The win menu.
- The game over menu.

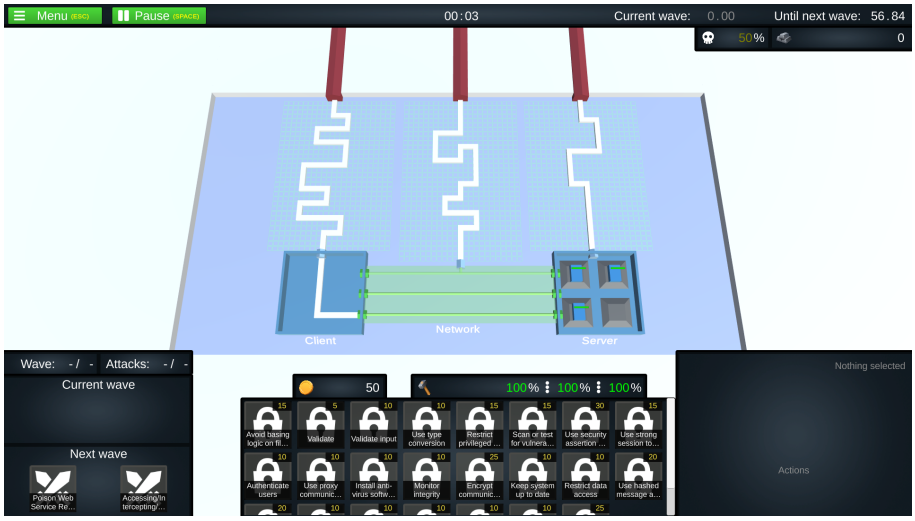


Figure 6.15: *Initial game view after game start.*

The initial game view presents the user with the game board, containing the injection vectors

where attacks may enter the user's system:

- Client.
- Network.
- Server.

It also contains the assets of the users. These are randomly chosen from the procured game entities at the start of each match and placed in the user's system, represented by the "Server" part, as can be seen in Figure 6.5. After the assets are chosen, attack patterns are selected. The game selects attack patterns that target an asset similar to the assets previously chosen. I.e. attack patterns that have a similar activation zone (Pauli and Engebretson, 2008, s. 3). These attack patterns are then added to a pool of potential attack patterns that may try to enter the system each wave.

A wave is the game's way of breaking up the incoming attacks into groups of random size that try to attack the system with a set amount of time between them. This allows the user to implement mitigations and prepare, to some extent, for the incoming attacks. This is a very common mechanic in tower defence games (Sutoyo et al., 2015)[6]. Each wave, a random amount of attacks are spawned. What attacks are spawned are chosen from a subset of two of the possible attack patterns that were selected at the start of the match. This is how a new wave is chosen by the script that controls the wave spawning:

```
1  public Wave GenerateNewWave() {
2      var possibleAttackPatterns = new List<AttackPattern>();
3      while (possibleAttackPatterns.Count
4             < Mathf.Min(State.I.GameEntities.SDOs.attack_patterns.Length,
       ↪ PossibleAttackPatternsPerWave)) {
5          var possibleAttackPattern = State.I.GameEntities.SDOs.attack_patterns
       ↪ .TakeRandom();
6          possibleAttackPatterns.Add(possibleAttackPattern);
7          // Ensure that no duplicate attack patterns are chosen for this wave
8          possibleAttackPatterns =
9              possibleAttackPatterns.Distinct().DistinctBy((aP) => aP.name.
       ↪ ToLower()).ToList();
10     }
11
12     return new Wave {
13         Count = Rnd.Gen.Next(MinAttacksPerWave, MaxAttacksPerWave + 1) +
       ↪ WaveIndex*ExtraPotentialAttacksPerWave,
14         AttackPatterns = possibleAttackPatterns.ToArray()
15     };
16 }
```

[6]Here, *wave* is used interchangeably with *level*.

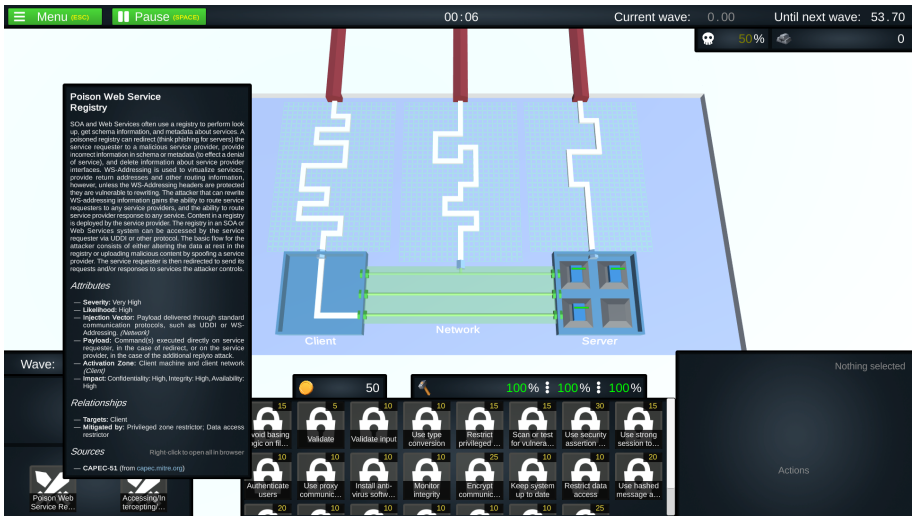


Figure 6.16: *The game shows information when the player hovers over an entity.*

After choosing potential attack patterns, all course of actions that mitigate these attack patterns are made available to the user, as can be seen in the bottom middle of the game Head-up Display (HUD). The user is able to click these course of actions to implement a corresponding mitigation on the board. The placement of these mitigations are restricted by the green grid we can see in the pictures, and are not allowed to be placed on the paths[7].

All the interactable parts of the game view, such as the course of actions, mitigations, assets, attacks, and incoming attack patterns show information to the user when they are hovered over. This can e.g. be seen in Figure 6.16, where the user is hovering their mouse over an incoming attack pattern. As all of these interactable entities or elements are pulled from a data source, the user can right-click them to open up their default web browser with the source material, i.e. the CAPEC attack pattern related to the entity or element.

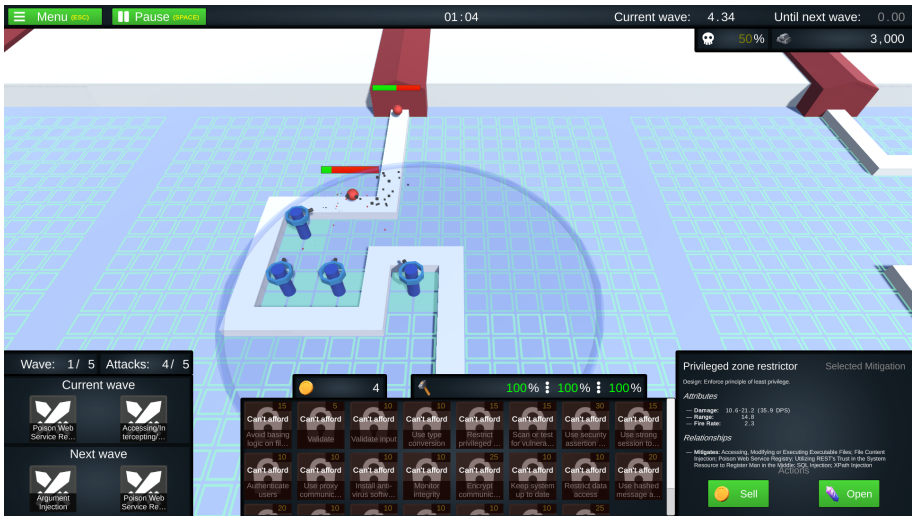


Figure 6.17: *Game view when game is in progress.*

[7]White paths leading from the top of the board and into the system at the bottom.

When the user has implemented a course of action, it will be represented as a structure[8] on the board. This structure will attack and try to destroy incoming attacks before they reach the end of their injection vector and can damage the asset they target.

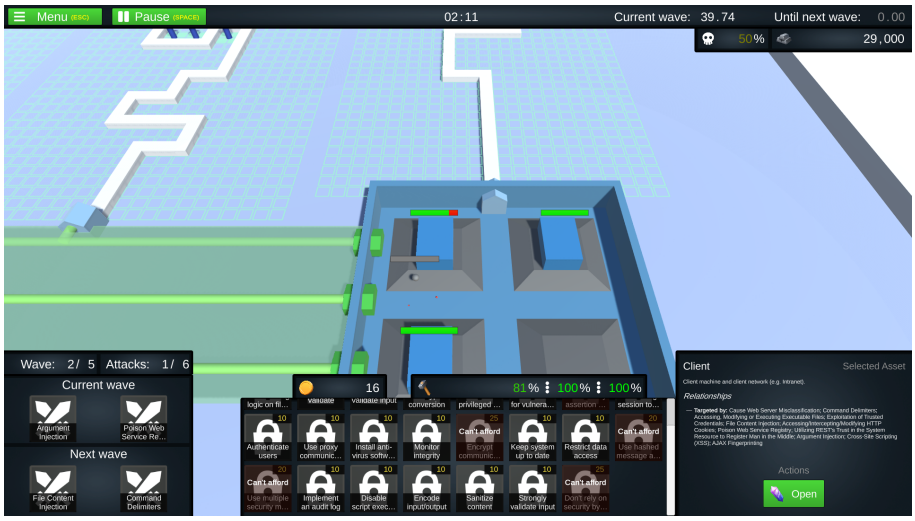


Figure 6.18: *Game view when assets are under attack.*

All incoming attacks will try to get to one of the assets in the server. Should they reach it, they will damage its integrity.

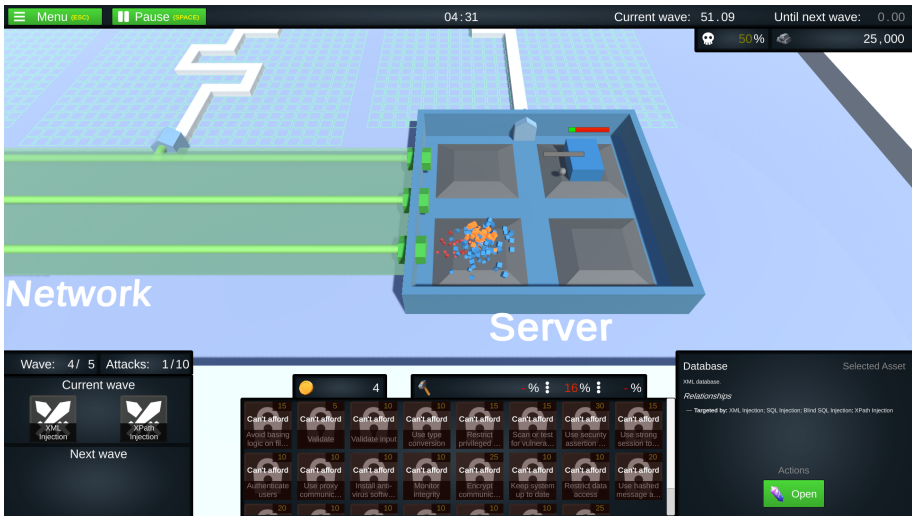


Figure 6.19: *Game view right before game over.*

If the asset's integrity reaches 0 %, it will be destroyed. This will prevent attacks targeting that asset from spawning anymore, but will penalize the user by redacting score points. Should *all assets* be destroyed, the user will lose, and the game will be over.

[8] I.e. a tower in the scope of TDs.

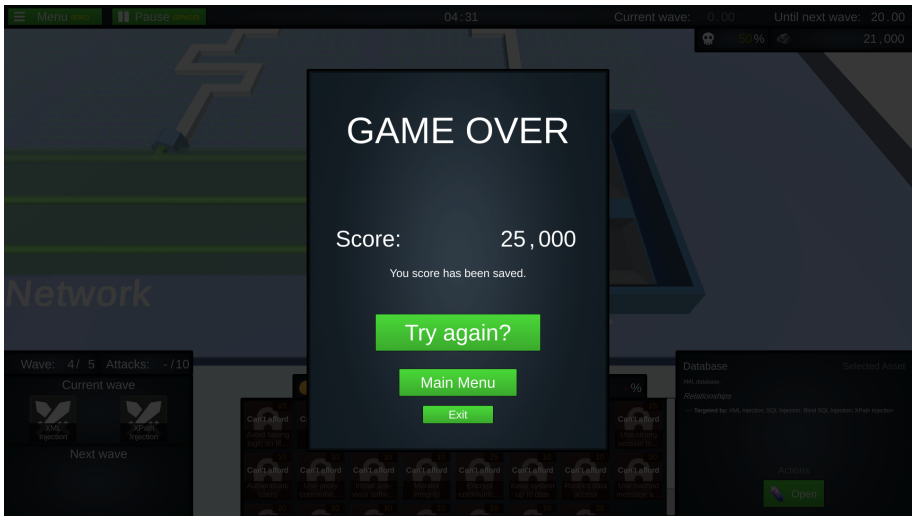


Figure 6.20: *Game over screen.*

To win the game, the user has to survive all the waves of attacks with at least one asset intact. As of now, there are five waves total, but this can be changed by the developer, and require more balancing and testing to explore and figure out what the most optimal amount is, to provide the most engaging experience to the users.

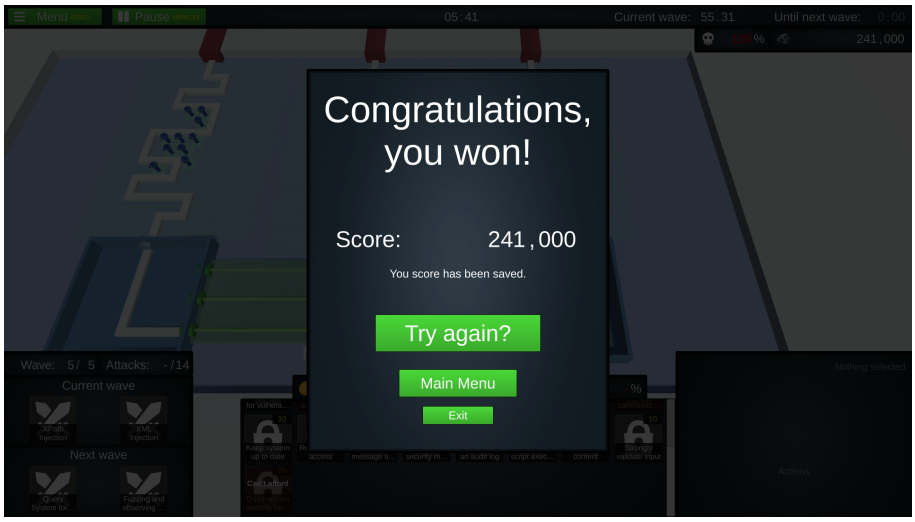


Figure 6.21: *Win screen.*

Whether the game ends in a loss or in a win, the score the user has achieved will be stored on the computer, and be listed in the highscore menu.

Chapter 7

Evaluation

7.1 Methodology

7.1.1 Usability Testing

To test the game DdSG, a standard setup for usability testing in software projects is used. This setup is based around tasks in task-based scenarios (C. Barnum, 2011, p. 19). The setup used here is slightly simplified. Because of the small size of this project, dividing up the testing in multiple scenarios is superfluous. Therefore, the entire testing session can be viewed as one scenario, with a set of tasks to be performed by the user.

The purpose of the tasks is to make the user test all the functionality the game has to offer, specifically the functionality described by the functional requirements of the SRS in Section 5.5.3.

While the user is performing the tasks, they provide feedback about the task they are performing through a form. The form in its entirety can be viewed in Appendix B.1. They are asked how difficult the task is, and in case they are not able to complete the task, they are asked to provide a reason for why they had trouble completing it.

To further evaluate the results from performing the tasks, the users are asked some more questions where they evaluate the usability further, and assess what value the game provides in relation to learning value and fun. This is explained further in the next section.

The full results of the user-testing, including both the feedback from the tasks, and the responses to the surveys, can be found in Appendix B.2.

7.1.2 Surveys

To get more qualitative data from the user-testing, one section on learning value and two schemas with 10 questions each follow the tasks. The section on learning value tries to map how much new knowledge the users were able to acquire during the tasks. The first schema asks questions about the perceived usability of the system, while the second schema, here called the assessment

scale, asks questions related to the assessment of DdSG as a learning game. Both of the schemas follows the well-know System Usability Scale (SUS) standard (Brooke, 1996).

For the first schema about usability, this makes 100 % semantic sense. For the second schema, the SUS is not being used for exactly what it was intended for. But the format of the SUS is well suited for testing any aspect of a system. That is, the format of a Likert scale (Albaum, 1997). It follows the same specification, but instead of asking questions about how usable the system seemed to the user, they are asked how fun and how much learning value they felt they gained from performing the tasks.

It is mainly the results from this survey that will serve as the basis for discussion and conclusion of research questions four and five.

7.2 User Group

This type of testing requires a smaller group than the user survey performed earlier, as it requires much more effort for the data collector, i.e. me, to perform user-testing with a group that is too large. Choosing a small group allows me to ask qualitative questions, and have the time and resources to properly analyze and discuss the answers given.

The group will be small group of 10 users that fit all the parameters of the target group.

Note: This small group is good for receiving qualitative answers about how the game can be improved, which will be of help in answering research question six, but the survey questions will suffer from not choosing a larger test group. The statistics gathered from such surveys would be more statistically significant and provide a larger value if the target group was larger. But to get an indication of how well this game performed, this group size is adequate. In the future, the game should be tested with a larger group to provide more statistical data.

Ideally, I would like to act as a facilitator and organizer *on location* with the users. However, this is not always easy to arrange and schedule for the users. Therefore, this testing was done remotely.

7.3 Data Collection

This user-testing is performed over the course of six days: from Wednesday, 23th of May, 2018, until Monday, 28th of May, 2018.

It was made available through a web link to a Google Form, which contained links to download the executable needed to run the game.

7.4 Background

To document that the users asked to test the game fell within the target group, they were asked some general background questions about their age, gender, and interests. The most interesting results from this section is how familiar the users were with computer games and information security before they started their tasks.

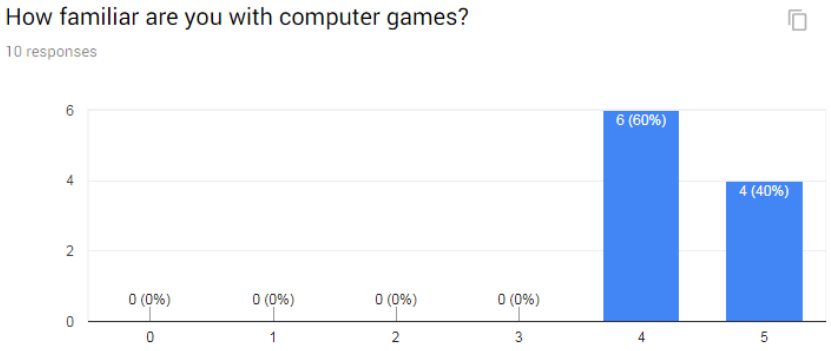


Figure 7.1: *How familiar users stated they were with computer games.*

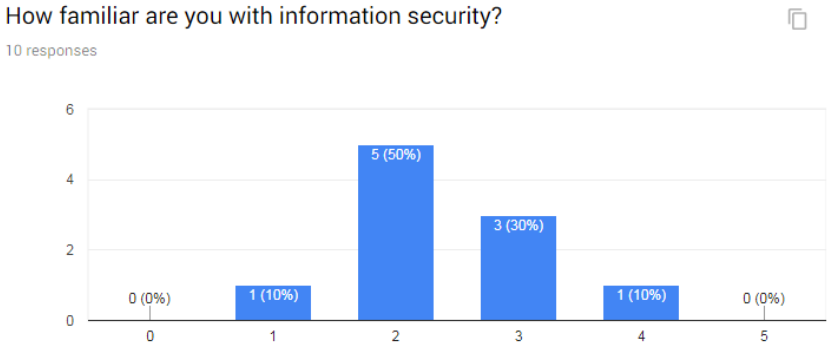


Figure 7.2: *How familiar users stated they were with information security.*

The full results can be seen in Appendix B.2.

7.5 Usability Testing

7.5.1 Tasks

All the tasks and their related feedback questions can be seen as a part of the form in Appendix B.1.

The users were asked to perform 13 tasks with the game:

1. Open the STIX And Stones application.
2. Adjust the volume for music in the game to your preferred setting.
3. Start a game with 25 % difficulty level and 1x game speed.
4. Pause the game.
5. Open the sources for the 2 incoming attack patterns in the next wave.
6. Move the camera to zoom in on the Server part of your system.
7. Select targeted asset of one of the 2 attack patterns and open its sources.
8. Implement mitigations to mitigate the 2 incoming attack patterns along the correct injection vectors.
9. Check that all of your implemented mitigations have sufficient range to target incoming attacks.
10. Sell one of your implemented mitigations.
11. Finish the game, protecting your assets as best you can.
12. Check out your new highscore.
13. Exit the game.

In addition, the users were given these two optional tasks, which they could perform if they they felt like playing more DdSG or try to improve their highscore.

14. Play the game again on whatever difficulty you want to see if you can beat your old highscore!
15. Play the game again on 100 % difficulty to see if you can beat the game at its hardest.

The tasks were formulated to provide the users with the freedom to experiment and perform the tasks as they wanted. This was done to avoid restricting the users too much or giving them a brain-dead list of tasks that required no reasoning. Doing that would have impeded the results of usability testing. Especially tasks 8, 11, and the optional tasks were loosely formulated, giving users the freedom to solve the tasks as they saw fit.

After each task, the users were asked to rank how easy the task was to perform, i.e. the *ease*[9]. This rating was given on a scale from 0–5, where 0 is impossible, and 5 is trivial.

[9] I.e. how *not difficult* the task was, for lack of a better word.

7.5.2 Results

Table 7.1: *Results from usability testing.*

#	Optional	Completed	Average ease score
1	No	100.0 %	4.6
2	No	100.0 %	5.0
3	No	100.0 %	5.0
4	No	100.0 %	4.9
5	No	100.0 %	3.9
6	No	100.0 %	4.7
7	No	90.0 %	3.5
8	No	100.0 %	3.3
9	No	100.0 %	4.2
10	No	100.0 %	4.5
11	No	80.0 %	2.9
12	No	90.0 %	4.1
13	No	100.0 %	5.0
14	Yes (5 respondents)	80.0 %	2.5
15	Yes (4 respondents)	75.0 %	2.0
Average			**4.0**

In Table 7.1, tasks that some users found impossible to complete, i.e. that had a completed percentage of less than 100.0 %, or that had an average ease score of less than 3.0, are marked. These tasks were not trivial for the users, and some had significant trouble completing them, thus requiring further exploration as to the reasons why.

Out of all the respondents, 60.0 % were able to complete all tasks when also counting the optional tasks, while 40.0 % had trouble with one or more tasks. Not counting the optional tasks, 70.0 % of the respondents were able to complete all non-optional tasks, while 30.0 % had trouble with one or more of them.

The easiest tasks to perform were tasks 2, 3, and 13, where 100.0 % of the respondents gave a score of 5.0, i.e. trivial, on the ease score. The two most difficult tasks were by far the optional ones: tasks 14 and 15. They had an average ease score of 2.5 and 2.0, respectively. The average ease score of all the tasks were 4.0, leaning toward trivial, keeping in mind that there are more simple tasks than there are complex ones.

Some users responded that there were some troubles with antivirus on their computer blocking the game from being run on task 1. This is not a reflection on the game or the task itself, merely on the way Unity creates its executables. But it is somewhat ironic that this should happen to a

game supposed to *help* people with information security.

Some minor bugs like keyboard shortcuts not working in pause mode and overlap of Graphical User Interface (GUI) elements were also reported.

One user reported having trouble finding the tooltip that states that any entity can be right-clicked to open its online sources, i.e. CAPEC source. Another user pointed out that right-clicking might not be the best shortcut for this.

Some users reported confusion about the fact that the *client* is a static part of the game board, and works as one of the injection vectors into the system, while *at the same time* it is the name of one of the assets that can be placed in the user's server at the start of each game.

According to some users, it was difficult to know where the attacks were coming from, i.e. how the *injection vector* was represented on the board. Finding out where the attacks were coming from and how to mitigate them required the user to move a lot between the different entities to read the tooltips. It was mentioned that this could enforce learning, but that it was detrimental to the *gaming experience*.

Several users reported that the challenge of the game was too severe; that it was too difficult. This was due to several reasons, but some that were reported were missing GUI elements such as health indicators on attacks and no help to users that had gotten a bad start. For the last one, it was also mentioned that this could be a valuable lesson to teach how severe security breaches work in the real-world, and their consequences.

7.6 Learning Value

7.6.1 Questions

The purpose of DdSG is to provide educational value about information security to the users. To assess whether or not DdSG had done this, they were asked to rate how much their knowledge of certain subjects they are exposed to through the game have improved after playing the game. They were asked to rate how much they felt their knowledge had improved on a scale from 0–5, where 0 is not at all, and 5 is enormously.

The users were asked to rate the their improved knowledge on the following subjects:

1. Assets.
2. Mitigations.
3. Attack patterns.
4. Information security in general.
5. Computer games.
6. TD games.

7.6.2 Results

Table 7.2: *Results from testing learning value.*

#	No improvement	Enormous improvement	>= 3.0 improvement	>= 2.0 improvement	Average improvement
1	10.0 %	0.0 %	40.0 %	80.0 %	2.2
2	0.0 %	0.0 %	80.0 %	90.0 %	3.0
3	0.0 %	0.0 %	60.0 %	90.0 %	2.9
4	0.0 %	0.0 %	40.0 %	90.0 %	2.6
5	10.0 %	10.0 %	30.0 %	40.0 %	1.9
6	10.0 %	10.0 %	60.0 %	80.0 %	2.6
Average					**2.5**

In Table 7.2, each row where a question has an average improvement score of less than 2.0 has been marked. These questions show that this subject is either: something the user knows a lot about from before, or a subject the game was unable to teach them much about.

The average perceived learning value in all categories were 2.5. This would equal somewhere between a C and D on a conventional grading scale from A – F, where A is best. For the questions concerning information security, i.e. questions 1–4, this average was 2.7; closer to a C on the grading scale. For the questions concerning computer games, i.e. questions 5 and 6, this average was 2.3; closer to a D on the grading scale.

7.7 System Usability Scale

7.7.1 Questions

To assess the usability of the system, the users were asked 10 questions in accordion with the SUS specification. In the SUS specification, every questions is alternately positive or negative towards the usability of the system. The goal of the scale is to produce a score that says something about the usability of the system. The score is given on a scale from 0–100, where 0 is an unusable system, and 100 is a system with a perfect user experience.

In each question, the users were asked to rate how strongly they disagreed or agreed with a statement on a scale from 0–5. Normally, SUS is scored on a scale from 1–5, but because of an oversight on my part, I have to adjust the scoring formulas according to this difference. To make this adjustment, I will use this formula on the average response to each question.

$$adjustedScore = score + 165$$

The mean score for SUSs is 68 (Bangor, Kortum, and Miller, 2009, p. 117). So ideally, DdSG should at least score above 68. Otherwise, it is shown that the usability of the game is sub-par, and needs work to provide a good user experience for the users.

To calculate the score of the system, a simple procedure is used:

1. Find the average of all responses for each question.
2. Adjust the average for the scale error.
3. Calculate SUS score for each question:
   (a) Odd-numbered questions: subtract 1 from the user's score.
   (b) Even-numbered questions: subtract the users score from 5.
4. Sum all SUS scores.
5. Multiply this by 2.5 to scale it up from a scale of 0–40 to one of 0–100.

The users were asked to what degree they agreed with the following statements:

1. I think that I would like to use this system frequently.
2. I found the system unnecessarily complex.
3. I thought the system was easy to use.
4. I think that I would need the support of a technical person to be able to use this system.
5. I found the various functions in this system were well integrated.
6. I thought there was too much inconsistency in this system.
7. I would imagine that most people would learn to use this system very quickly.
8. I found the system very cumbersome to use.
9. I felt very confident using the system.
10. I needed to learn a lot of things before I could get going with this system.

7.7.2 Results

Table 7.3: *Results from testing with the SUS.*

#	Average score	Adjusted average	SUS score
1	1.4	2.0	1.0
2	3.3	3.6	1.4
3	2.8	3.2	2.2
4	1.4	2.0	3.0
5	3.2	3.5	2.5
6	1.3	1.9	3.1
7	1.9	2.4	1.4

Continued: *Results from testing with the SUS.*

#	Average score	Adjusted average	SUS score
8	2.0	2.5	2.5
9	2.5	2.9	1.9
10	2.3	2.8	2.3
Sum			**21.3**
Scaled sum			**53.1**

The rows marked red in Table 7.3 have a SUS score below 2.0, which is very low compared to the max of 4.0. Though the score for question 1 is particularly, it is worth noting that this question is very broad, and could have a multitude of reasons.

The scaled score for the SUS was 53.1 for DdSG. This is significantly below what the mean for software systems are.

7.8 Assessment Scale

7.8.1 Questions

To calculate the score of this scale, the same procedure as described in Section 7.7.1 were used.

The users were asked to what degree they agreed with the following statements:

1. I thought this game was fun to play.
2. I found the purpose of the game confusing.
3. I feel more interested in information security after playing the game.
4. I thought there was too much information in the game.
5. I felt I had adequate control over how difficulty I wanted the game to be.
6. I found the game too challenging.
7. I want to play the game again.
8. I experienced many bugs while playing the game.
9. I want to see this game developed further.
10. I would rather learn about information security in a normal lecture than through this game.

7.8.2 Results

Table 7.4: *Results from testing with the assessment scale.*

#	Average score	Adjusted average	Assessment score
1	2.6	3.0	2.0
2	1.0	1.7	3.3
3	2.1	2.6	1.6
4	3.6	3.8	1.2
5	3.3	3.6	2.6
6	2.6	3.0	2.0
7	2.3	2.8	1.8
8	1.1	1.8	3.3
9	3.5	3.8	2.8
10	2.2	2.7	2.3
Sum			**22.8**
Scaled sum			**56.9**

The marked rows for the assessment scale indicates that the game did not strongly increase the users' interest in information security, that there was too much information in the game, and that not many would like to play the game again. The score on question 7 mirrors what was responded to question 1 in the SUS.

7.9 Optional Feedback

7.9.1 Questions

To produce more qualitative feedback, especially due to the small group size of this user-testing, the users were asked some optional open-ended questions. In these questions, the users were free to express any additional feedback, opinions, or anything they felt like sharing. This allows the users an arena for providing feedback about features, problems, or anything, that I, as the author, might not have considered.

The users were given three optional questions:

1. I found this especially good about the game.
2. This could be improved in the game.
3. I found this especially bad about the game.
4. General feedback.

7.9.2 Results

When asked to provide optional additional feedback about the game, the majority of the users, between 70.0 % – 80.0 %, chose to do so.

The users found these things especially *positive* about the game:

- Level design with client, network, and server injection vectors.
- Lots of content.
- Few bugs.
- Required learning information security.
- Pleasing aesthetics.
- The option to view the source of entities.
- The idea behind it.
- Functional UI.

The users found these things could be *improved* in the game:

- Balancing of currency, integrity, damage, etc.
- Missing a tutorial to teach new players how to play.
- Missing a gradual introduction to the concepts of the game, one at the time.
- Too much randomness, leading to the users feeling like *the luck of the draw* decides whether they win or lose.
- Different names for course of actions and mitigations.
- Missing more graphical elements like icons.
- Reduce need to hover over entities to view critical information about them.
- Pause disabled some functionalities like selecting entities.

The users found these things especially *negative* about the game:

- Too loud game music.
- Too much initial information.
- The connection between injection vectors and the client, network, and server paths on the board was too obscure and not indicated clearly enough.
- No difference in graphical style of the different attacks[10].
- Too slow; too much time is spent waiting.
- Right-clicking to open links.
- Too short gameplay; no upgrades to mitigations and too few waves.
- Lacking incentive to learn about information security apart from which mitigations mitigate which attacks.

The users had this additional general feedback about the game:

- A *start next wave* button would be nice to avoid waiting.

[10]The attacks actually differ in speed and size based on their type, but this might not be seen clearly enough.

- Good idea and basis, needs polishing and more features.
- “Evolving the game with a tutorial and increasing fixed levels with the goal of learning is not that far off if needed. A solid foundation to build from.”
- “Needs a bit more polishing, but I can see the value of this game if more time is spent on improving it. I think the game itself is a good idea.”
- The game could benefit from some sort of progression system, e.g. one that limits possible attacks and course of actions to a smaller subsets in the early waves, then grows as the player progresses.

Chapter 8

Discussion

8.1 Requirements of an Information Security Game

Because of the nature of requirements in development projects, and how the implementation is entirely dependent on the requirements, some of the discussion about this subject has already been covered in Chapter 5. Especially to meet the needs of the users, much of the discussion in relation to the user survey is covered in Section 5.4.

When identifying the requirements, one of the main things I kept in mind was that the back-end, i.e. the server part of the system, needed to fulfill its role as the data-driven part of the system. Its tasks were to get the data and and make it available to the users[11]. As discussed in Section 1.1, this is one of the things that will lend DdSG an advantage over the other existing solutions for teaching information security through games. To do this, the requirements had to reflect that the server needed to be totally automatic and not require any human interaction to function or to ensure that the newest and most up-to-date information was gathered. While identifying the requirements for the server I discovered that to do this, it was necessary to make other requirements more lax. E.g., the *quality* of the entities is not specified in any strict sense. To impose strict requirements on aspects such as quality, connectivity, conformity, and other attributes, would have led to big problems in the implementation part because the logic would have become too complex.

The game client, on the other hand, dealt directly with the users, and was identified with the assumption that the requirements of the server was going to be covered by the server in the implementation. During the identification of the requirements for the game client, involving the users was vital. To involve the users at this early stage is essential when developing some client software. According to Kaur and Sengupta (2011, p. 3), not involving the users at an early stage in the development process can be one of the main reasons for a software project to fail. Wallace, Keil, and Rai (2004, p. 117) also lists user risk, i.e. not involving the users, as a major risk of software projects. There have been many examples of software projects failing because the developers have implemented a system without involving the users and ended up with a product that does not meet the users' needs (G. Davis, 1982, pp. 4-30; Ives and Olson,

[11] Through the game client.

1984, pp. 586-603; Jiang and Klein, 1999, pp. 264-272; McComb and Smith, 1991, pp. 25-34; Robey and Farrow, 1982, pp. 73-85; Tait and Vessey, 1988, pp. 91-110).

The user survey was able to bring focus areas that were important for the users to light, and by doing that it was part of shaping how the SRS of the system was produced, and in turn how the the system was implemented.

As the product of identifying requirements, the section produces the SRS, a specific list of requirements that the implementation of the game has to adhere to. With the aid of the users, and a more abstract concept to build on, this SRS was produced with relative ease. The goal was *not* to insert too many features and functionalities into DdSG. At least not right away. This allowed me to keep the SRS fairly simple and it was possible to maintain an overview of all the functionalities it specifies.

8.2 Implementation

8.2.1 In General

When implementing the system, both the data-driven back-end server and the front-end game client, the SRS acted as the script and recipe for how to do it. The features that were implemented was extrapolated directly from the SRS. Some alterations or adaptations to account for unexpected issues with technology or software restrictions had to be done, but this was a minor issue. The features to implement were prioritized according to the SRS and implemented in the prioritized order, with some alterations when a feature was discovered to be dependent on a feature with a lower priority score. E.g. in some cases, the hover behaviour described in the FRs of the game client was necessary to implement the feature to select the entity.

During the implementation, I was most pleasantly surprised to find out that most, or actually *all*, of the FRs were possible to implement with the technology I had chosen for the task. I have done a few software development projects before, and this is seldom the case. This might be one of the benefits of having a software project completely developed by only one person[12], that there are no conflicting mental models and the developer, i.e. me, is likely to choose technologies they are familiar with or at least know the main constraints of. In this particular case I was very familiar with Node.js, but had no previous experience with Unity, so I expect some luck was involved as well.

[12]Not saying that there is not *a lot* of benefits to being multiple developers.

8.2.2 Data-driven Implementation

To ensure that all the data was up to date at all times and require no human labour to update it, it was necessary to automate the server. This was implemented by letting the server fetch the newest information on CAPEC every 24 hours. This may be a simple solution, but it ensures that the data is kept up to date.

The hardest part about implementing a completely data-driven implementation was parsing the unstructured data received from CAPEC. While MITRE is good at using consistent structures for their data compared to other sources which often have no schema to structure their data on at all, much of the data is still unstructured. They have a strict structure as to what fields an attack pattern can contain, but within these fields the raw data is very unstructured and inconcise. To be able to parse and categorize this data to comprehensible entities for use in the game client, the server had to implement some form of categorization algorithm. Though I had wished to implement something *smarter*. So the parsing and categorization the server does is not as elegant as I would like it to be, had I had the know-how and time to implement it in a better fashion. But, from what I can see myself from the parsed entities, how they present in the game, and the users' responses when playing the game[13], the server does a good enough job to demonstrate the idea. And it is 100 % autonomous.

One other thing I discovered while implementing the server as a Node.js application using only JavaScript ES5, was that writing an application, especially a back-end application, without *any form* of type-checking lead to some unforeseen bugs and a lot of debugging. Were I to do it again, I would not choose JavaScript ES5 as the language, at least not without using some framework or transpiler language like TypeScript (Bierman, Abadi, and Torgersen, 2014).

8.2.3 Fun and Intuitive Game Client

To implement the game client, it was necessary to set up a new design based on the original idea, modified by the feedback received from the users and the work done to identify requirements in Chapter 5. The design was based on the FRs for the game client, and in turn served as the basis for the implementation. As is often the case with software development projects: formulate idea $\longrightarrow$ identify requirements $\longrightarrow$ design solution $\longrightarrow$ implement solution $\longrightarrow$ [...]

The major obstacle when implementing a system that is supposed to be *fun* and *intuitive* is that these concepts are very abstract and not easy to translate into specific requirements or features. In other words, these are hard things to implement. To be able to do that I had to try to quantify *what makes games fun* and what makes them intuitive. To do that I reviewed literature on the subject and tried to avoid the mistakes that were made with existing games solving the same problem, as discussed in Chapter 3. E.g. that games like EoP (Shostack, 2014) can be experienced as too

[13] See the discussion on evaluation.

bland and boring because it can feel like they try to force too much educational content on the users. As we will see in Section 8.3, I was not entirely able to avoid this trap myself.

To make the game fun I leaned heavily on the inherent mechanics of TDs because it has proven to be such a popular game-type, and the article on "The Four Fun Keys" by Lazzaro from 2008. In short, Lazzaro talks about the four types of fun gamers have while gaming: hard fun, about challenge and mastery; easy fun, about imagination and exploration; serious fun, about changing the player's *internal state* or doing real work; and people fun, social interaction. In DdSG, I tried to pay special attention to these four types of fun.

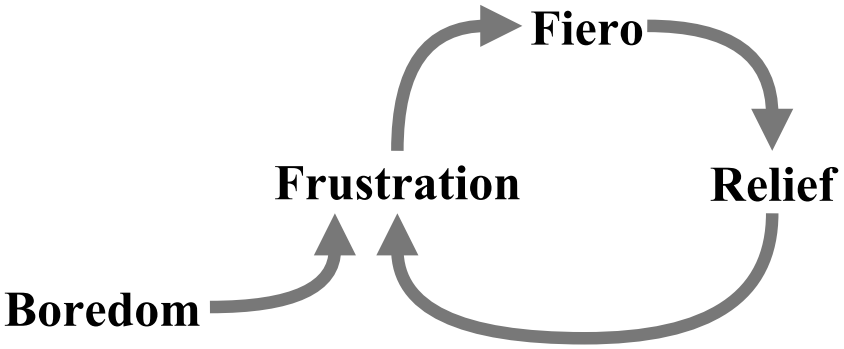


Figure 8.1: *The emotions players go through when experiencing hard fun in a game.*

The hard fun was implemented through different types of attacks, waves with varying level of difficulty, and increasing the difficulty of each wave as the player progresses. The players were also allowed to choose how much of the hard fun they wanted by allowing them to set their own difficulty level. Hard fun is about creating the feeling of *fiero* (Lazzaro, 2008, p. 325) in the user, i.e. the feeling you get when you manage to do something that you were having trouble with and was almost ready to give up on. It is about holding the users at the edge of this threshold between frustration and relief. This feeling could be attained by narrowly surviving a wave, just managing to destroy an attack before it moves out of range, or other situations. Luckily for me, a lot of the standard mechanics in TDs support the notion of hard fun.

The easy fun did unfortunately suffer somewhat. For easy fun to take place, the game has to allow for an emergent game style (Sweetser and Wiles, 2005). However, implementing for an emergent game style requires a lot of testing. To put it cynically: the more emergent gameplay is allowed, the more opportunities the users get to break the game. But I did try to implement features to allow the users some exploration and use of own thoughts. Allowing them to place their mitigations on a board lets them explore and figure out what is the best placement for mitigations. Allowing for several mitigations to mitigate one attack, without making it deterministically so that one mitigation is stronger than the other, the users are allowed to explore and try different combinations.

The serious fun is about changing the player's internal state, e.g. giving them a feeling of relaxation or of excitement. DdSG is not a particularly relaxing game, but is intended to create interest and excitement that motivates the players to continue playing and educate themselves

on the content of the game. In DdSG, this was implemented by several features, many of them relating directly to the hard fun of it as hard fun can create excitement. It is also about doing real work. Now, this can mean doing manual labour or training, like in the WII Fit games (Heick et al., 2012). But it can also mean intellectual work. And this is *exactly* what the intention of an educational game is. The intention behind DdSG is that every minute you spend playing the game, is a minute spent improving your knowledge about information security. The game is implemented in such a way that the players will become more proficient in *the game* as they become more proficient in *information security*.

As DdSG is a single-player game without any multiplayer features, the social fun Lazzaro talks about is not implemented. In the future it could be, however. This is discussed in more detail in Chapter 9.

Ultimately, there is no stone-written recipe to implement a fun and intuitive game yet. Although there are some that are working on a science for this for serious games (Zyda, 2005). The only way to see that the game has become fun and intuitive for the users is to test it. And this will be discussed in the next section.

8.3 Reflection on Evaluation Results

8.3.1 Usability Testing

The results produced when the users were carrying out the tasks given to them in the usability testing part seems to indicate that most tasks were easy to perform. The tasks that provided the highest level of challenge were the tasks that were most loosely formulated and afforded the users the most freedom. This is to be expected, as the users are given less guidance with such tasks.

However, looking at task 11 and the two optional tasks, a great drop in ease score can be observed. This could be due to the reason mentioned above, but because of the drastic drop for these three tasks and their similarity, it can seem like these tasks were not granular enough. Since the tasks asks the users to perform what is essentially an entire playthrough of a match in the game, the score the users assign their ease is not only based on the *usability* of the system, but the *challenge* of the game itself. This makes it hard to filter out what reflects on the usability, and what reflects on the game as a whole, including things like balancing. To do this, more small parts of the game's functionality should have had their own tasks. Then it would have been easier to elicit what part of the low scores for these tasks were connected to the usability.

When also considering the comments the users left as feedback to the tasks, it becomes apparent that a large part of what makes these tasks hard is that many of the users find the game very challenging.

The comments also indicate that the game is a bit confusing and overwhelming in general. Too much information and no introduction or tutorial makes the game hard to use the first time, and may put off new users quickly. While the game is intended as an educational game, the responses from the users seems to indicate that the game needs to tone down how much educational content it presents to the user, at least initially. The amount of information could be limited to a smaller subset at first, and then grow as the player progresses and becomes more familiar with the game itself, and the educational content.

The high average ease score indicates that, in general, the game did not have many problems with the separate parts of the UI being hard to use. Most of the negative feedback concerned the amount of information and lack of an introductory tutorial. There are, however, some concrete parts of the UI that the users had trouble with:

- It was not possible to move the camera using the keyboard arrows when the game was paused. This is a straight bug and should be fixed.
- Some entities have very long descriptions which may cause the UI container that is supposed to hold the text to overflow and spill the text out across other UI elements. Again, a straight bug in the system that should be fixed.
- Tooltips appearing when the user hovers over an entity may go off-screen if the entity is far enough to one edge of the screen. Also a bug in the system that needs to be handled.
- Right-clicking entities or icons to open their sources is not an intuitive way of doing this. This would need to be handled some other way. To figure out the best way to handle this, multiple users should be involved in the process of finding out what the most intuitive way of doing this is.
- Having an asset called *client*, as well as a static part of the board called the same is confusing. A possible solution to this would be letting the client asset be a special case assets that resides *in the static client*, see Figure 6.5, to remove this confusion.
- In the game, there is no clear connection between the injection vector of the attacks and the paths leading down to the client, network, and server in the game. Users had to figure this out themselves, often by trial and error, which was frustrating for them.
- One user reported problems with parts of the UI disappearing off-screen, such as the highscore, explaining why not all users were able to complete task 12. This might have been caused by a multiscreen setup on the user's part, and indicates that the game needs to be tested with more than one setup, i.e. more than my setup.

Some of these concerns are bugs that can be fairly easily fixed by a developer, and are therefore not very interesting. Others however, are more complex, and requires more complex solutions. Many of them would also benefit greatly from involving the users in the process, ensuring that the solution meets the needs of the users.

8.3.2 Learning Value

In the part asking feedback on what learning value the game provides, I want to discuss the first four and the last two questions separately. This is because the first four questions are about the learning value gained in information security, and the the last two are about the learning value gained in computer games. For this thesis, the first four are of more value and interest. I will briefly discuss the less interesting questions first.

According to the results of the survey on learning value, not many of the users increased their knowledge of computer games by any significant amount. Their increase in knowledge about TDs was somewhat higher. I think this is easily explained by the fact that 100.0 % of the users rated their familiarity with computer games as 4.0 or greater on a scale of 0–5, see question "How familiar are you with computer games" in Figure 7.1. This would also explain why more people increased their knowledge of TDs than of computer games in general, as not everyone familiar with computer games are familiar with TDs.

As can be seen in Table 7.2, on the first four questions, *over* 80.0 % reported an improvement score of their knowledge on the subject of 2.0 or more. This is a substantial part of the test group, and they reported a significant improvement in their knowledge of information security. They did not become experts, but they indicate that their knowledge within the field has expanded by a significant amount. These results are exciting, and gives me hope that the goal of this project is attainable, and can provide some real value to the academic world. At the very least, serve as an inspiration to future projects that set course on the same venture.

When looking at a value of 3.0 or greater, some differences between the knowledge gained about the different subjects becomes apparent. 80.0 % scored mitigations with 3.0 or greater, so the game is doing a good job of teaching about mitigations and course of actions[14].

The game is doing an OK job of teaching about attack patterns, with 60.0 % giving this a score of 3.0 or greater. There is room for improvement here, as the attack patterns can be seen as the most critical subject to learn about. After all, what good is knowing what you are to protect or how to defend it, if you do not know what will attack it?

[14]For the users, these two concepts becomes almost entirely interchangeable as the game is implemented.

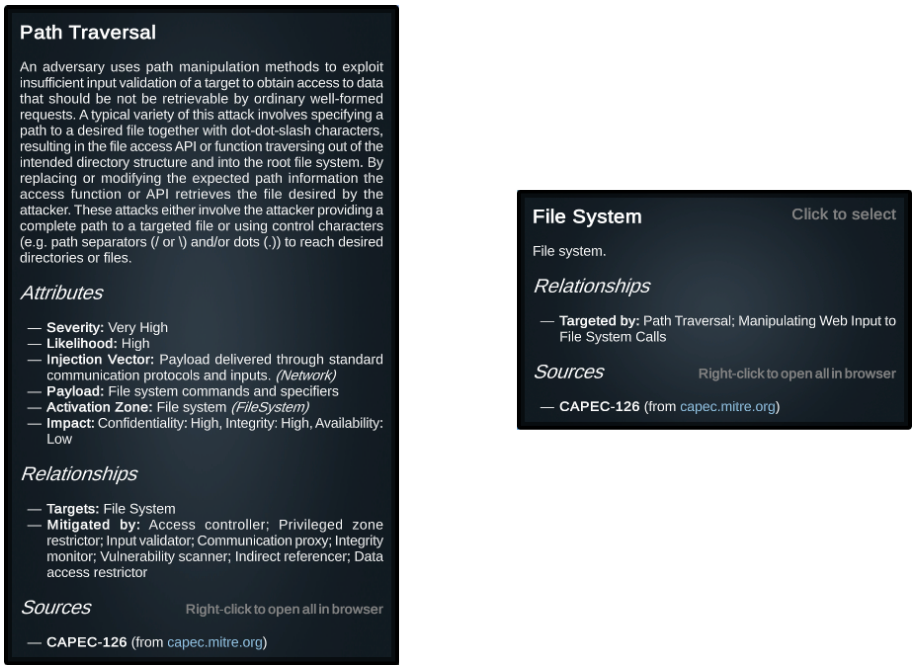


Figure 8.2: *To the left: the hover overlay shown when a player hovers their mouse over an attack pattern. To the right: the hover overlay shown when hovering over an asset.*

For assets, the game is doing a poorer job of teaching the users. Not more than 40.0 % of the users gave this a score of 3.0 or greater. I can imagine some possible reasons for this:

1. The assets, unlike the mitigations and attack patterns, do not contain much information. This makes them less interesting. See Figure 8.2.
2. From a purely gaming perspective, they provide little valuable information. Yes, attack patterns target them, but to target them they go through injection vectors. Thus, the user does not need to know what asset an attack pattern targets, the only information that is needed is where they are coming, and how to mitigate them.

The assets, as they are now, provide little knowledge or value to the game. They would need to be reworked, or included in the mechanics of the game in some other capacity than being the things the user has to protect and prevent from getting destroyed.

Though these results are exciting and indicates a positive outcome of the game's attempt at teaching the users information security, it is important to keep the small target group size in mind. The results here might look different had a larger group, and thus carrying more statistical weight, been tested.

8.3.3 System Usability Scale

The low scores marked in Table 7.3 indicates that the users thought the system was too complex, people would struggle to learn it easily, that it felt cumbersome to use, and that the users that tested the game would not be likely to use it frequently.

This reflects what was discovered during the tasks in the usability testing, that the system seemed complex and confusing to the user. The low score on question 7 reflects the feedback from several users that some form of tutorial for new players would be a good idea, to introduce them to the concepts of the game. The lack of such a tutorial is likely the reason for the low score on question 9 as well. With no introductory tutorial teaching the players how to play the game, they felt less confident when playing it.

Two of the questions also had a significantly higher score than the others: questions 4 and 6. Question 4 asks the users whether or not they think they would need the assistance of a technical person to use the system, and few users felt this would be needed. This is somewhat contradictory to the answers given on how confident the users felt while using the system. This could indicate that the system feels confusing and cumbersome *at first*, but that the users are able to understand and increase their confidence as they play, i.e. they are able to figure things out on their own and therefore does not feel they require the assistance of a technical person.

Question 6 indicates that the system succeeded in being consistent in its presentation to the user, and how the user interacts with it. I believe much of the reason for this is that this is a single-developer project. That makes it easier to be consistent across the entire implementation, as there is no room for differences in the mental model of the developers. In larger projects with multiple developers, this becomes more of an issue and requires more testing and cooperation among the developers, which can be viewed as overhead for the project. When developing the game, I made a conscious effort to keep the game and its UI consistent for the users, and the score given this questions indicates that this effort has paid off.

The total SUS score of the system is not as high as one would like it to be. As stated, the mean for software projects is 68.0, and the score of this game, 53.1, is significantly below that. According to Sauro (2011), and summarized in his web article from 2013 (Sauro, 2013), it is possible to grade the SUS score on a grading scale from A+ – F, where A is best. A score of 53.1 would give this system roughly an E on this scale. It is not a complete failure when it comes to usability, but it definitely needs work.

This low SUS score indicates clearly that there are some serious issues with usability, and that for any future version of this game, those issues needs to be addressed. Exactly what the most significant issues are is discussed in more depth in Section 8.3.1 discussing usability testing and in Section 8.3.5 discussing optional feedback provided by the users.

8.3.4 Assessment Scale

In Table 7.4, three questions receive significantly lower scores than the rest: questions 3, 4, and 7.

Question 3 is, to some extent, a summary question for how well DdSG did as a *learning game*. Much of the intention behind the game is to interest people that have no interest, or a mild

interest, in information security. The responses to this question seems to indicate that the game has not succeeded in this endeavour. A possible explanation for why this question scored so low could be that the users are already interested in information security, but if we look at the results from the background section in Figure 7.2, most users responded that they were not very familiar with information security. From this, we can extrapolate that most of the users do not have a particular interest in information security either.

The answers to question 4 also reflects some of the feedback received in other parts of the user-testing, that the game is confusing. This question reveals that at least one of the reasons for that, is that there is too much information. This causes the game to become confusing and overwhelming, especially for new players. The fact the score for this question is as low as it is might indicate that this is one of the primary reasons the users found the game confusing, if not *the* prime reason.

Similarly to question 1 in the SUS, the answers to question 7 reveal that not many of the users would like to play this game again. If viewed together with the responses on question 3, this becomes a serious concern for succeeding with DdSG as an educational game. The reasons for such a low score on these questions would need to be explored and improved upon should this project be continued in the future. Since the subject of the questions are so general and abstract, improving upon the more specific things that are wrong or lacking that the users have reported may be a good start to get the score of these questions up.

The assessment score here is not as low as the SUS score, but still lower than the mean for software systems evaluated by SUS. However, this is not a SUS scale, it merely uses the same system. There is no previous research indicating what is a good and what is a bad score for this scale. Therefore, it is assumed that the scale is linear and that 56.9, as it is above 50.0, is a *positive* assessment of the game. The scale is likely not 100.0 % linear, but without a case-study to assert the distribution of scores for this scale, it is a necessary assumption.

8.3.5 Optional Feedback

In many cases the optional feedback repeated what was given as feedback during the tasks, at least for some respondents. But often, they included more reasoning and suggestions for how to improve upon the shortcomings or errors.

What was positively surprising was how many of the respondents chose to provide me with their optional feedback, even though they were not required to. This shows that there is value in choosing such a small group as this, and having the resources to analyze the qualitative feedback.

Most of the specific errors, features, shortcomings, and positive aspects that were received during the optional feedback has been mentioned in the previous parts of this discussion, so this section aims to discuss the feedback in more general terms, eliciting the overall feel of what the users liked, disliked, and their suggestions for improvement.

Reading the optional feedback from the users, it seems many of them think the idea behind the game, and the concept of it, are sound. They liked being able to learn about information security through a computer game, having easy access to the sources, and some expressed that they thought the layout of the board how this was solved as a TD was an exciting way of doing it. They were also pleased by the consistency of the system, and the low count of bugs, especially game-breaking bugs.

Though the responses indicate that the users liked the concept and idea of DdSG, there were some obvious shortcomings. Most of the comments related to this can be boiled down to one of these categories:

- Missing features.
- Balancing issues.
- Information overload.
- Too simple mechanics.

Missing features is not a surprise in projects with such a limited time perspective like this project has had, it is necessary to keep the list of features short and focus on them to be able to deliver and test a product when time is limited. This has been a conscious choice for this project, and it pays off in a very rewarding way: the feedback from the users provides excellent ideas for new features to be implemented in the future. What these features could be will be further explored in Section 9.2. The balancing issues faced by the users are relatively simple to fix in such a small game, and of little importance. The comments relating to the mechanics being too simple, such as the lack of incentive to learn more about information security apart from how and what mitigations mitigate attacks, are important to address. Some of this could be improved by adding new features, as this would make the game more complex, and thus more interesting for the user. But some of these are deeper concerns that would need to be researched and explored further before they can be fixed. This would ideally also include the users in some capacity.

In short, it looks like most of the users agree that the idea is a good one, but that it needs to be polished and developed further to provide real value. This also mirrors what I think. I think the idea for using games in teaching information security is a good idea, but for it to work, more time needs to be put into developing a product for it. My hope is that this project could serve as a basis for this future work.

Chapter 9

Conclusion and Future Work

The purpose of this master's thesis has been to create a completely autonomous computer game for teaching information security.

With today's ever-changing security landscape, more and more information security experts are needed. Unfortunately, the field is not the one most often chosen by computer science students. To address this problem, attempts have been made at using gamification and educational games to engage and raise interest for the field. However, previous games have not been able to become the fun and engaging successes needed to sufficiently raise the popularity of information security.

This thesis has attempted to identify the requirements for creating a game that solves that problem. The users, as computer science students, have been involved in the process to discover what their needs for the game are. It was shown that the users agreed with the idea for DdSG that was proposed: to create it as a single-player Tower Defence game with interesting game entities like assets, mitigations, and attack patterns. Other feedback from the users emphasized things that were important to them: for the game to provide real-world educational content, and to do this in a fun and intuitive way.

In information security, new threats can appear quickly. To ensure that DdSG always stayed up to date, a server application was implemented to be completely autonomous and require no human interaction to function. Its responsibility was to provide the game with up-to-date entities by fetching and parsing the newest data from online sources like CAPEC at regular intervals.

Although the parsing done by the server was fairly simple, it was able to provide the game with up-to-date entities pulled directly from the newest source at CAPEC at all times. Thus, it eliminated the need for any manual labour whenever new security information was published.

To best implement a game to fulfill these requirements, it was designed to be *educational* without sacrificing the *fun factor* that draws people to games. DdSG was implemented in Unity for personal computers. The game design was based on existing and popular game mechanics, and it tried to conform to common and intuitive standards already established in the *gaming community*.

Testing the game with the users revealed that the concept was good, but that the game needed further development and polish to become a success. Results showed that users increased their knowledge of information security after playing the game, but often felt overwhelmed by the amount of information they had to process. It provided a fun experience to some, but it was

discovered that more features were needed for users to want to play the game regularly.

In short, DdSG shows potential for becoming a good learning tool for people wanting to learn about information security, but it has some flaws that needs to be fixed, and some more features that need to be developed to *really* capture its audience. To become a success, it needs some future developers to take the project one step further.

Here, I have listed some detailed suggestions for what can be done to improve the current version of the game. Based on my own thoughts, feedback from my supervisors, and ideas generated by the users, I also provide some suggestions on what features and ideas could be added to the game for it to grow into the success it has the potential to become.

9.1 Detailed Suggestions for Improving this Version of the Game

In the previous section on discussion of the results from the user-testing, Section 8.3, many bugs and shortcomings of DdSG has come to light. This section tries to comprise a table representing these problems, my own suggestions for how they can be fixed, and how much effort I estimate it would require to fix them.

These estimates are not to be taken as an absolute value, and is very rough. But since I have spent significant time developing this system, I feel my grasp of how much effort any future work on the project will require will be of value to any future developers. I will not estimate the required effort in hours, but on a scale from 0–5, where 0 is no effort and time at all and 5 is a lot of effort and time, measured in days of development time. The reader is encouraged to use this scale more to compare the relative effort of the different tasks in relation to each other, than as an absolute estimate of effort.

Table 9.1: *Detailed suggestions for improving this version of the game.*

Type	Problem	Suggested solution	Effort
Bug	Antivirus blocks game executable.	A common problem with Unity, and might not have any real solution[15], especially for a project this small. It might be better to include instructions to the users on how to whitelist the executable themselves.	1

[15] A possible solution is posted here: https://answers.unity.com/questions/1133613/how-to-deal-with-antivirus-blocking-your-build-for.html, but it may require too much effort.

Continued: *Detailed suggestions for improving this version of the game.*

Type	Problem	Suggested solution	Effort
Improvement	Tooltip documenting how to open sources is difficult to find.	I suggest two solutions: • Improve the visibility of the UI tooltip stating "Right-click to open all in browser". • Establish a convention in the game that a standard key code, e.g. right-click, opens up the sources for the thing the user is hovering over, and inform about this at the start of the game.	2
Improvement	Right-clicking is not an intuitive shortcut for opening sources.	Change the shortcut to something more common for computer games. Requires testing with users to figure out what they respond best to. A possible suggestion from the users is CTRL+SHIFT+Click.	2
Improvement	Client is both a static part of the board, *and* an asset.	Make the client asset a special case asset and place it in the client part of the board if it is picked. Then, let all attacks targeting it run towards it there, instead of going all the way into the server.	3
Improvement	How an injection vector was represented on the board was not explained.	Make a clearer connection between the *injection vector* attribute of attack patterns and the three injection vectors on the board: client, network, and server. A possible way to do this would be to show the current and next wave icons for the attack patterns *at the start of the injection vector they might use*.	4
Improvement	Finding out how to mitigate an attack required a lot of movement to read tooltips of entities.	There are several solutions to this, not mutually exclusive: • Show more verbose information about the incoming attack patterns in the HUD. • Make it possible to select more than one entity: e.g. to be able to select an attack and a mitigation at the same time. • Provide the user with some form of pop-up with information about the next incoming wave and its attack patterns.	4
Improvement	Difficult to gauge the integrity of incoming attacks.	Show numbers on the attacks' integrity bars indicating how much integrity they have compared to their max integrity, e.g. "43/100". This could also be show in the HUD when an attack has been selected.	2 (4 if in HUD)

Continued: *Detailed suggestions for improving this version of the game.*

Type	Problem	Suggested solution	Effort
Balancing	The game was too hard.	Values needs to be adjusted to balance how much help the user is getting and how much a challenge the incoming attacks pose. Values that could be adjusted are: • Value gained by mitigating attacks. • Cost of implementing mitigations. • Starting amount of currency. • Integrity of attacks. • Damage of mitigations. • Fire rate of mitigations. • Attack range of mitigations. • Attacks' damage to assets. • Speed of attacks. • Length of injection vector paths. • Number of turns in paths. • Time between waves. • Time between each attack spawn. Balancing a game is a delicate process, and is often *best done at the end of development* (Rouse, 2005, pp. 493-497)[16]. Balancing before this will undoubtedly lead to having to re-balance the game later. Any new feature will require the game to be re-balanced.	4
Balancing	No help afforded to players who got a bad start.	There are several solutions to this, not mutually exclusive: • Give the players some currency after each wave is completed. • Perhaps, on easier difficulties, give the players some resources if one of their assets are destroyed, to provide them with the means to "bounce back". • Make the first wave a "trial wave" that does not affect the rest of the game, but is merely a trial run with no consequences.	3
Improvement	Too much initial information.	Limit the amount of information at the start of the game to a subset of information. E.g. limit the number of possible assets, attack patterns, and mitigations to a subset for the early waves, then slowly grow the subset as the game progresses.	4
Bug	Can not move camera with arrow keys when paused.	The logic for moving the camera with the by pressing the arrow keys can be found in the `Move()` method of the camera manger in `CameraManager.cs`. This method will most likely have to be altered to fix the bug.	2

[16] Any future developers of this project are encouraged to read Rouse's book on the subject of game design (Rouse, 2005).

Continued: *Detailed suggestions for improving this version of the game.*

Type	Problem	Suggested solution	Effort
Bug	Long descriptions overflow the UI.	A simple solution comes to mind: Let the description of entities in the UI be wrapped in scrollable views so they will never overflow, just become scrollable.	3
Bug	Hover descriptions can go off screen.	Implement logic to adjust the placement of all hover overlays by the borders of the screen. All logic for the placement of the hover overlays reside in `HoverOverlay.cs`.	3
Bug	Multiple monitors may cause parts of the UI to disappear off-screen.	Requires testing on different systems to figure out if the error was a fluke, or a bug, and if it was a bug, what causes it.	3

9.2 Detailed Suggestions for the Next Generation of the Game

Figure 9.1: *Blizzard's popular game Hearthstone provides labels for the UI to help new players understand how to play the game.*

Source: https://camo.githubusercontent.com/f577fb2c3b2a5bcdef932346105a41463342c198/687474703a2f2f692e696d6775722e636f6d2f335a5839456c6d2e6a7067 [Accessed 2018-06-01].

In this section, I will describe some ideas for features that could make DdSG a better game, and help it become a better tool for learning. I will mention what the source of the idea is, to assert the background of the source as it may be relevant to the idea itself. I will also describe the value

I believe the idea will add to the project. And I will estimate how much effort implementing the idea would take. Again, these are rough estimates and should be considered in comparison with each others. Because I assume that new features takes overall more effort than making improvements and fixing bugs, I will here allow the scale for estimated effort to go from 0–10. This way, the amount of estimated effort will remain comparable between the fixes and the ideas.

Table 9.2: *Detailed suggestions for a future version of the game.*

Source	Idea	Adds value	Effort
Users	Grouping or colour-coding of attacks and mitigations that are related.	Makes the connection between mitigations and attack patterns more intuitive for the players.	4
Users	Highlight related attacks when hovering over a mitigation.	Same as above.	4
Users	Add an introduction tutorial for new players. This is definitively a critical and wanted feature, as it has been suggested and mentioned by several of the users in the user-testing. The solution is simple: to implement a tutorial level for all first-time players. To implement the solution is not so simple, but all the logic for a level of the game exists now, it only needs to be duplicated and adapted to fit the purpose of a tutorial. And help in form of descriptions, popups, and other helper objects. See Figure 9.1 for a good example of help provided to first-time players on how the UI works when they start the game the first time.	Introduces all the concepts of the game to the players in an easy way where there are no real consequences to not knowing how the game works beforehand. Would decrease frustration for first-time players. Would allow players to get to know the mechanics of the game so they can focus on the educational content in normal gameplay.	6

Continued: *Detailed suggestions for a future version of the game.*

Source	Idea	Adds value	Effort
Users	"Campaign mode". In this mode, the selected entities would be fewer and more scripted, so each playthrough of the campaign would be the same. For this to be fun, there needs to be several levels, i.e. matches, that get progressively harder that the players need to progress through.	Some players like to play games in campaign mode, some like to play it in "skirmish" mode[17]. A campaign mode would remove some of the feeling of "luck of the draw" some users refer to, making each level more deterministic. Would allow players to easily observe the increase in their skills over time, by replaying the campaign mode and see how much easier it has gotten.	8
Users	Add labels for assets that are visible at all times.	Avoids having to move the mouse to hover over the assets each time the player wants to view information about it.	2
Users	Open sources in an in-game pop-up-window instead of in the browser.	The players would not have to change applications to look at the sources. This would make for a smoother game experience.	?[18]
Users	Style the attacks according to their attack pattern so they are more easily distinguished from each other.	Allows the players to instantly recognize and group attacks into attack patterns by visual cues, instead of having to select or hover over the attacks to assert what type of attack it is.	4
Users	Upgradable mitigations. A common mechanic in TDs where each tower can be upgraded for a cost to a better or different version of itself. E.g. upgrading costs 10 currency, and improves the damage of the tower by 10.	In my own experience, this is one of the core things that makes TDs *fun*. It lets the users customize their system to a much greater extent, and increases the replayability of the game immensely. Also allows the players to feel they have a better chance of winning as the waves progressively gets harder and harder (Wong and Kang, 2015, p. 45).	9
Users	A button to start the next wave early.	Allows players that have understood the game and can place mitigations easily and fast to avoid waiting long periods of time. This alleviates boredom.	2

[17] Like the game is now, with entities randomly determined. Often does not have a win condition, but proceeds until the player loses, and the score is based on how far they got.

[18] I do not know if this is possible in Unity, and how much effort it would cost to do.

Continued: *Detailed suggestions for a future version of the game.*

Source	Idea	Adds value	Effort
Supervisors	An client interface for the server for administrators, e.g. lecturers or professors that want to use this as a learning tool. Here, they can update, alter, quality assure, and add new data in the form of entities.	Allows new entities with better quality to be created, or existing entities to be improved upon by an expert. As of now, the generation of entities is 100 % automatic, somewhat simple, and with no quality assurance. Letting experts edit this would allow them to improve upon the automatic parsing of entities from very unstructured sources. This would allow the game to present more readable and better entity descriptions to the users. With one caveat: it would rely on the administrators.	10
Supervisors	Allow the users themselves to choose what kind of assets they want to start with.	Allows users to "simulate" their own system to some extent so they may learn what attack patterns they need to be wary of, and what mitigations they need to implement.	4
Supervisors and me	Adding more sources to pull from, e.g. CWE or NVD.	Would allow the creation of more types of entities. E.g. adding CWE as a source would allow the introduction of weakness entities, or adding NVD as a source would allow introducing vulnerability entities. To support this change, new features would have to be implemented in the game client to make use of these new entities.	8
Me	Providing the player with some helpful feedback if the lose the game. E.g. "You were wiped out by Structured Query Language (SQL) injections reaching your back-end database. Perhaps next time you should try implementing more input validation?"	Educates the players on why they lost, increasing their knowledge of information security while simultaneously decreasing their risk of losing to the same tactic the next time they play.	3

Continued: *Detailed suggestions for a future version of the game.*

Source	Idea	Adds value	Effort
Me	Cooperative multiplayer mode where multiple players play together to protect the same assets. E.g. each player could be assigned an injection vector to protect.	Allows for fun sessions of multiplayer with classmates or friends. Very standard feature of TDs, and games in general, because *a lot* of the fun we have with games are based on social interaction. As Lazzaro states in her article from 2008: "People fun [i.e. social interaction] is [...] the source of more emotions than all the other types of fun combined [hard fun, easy fun, serious fun]." (Lazzaro, 2008, p. 336). Note: this would require a lot of work, as the entire premise of the game would have to be reworked.	10
Me	Competitive multiplayer mode where two players play one role each: one as the defender of the system, and one as the attacker. The defender has the same role as the player has in the game as of now. The attacker would determine what attacks to send to destroy the defender's assets. Could be combined with cooperative multiplayer mode to include more players.	Same as above. This type of multiplayer mode would also allow the players to observe an entirely new point of view, which could increase what learning value they get from the game.	10

Bibliography

Albaum, Gerald (1997). “The Likert scale revisited”. In: *Journal of the Market Research Society* 331.2, pp. 1–12. ISSN: 0025-3618.

Bangor, Aaron, Philip Kortum, and James Miller (2009). “Determining what individual SUS scores mean: Adding an adjective rating scale”. In: *Journal of usability studies* 4.3, pp. 114–123. ISSN: 1931-3357.

Barnum, Carol (2011). *Usability Testing Essentials*. ISBN: 9780123750921. DOI: 10.1016/C2009-0-20478-8.

Barnum, Sean (2014). “Standardizing cyber threat intelligence information with the Structured Threat Information eXpression (STIX™)”. In: *MITRE Corporation, July*, pp. 1–20. ISSN: 1011-6702.

Bierman, Gavin, Martín Abadi, and Mads Torgersen (2014). “Understanding TypeScript”. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783662442012.

Brooke, John (1996). “SUS - A quick and dirty usability scale”. In: *Usability evaluation in industry*. ISSN: 1097-0193. DOI: 10.1002/hbm.20701.

Bunge, Mario (1963). “A General Black Box Theory”. In: *Philosophy of Science* 30.4, pp. 346–358. ISSN: 0031-8248. DOI: 10.1086/287954.

Crockford, Douglas (2006). “RFC4627 - The application/json Media Type for JavaScript Object Notation (JSON)”. In: *JSON.org*.

Davis, G.B. (1982). “Strategies for information requirements determination”. In: *IBM Systems Journal*.

Davis, Keir, John W. Turner, and Nathan Yocom (2004). “Client-Server Architecture”. In: *The Definitive Guide to Linux Network Programming*, pp. 99–135. ISBN: 9781430207481. DOI: 10.1007/978-1-4302-0748-1.

Eberly, David H. (2004). *3D game engine architecture: engineering real-time applications with Wild Magic*. ISBN: 9781482267310.

Hahn, Evan M. (2016). *Express in Action: Writing, building, and testing Node. js applications*. November 2010. Manning Publications. ISBN: 9781933988771.

Heick, John D. et al. (2012). “Wii Fit and Balance”. In: *Topics in Geriatric Rehabilitation*. ISSN: 0882-7524. DOI: 10.1097/TGR.0b013e31825fca0e.

Hevner, Alan R. et al. (2004). “Design Science in Information Systems Research”. In: *MIS Quarterly* 28.1, p. 75. ISSN: 0276-7783. DOI: 10.2307/25148625.

Institute of Electrical and Electronic Engineers (1984). “IEEE Guide to Software Requirements Specifications”. In:

Ives, Blake and Margrethe H. Olson (1984). “User Involvement and MIS Success: A Review of Research”. In: *Management Science*. ISSN: 0025-1909. DOI: 10.1287/mnsc.30.5.586.

Jenkins, Henry (2004). "Game Design as Narrative Architecture". In: *Computer* 44, pp. 118–130.

Jiang, James J. and Gary Klein (1999). "Risks to different aspects of system success". In: *Information and Management*. ISSN: 0378-7206. DOI: 10.1016/S0378-7206(99)00024-5.

Kaur, Rupinder and Jyotsna Sengupta (2011). "Software Process Models and Analysis on Failure of Software Development Projects". In: *International Journal of Scientific & Engineering Research* 2.2, pp. 1–4. ISSN: 2229-5518.

Lazzaro, Nicole (2008). "The Four Fun Keys". In: *Game Usability: Advice from the Experts for Advancing the Player Experience*. Burlington: Taylor & Francis, pp. 317–343. ISBN: 9780123744470. DOI: 10.1016/b978-0-12-374447-0.00020-2.

Løvgren, Dag Erik Homdrum (2017). *Data-driven Security Board Game*. Tech. rep. Trondheim: Norwegian University of Science and Technology (NTNU). URL: https://github.com/dagerikhl/ddsg-docs/blob/master/Documents/DagErikHomdrumL%C3%B8vgren_2017_DdSG_Prestudy.pdf.

McComb, David and Jill Y. Smith (1991). "System project failure: The heuristics of risk". In: *Journal of Information Systems Management* 8.1, pp. 25–34. ISSN: 0739-9014. DOI: 10.1080/07399019108964967.

Pauli, Joshua J. and Patrick H. Engebretson (2008). "Towards a specification prototype for hierarchy-driven attack patterns". In: *Proceedings - International Conference on Information Technology: New Generations, ITNG 2008*, pp. 1168–1169. DOI: 10.1109/ITNG.2008.23.

Peffers, Ken et al. (2007). "A Design Science Research Methodology for Information Systems Research". In: *Journal of Management Information Systems* 24.3, pp. 45–77. ISSN: 0742-1222.

Robey, Daniel and Dana Farrow (1982). "User Involvement in Information System Development: A Conflict Model and Empirical Test". In: *Management Science*. ISSN: 0025-1909. DOI: 10.1287/mnsc.28.1.73.

Rouse, Richard (2005). *Game design: Theory and practice*. Vol. 8, p. 698. ISBN: 9780763798116.

Sauro, Jeff (2011). *A Practical Guide to the System Usability Scale: Background, Benchmarks & Best Practices*. Measuring Usability LLC. ISBN: 9781461062707.

— (2013). *10 Things to Know About the System Usability Scale (SUS)*. URL: https://measuringu.com/10-things-sus/ (visited on 2018-05-29).

Shostack, Adam (2014). "Elevation of Privilege: Drawing Developers into Threat Modeling". In: *USENIX Summit on Gaming, Games, and Gamification in Security Education*, pp. 1–15.

Sutoyo, Rhio et al. (2015). "Dynamic Difficulty Adjustment in Tower Defence". In: *Procedia Computer Science* 59, pp. 435–444. ISSN: 1877-0509. DOI: 10.1016/j.procs.2015.07.563.

Sweetser, Penelope and Janet Wiles (2005). "Scripting Versus Emergence: Issues for Game Developers and Players in Game Environment Design". In: *International Journal of Intelligent Games and Simulations* 4.1, pp. 1–9. ISSN: 1477-2043.

Tait, P and I Vessey (1988). *The Effect of User Involvement on System Success: A Contingency Approach*. DOI: 10.2307/248809.

Unity Technologies (2018a). *Unity Manual*. URL: https://docs.unity3d.com/Manual/UnityManual.html (visited on 2018-03-20).

— (2018b). *Unity - System Requirements*. URL: https://unity3d.com/unity/system-requirements (visited on 2018-04-06).

Van Acker, Steven and "morla" (2018). *OverTheWire: Wargames*. URL: http://overthewire.org/wargames/ (visited on 2018-03-08).

Wallace, Linda, Mark Keil, and Arun Rai (2004). "Understanding software project risk: A cluster analysis". In: *Information and Management* 42.1, pp. 115–125. ISSN: 0378-7206. DOI: 10.1016/j.im.2003.12.007.

Williams, Laurie, Andrew Meneely, and Grant Shipley (2010). "Protection poker: The new software security "game"". In: *IEEE Security and Privacy* 8.3, pp. 14–20. ISSN: 1540-7993. DOI: 10.1109/MSP.2010.58.

Wong, A. M. H. and D.-K. Kang (2015). "Game layout and artificial intelligence implementation in mobile 3D tower defence game". In: *International Journal of Security and Networks* 10.1, pp. 42–47. ISSN: 1747-8413. DOI: 10.1504/IJSN.2015.068410.

Xu, Dianxiang et al. (2012). "Automated security test generation with formal threat models". In: *IEEE Transactions on Dependable and Secure Computing* 9.4, pp. 526–540. ISSN: 1545-5971. DOI: 10.1109/TDSC.2012.24.

Zyda, M. (2005). "From visual simulation to virtual reality to games". In: *Computer* 38.9, pp. 25–32. ISSN: 0018-9162. DOI: 10.1109/MC.2005.297.

Appendices

Appendix A

User Survey

A.1 User Survey Questions

Data-driven Security Game: Survey

First of all, thank you very much for participating in this survey!

My name is Dag Erik Homdrum Løvgren, and I'm currently writing my Master Thesis in Computer Science at NTNU.

The goal of my thesis is to create an educational game for teaching people Information Security Knowledge (ISK). My main focus is to make the game fun, and base it directly on live sources of ISK. This would mean that it's kept up to date with any new information on Information Security (IS) on the web. I will be implementing this game as a 3D game in the Unity game engine.

The working title of the game is Data-driven Security Game (DdSG).

The purpose of this survey is to map what you, as part of the target group, think about the concept, what features you think should be part of the game, and how I can make this game a success. Any and all feedback is greatly appreciated!

This survey is completely anonymous.

The survey consists of 3 parts: Background information about you, the main questions about the game and concept, and lastly some optional questions for more detailed feedback.

The survey should not take longer than 5 minutes.

* Required

Target group

The target group is students currently enrolled in any IS / Software Security course with a little knowledge of IS, or anyone else that have an interest in ISK.

I assume all users will be at least somewhat familiar with computers, and that many will be familiar with computer games.

Vision

I envision DdSG as a Tower Defence (TD) type game (https://en.wikipedia.org/wiki/Tower_defense), similar to popular titles like Desktop Tower Defence, Plants vs. Zombies, Element TD, Bloons TD, Orcs Must Die!, Dungeon Defenders, GemCraft, and many more.

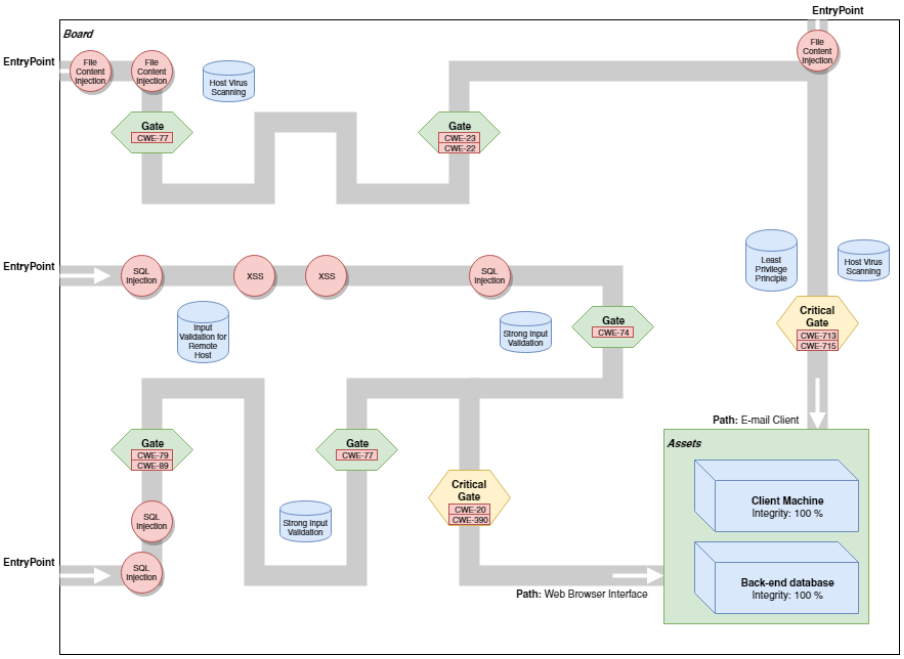
I believe the concepts in IS map well to the mechanics of TDs, such as attack patterns -> enemy types, mitigations -> defensive structures, assets -> base to defend, etc.

Concept sketches

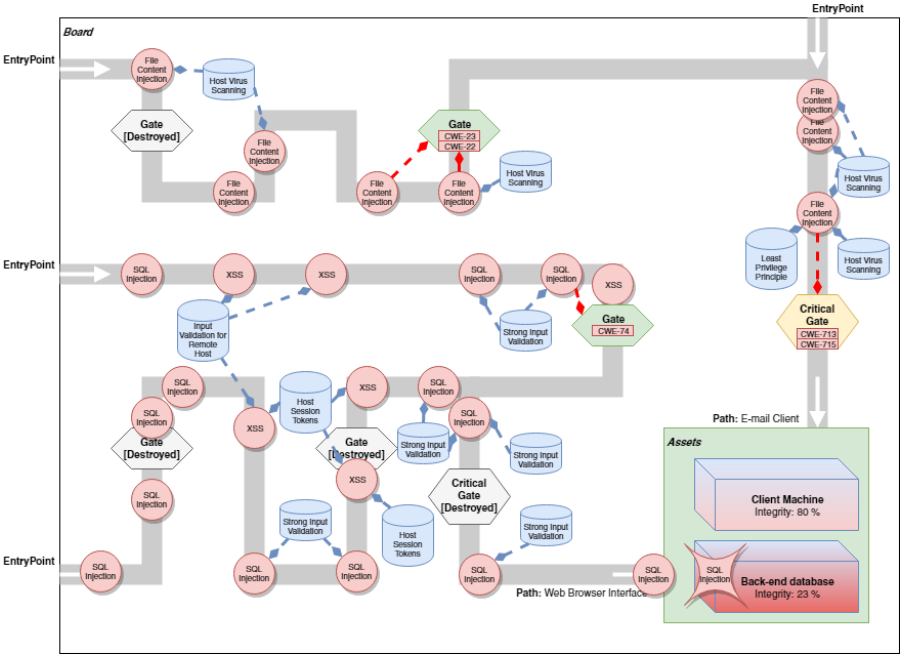
The images below shows a (very) preliminary concept sketch of DdSG. Here, the player tries to defend the Gates and Assets from the incoming Attacks, such as SQL Injection or Cross-Site Scripting (XSS).

The blue boxes represents the player's assets, the blue cylinders the player's defences / mitigations, the red circles the enemies / attacks, the green and yellow hexagons secure gates enemies must destroy to pass, the red boxes weaknesses of the gates that allow certain enemies to attack them.

Before attack has commenced



While attack is happening



Background info (1/3)

This section is to map some simple information about your background.

1. **Age** *

_____

2. **Gender** *
Mark only one oval.

- ◯ Female
- ◯ Male
- ◯ Prefer not to say

3. **Occupation** *
Mark only one oval.

- ◯ Student
- ◯ Working with Information Security
- ◯ Working with Computer Games
- ◯ Working with something else
- ◯ None at the moment
- ◯ Prefer not to say
- ◯ Other: _____

4. **How familiar are you with Information Security Knowledge?** *
Mark only one oval.

	0	1	2	3	4	5	
Not familiar	◯	◯	◯	◯	◯	◯	Expert on the subject

5. **How have you learned what you currently know about IS?**
Check all that apply.

- ☐ Lectures
- ☐ Self-study
- ☐ Friends
- ☐ Books
- ☐ The internet
- ☐ Articles
- ☐ Educational games
- ☐ Not relevant / I don't know any IS
- ☐ Other: _____

6. **How familiar are you with programming information systems?** *
Mark only one oval.

	0	1	2	3	4	5	
Not familiar	◯	◯	◯	◯	◯	◯	Expert on the subject

7. **How often do you use Information Security Knowledge when programming an information system?** *
Mark only one oval.

- ◯ Daily
- ◯ Weekly
- ◯ Monthly
- ◯ A couple of times a year
- ◯ Maybe once a year
- ◯ Never
- ◯ Not relevant / I don't create information systems
- ◯ Other: _____

8. **How often do you play Computer Games?** *
Mark only one oval.

- ◯ More than once per day
- ◯ Once per day
- ◯ Every week
- ◯ A couple of times a month
- ◯ A couple of times a year
- ◯ Never

9. **What types of games do you usually play?** *
Check all that apply.

- ☐ First-person Shooters (FPS), e.g. Counter-Strike
- ☐ Multiplayer Online Battle Arena (MOBA), e.g. DotA 2
- ☐ Sport games, e.g. FIFA
- ☐ Massive Multiplayer Online RPG (MMORPG), e.g. World of Warcraft
- ☐ Role-playing Games (RPG), e.g. Diablo
- ☐ Real-time Strategy (RTS), e.g. Warcraft
- ☐ Tower Defence (TD), e.g. Orcs Must Die!
- ☐ Racing games, e.g. Need for Speed
- ☐ Adventure games, e.g. Minecraft
- ☐ Fighting games, e.g. Tekken
- ☐ Casual games, e.g. Bubble Witch Saga
- ☐ Not relevant / I don't play computer games
- ☐ Other: _____

10. **How familiar are you with Tower Defence games?** *
Mark only one oval.

	0	1	2	3	4	5	
Not familiar	◯	◯	◯	◯	◯	◯	Play them on a regular basis

11. **How well do you like Tower Defence games?** *
Mark only one oval.

	0	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	◯	I love them

Questions (2/3)

This section contains all the questions about the current problem with teaching ISK and the current attempted solutions, and my game concept as a new suggested solution.

12. **What current educational games to teach Information Security have you...**
Check all that apply.

	Protection Poker	Elevation of Privilege	Other game
Heard of?	☐	☐	☐
Personally tried?	☐	☐	☐

13. **If you selected "Other game" above, please specify which one(s)**

14. **If you have tried Protection Poker, how well did you feel it taught you Information Security?**
Mark only one oval.

	0	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	◯	Very well

15. **If you have tried Protection Poker, how fun was it?**
Mark only one oval.

	0	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	◯	Very

16. **If you have tried Elevation of Privilege, how well did you feel it taught you Information Security?**
Mark only one oval.

	0	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	◯	Very well

17. **If you have tried Elevation of Privilege, how fun was it?**
Mark only one oval.

	0	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	◯	Very

18. **If you have tried another game, how well did you feel it taught you Information Security?**
Mark only one oval.

	0	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	◯	Very well

19. **If you have tried another game, how fun was it?**
Mark only one oval.

	0	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	◯	Very

20. **How important is this aspect for an educational game:** *
Mark only one oval per row.

	0 - Not at all	1	2	3	4	5 - Very
Fun factor	◯	◯	◯	◯	◯	◯
Educational content	◯	◯	◯	◯	◯	◯
Ease of use	◯	◯	◯	◯	◯	◯
Graphics	◯	◯	◯	◯	◯	◯
Performance	◯	◯	◯	◯	◯	◯
Documentation (online documentation, help pages, etc.)	◯	◯	◯	◯	◯	◯

21. **Do you think a Tower Defence sounds like a good choice for this game?** *
Mark only one oval.

- ◯ Yes
- ◯ No
- ◯ Maybe
- ◯ Not relevant / I don't know what a Tower Defence is
- ◯ Other: _____

22. **Would you try an educational game to gain more Information Security Knowledge?** *
Mark only one oval.

- ◯ Yes
- ◯ No
- ◯ Maybe

23. **What would you like to see in an educational game about Information Security?** *
Check all that apply.

- ☐ Up-to-date data
- ☐ Links to the underlying Information Security Knowledge
- ☐ Close relations to real-world Information Security
- ☐ Cool and interesting enemies
- ☐ Cool and interesting defences
- ☐ Many different types of defences
- ☐ Easy-to-use user interface
- ☐ Cool graphics
- ☐ Availability on several platforms (Windows, MAC, Android, iPhone, ++)
- ☐ Other: _____

24. **What would you MOST like to see in an educational game about Information Security?** *
Mark only one oval.

- ◯ Up-to-date data
- ◯ Links to the underlying Information Security Knowledge
- ◯ Close relations to real-world Information Security
- ◯ Cool and interesting enemies
- ◯ Cool and interesting defences
- ◯ Many different types of defences
- ◯ Easy-to-use user interface
- ◯ Cool graphics
- ◯ Availability on several platforms (Windows, MAC, Android, iPhone, ++)
- ◯ Other: _____

25. **What would deter you MOST from trying an educational game about Information Security Knowledge?** *

Mark only one oval.

- ◯ Boring game entities
- ◯ Boring graphics
- ◯ User Interface that is hard to use
- ◯ Bugs
- ◯ Lack of documentation
- ◯ Long loading time
- ◯ Low performance
- ◯ High system requirements
- ◯ Other: _____

Optional Questions (3/3)

These questions are all optional questions for additional feedback, if you feel like making some more detailed comments. If you do not have any additional feedback you would like to supply, you can scroll past this section.

26. **What are your thoughts on teaching Information Security through games?**

_____

_____

_____

_____

_____

27. **What are some faults with my game concept?**

_____

_____

_____

_____

_____

28. **What are some positive aspects of the concept?**

_____

_____

_____

_____

_____

29. **How would you improve the concept I have described?**

30. **Is there something specific I should maintain extra focus on when making this game?**

31. **How would YOU make the perfect educational game for teaching Information Security?**

32. **Any other comments or feedback?**

A.2 User Survey Results — Unknown Respondents

Data-driven Security Game: Survey

22 responses

Target group

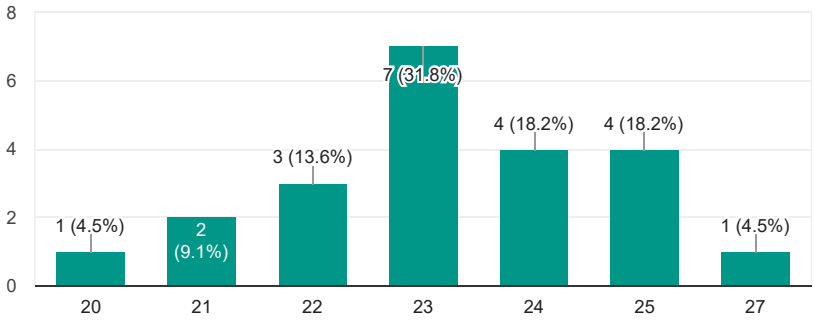
Vision

Concept sketches

Background info (1/3)

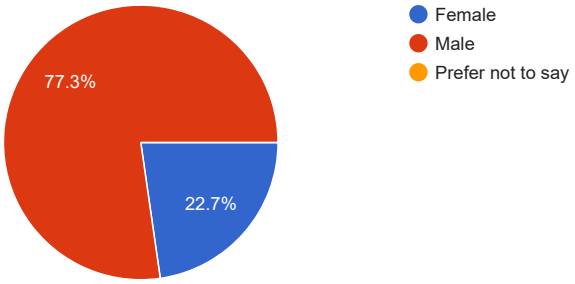
Age

22 responses



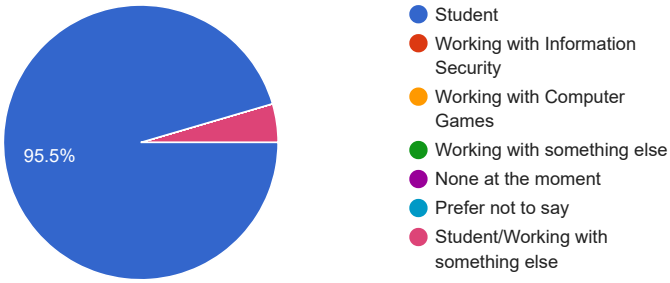
Gender

22 responses



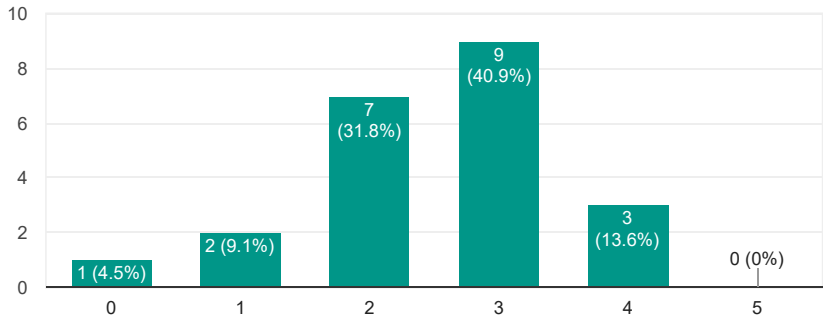
Occupation

22 responses



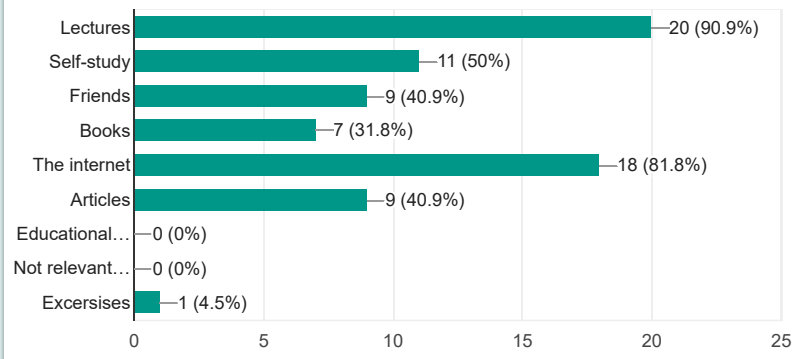
How familiar are you with Information Security Knowledge?

22 responses



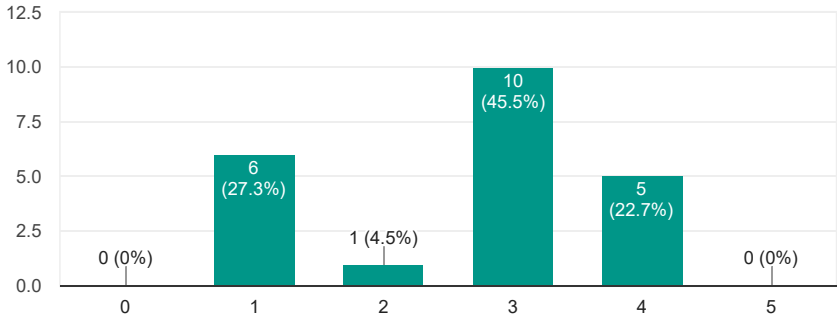
How have you learned what you currently know about IS?

22 responses



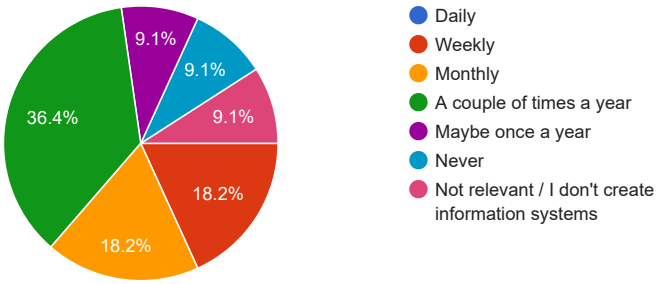
How familiar are you with programming information systems?

22 responses



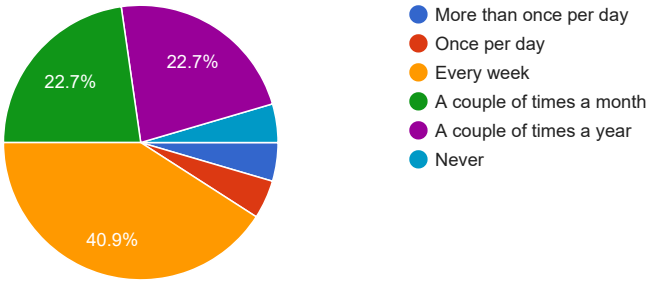
How often do you use Information Security Knowledge when programming an information system?

22 responses



How often do you play Computer Games?

22 responses

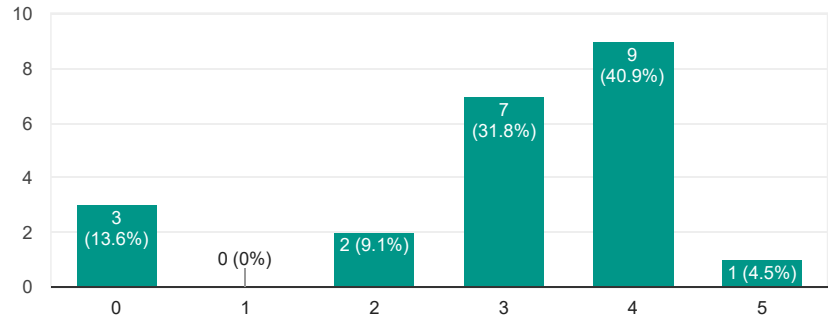


What types of games do you usually play?

22 responses

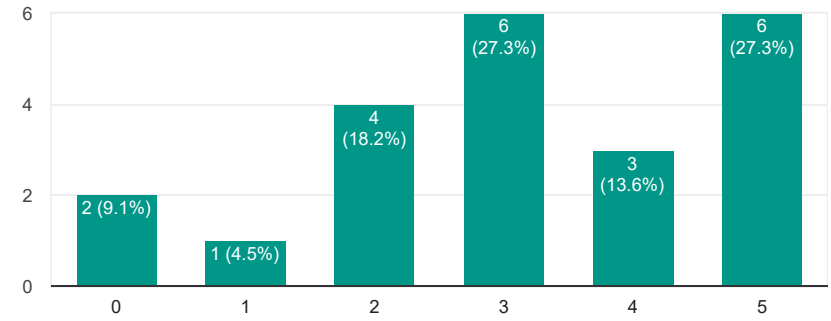
How familiar are you with Tower Defence games?

22 responses



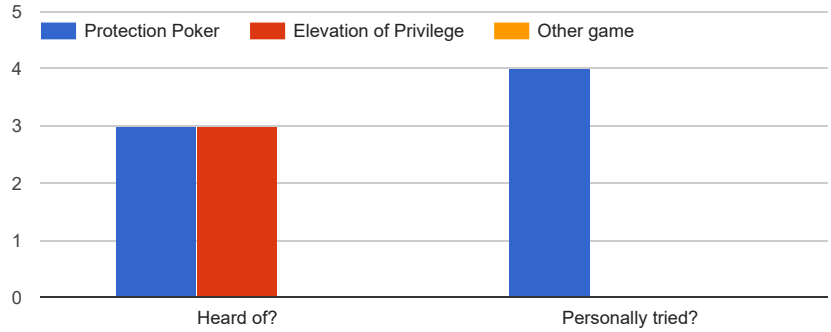
How well do you like Tower Defence games?

22 responses



Questions (2/3)

What current educational games to teach Information Security have you...



If you selected "Other game" above, please specify which one(s)

1 response

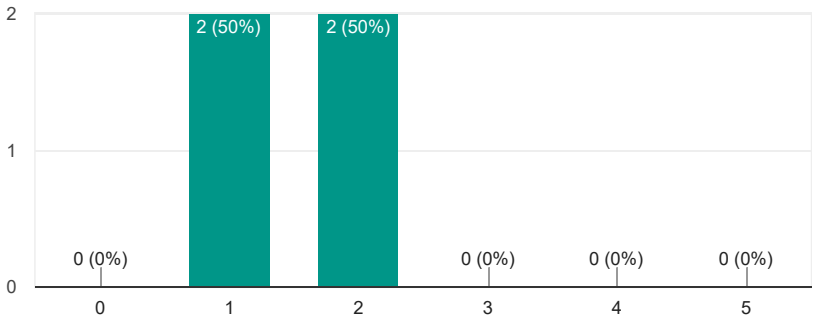
I've not been in contact with IS education games yet.

If you have tried Protection Poker, how well did you feel it taught you Information Security?

4 responses

If you have tried Protection Poker, how fun was it?

4 responses



If you have tried Elevation of Privilege, how well did you feel it taught you Information Security?

0 responses

No responses yet for this question.

If you have tried Elevation of Privilege, how fun was it?

0 responses

No responses yet for this question.

If you have tried another game, how well did you feel it taught you Information Security?

0 responses

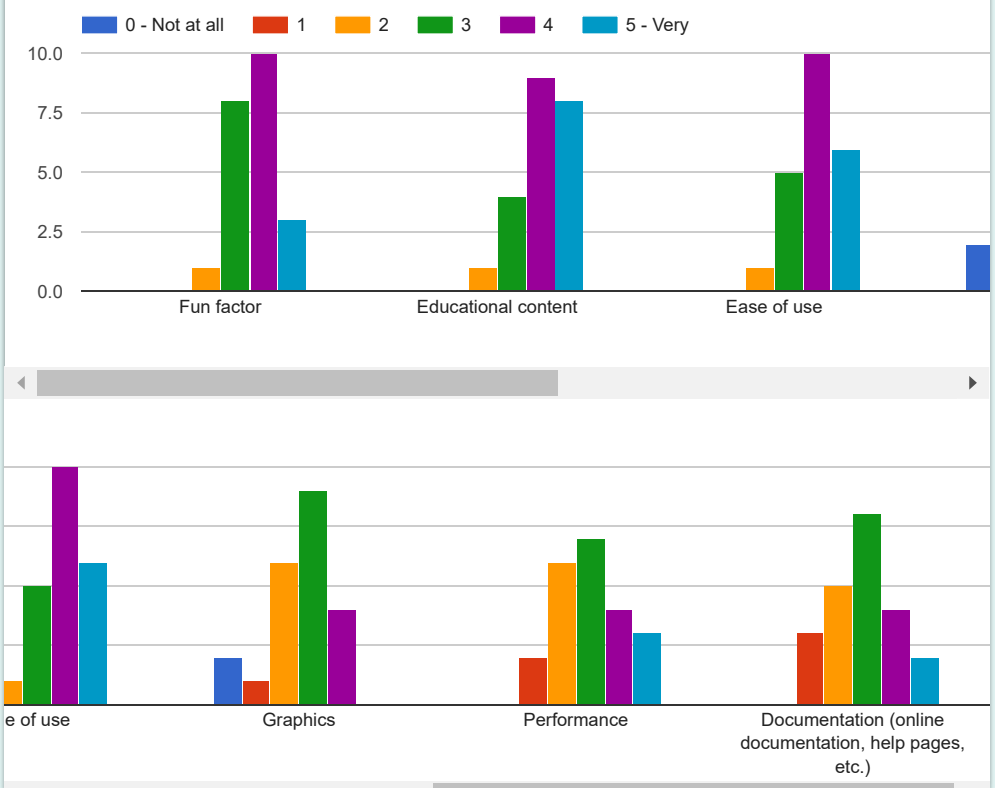
No responses yet for this question.

If you have tried another game, how fun was it?

0 responses

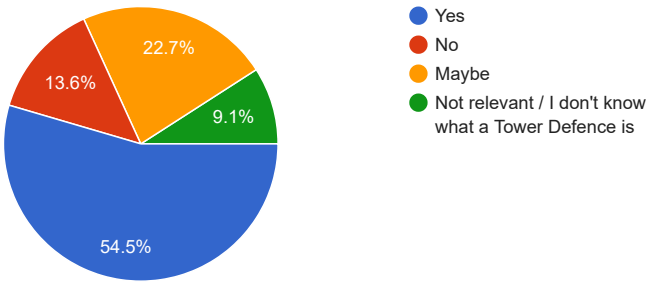
No responses yet for this question.

How important is this aspect for an educational game:



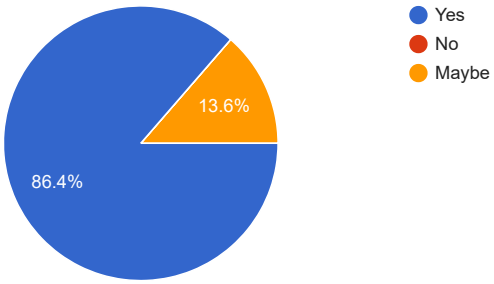
Do you think a Tower Defence sounds like a good choice for this game?

22 responses



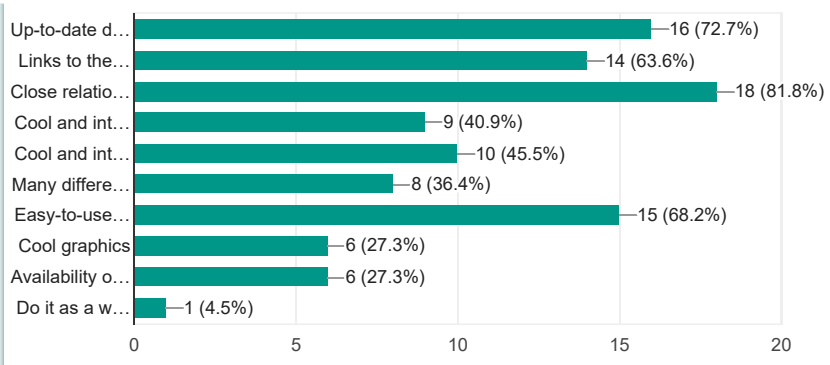
Would you try an educational game to gain more Information Security Knowledge?

22 responses



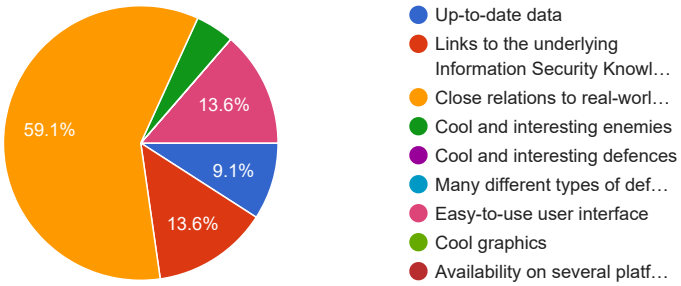
What would you like to see in an educational game about Information Security?

22 responses



What would you MOST like to see in an educational game about Information Security?

22 responses



What would deter you MOST from trying an educational game about Information Security Knowledge?

22 responses

Optional Questions (3/3)

What are your thoughts on teaching Information Security through games?

10 responses

Sounds like a very good idea!
cool ide, but as a game is a spare time thing it need to be simple enough at the start. If it is to complicated and or am not able to understand it from a none information perspective, i will probably not play it much.
Sceptical. Seems like I would spend more time learning playing this game, rather than study/hands on.
I think it is a good idea for teaching the basics
Sounds like a great idea!
It's very good idea.
Might be fun!
Teaching anything through games is always good if you can do it in a good way and make it enjoyable.
Sounds difficult, but interesting, and a good way to learn if executed well.
I personally believe in games having the capacity to teach or inform, and IS should not be any exception to this

What are some faults with my game concept?

5 responses

it cant be made for only students, it needs to be for "stupid" people. atlaset at the starting levels.
A potential time limit might stress the user to make rushed or uneducated choices
If you are gonna have a camera view that shows the whole board all the time, that might be too much information all at once? But thats also the point of a TD game i guess to see the path of the attackers, and have multiple defences. Maybe something to think about?

I'm not sure if a TD style game is well suited for learning.

Could seem a bit complex and a steep learning curve in some aspects. WOuld recommend simplifying some concepts a bit maybe?

What are some positive aspects of the concept?

5 responses

Its perfectly doable, sounds funn, interactive learning.

Tower defence is a great type of game for this purpose.

I like the format, TD fits well. Infosec games are often in a CTF (Capture the flag) style, and tower defense is kind of like CTF, where you go further and further through barriers/tasks to get to the "flag".

Good idea to keep information up to date and relevant to real world problems. Though TD might not be suited for learning, it is more easily executed than making another game type.

Clever use of existing game mechanics to teach a new subject

How would you improve the concept I have described?

5 responses

gamefy it even more. this is whats gone make people use it.

I would add some elements from RPG games. It's always motivating.

From the pictures i see you've thought about stuff like this, but since you are asking about relating the game information to real world stuff, maybe add "defences"/"towers" in the form of things are directly related to real world applications, like "login pages", and you could maybe upgrade the login page with better password hashing, salt, form validation etc. Things that make sense. I havent played a lot of TD games but usually the fun part is being able to upgrade and buy cool things. Basically as the game goes on you become powerful and you kill a lot of enemies, and you can buy cool one time use items. Game design is important.

Personally I would rather have made a strategic 2D game with a sort of story or at least some kind of text based plot/scenario in which the player must face different attack/defence scenarios and learn to prepare for them. Incorporating different scenarios into your game would make it more interesting in my opinion, but that may be because I generally don't find TD games interesting, or suited for learning about different concepts.

Perhaps implement a Tower Wars concept. It is a quite normal gametype in e.g. warcraft 3 where you have a multiplayer setting where you both build defences and send attackers to the opponent. Sending

attackers weakens the other players on a successful attack and builds income in general. Could be more engaging in a multiplayer 1v1 or a team v team setting

Is there something specific I should maintain extra focus on when making this game?

8 responses

making it understandable for people that dont have so many gaming references.

easy levels at the start, harder and more complicated as it goes. Instead of blocks simple goblin etc, piksel art would help a lot

That it is fun! A serious gaming often get boring because is focuses too much on teaching, that is too focused on learning the player something (digital lecture more then a game).

Integrate the learning in the game, so that it won't be a game that is interupted by some learning

To start with easy levels and gradually increase difficulty.

Knowledge

The most important part of any game not matter the style or genre is that it needs to have intuitive and easy to use controls/game mechanics.

It is important in my experience to focus on something that is entertaining foremost and educational while playing it.

How would YOU make the perfect educational game for teaching Information Security?

4 responses

no ide

I would do it as a web app with awesome WebGL graphics.

As mentioned in a previous answer I would base my game around story driven scenarios, either with or without a basic sub-plot such as being hired as a security consultant for a web/tech company. Though game mechanics are the most important part of a game, I feel a sort of story or plot would help making the learning aspect of this game interesting, as an arcade style game based around only the game mechanics would not supplement the learning experience in a meaningful way.

See other comments. Would perhaps try a Hero Tower Wars concept where you as a "Hero" act as an admin trying to push back attackers

Any other comments or feedback?

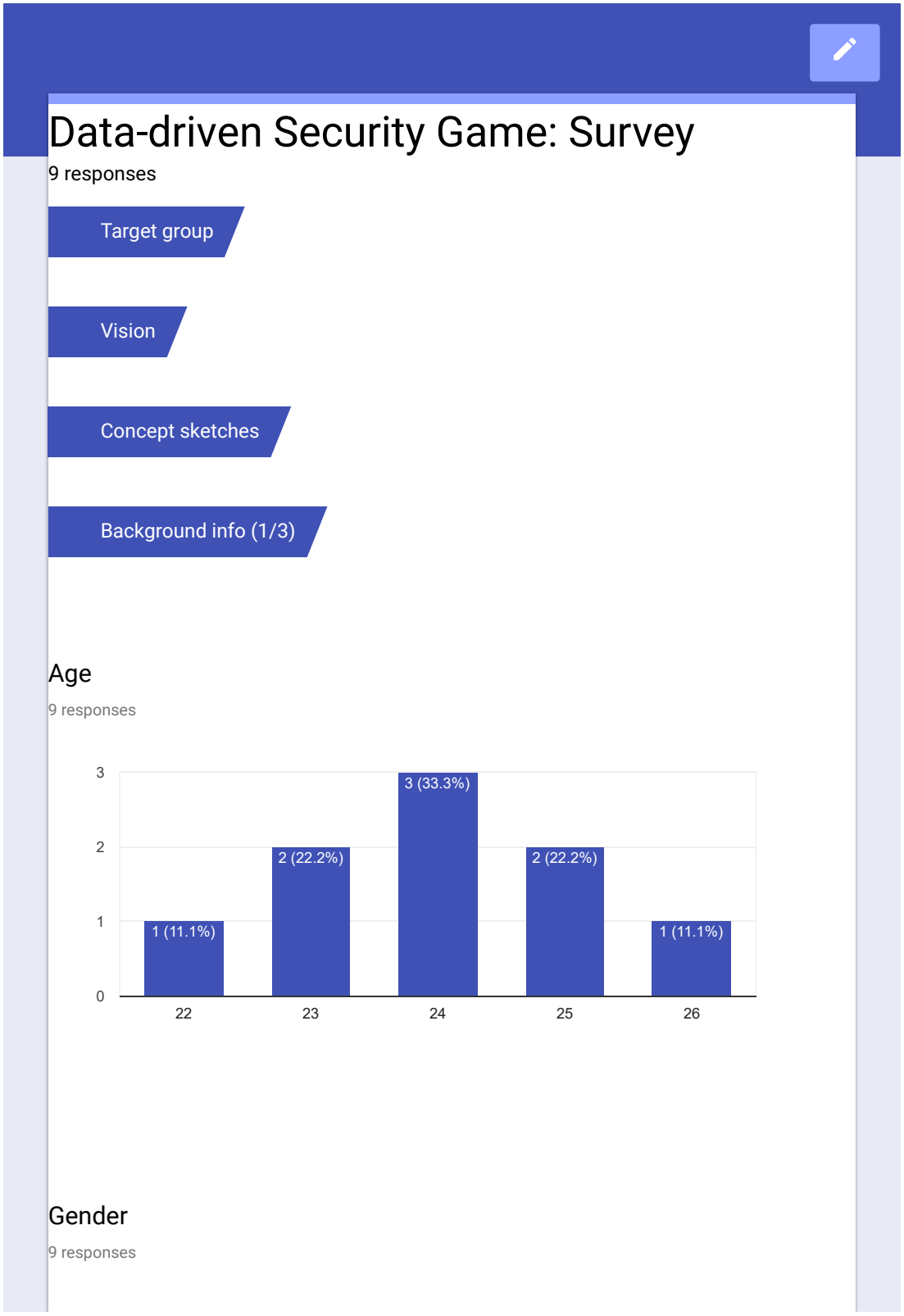
3 responses

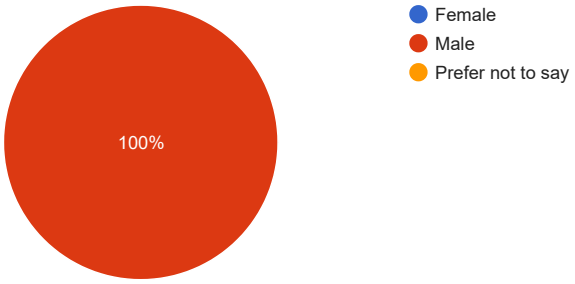
good luck

I really like the idea of making games to teach various subjects/concepts, good luck!

I would check out Line Tower Wars and Winter Maul Wars on Warcraft 3 if you want to see what i'm talking about. Perhaps check out Hero Line Tower Wars for a different concept

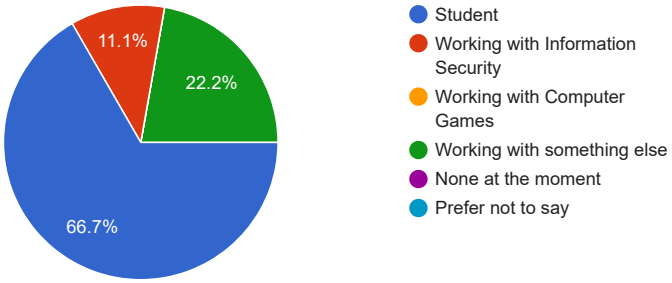
A.3 User Survey Results — Known Respondents





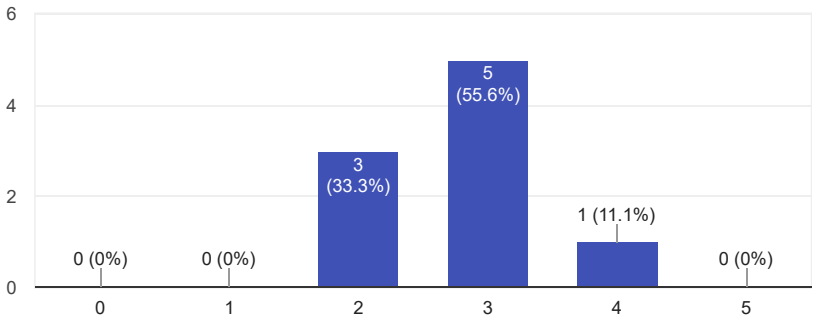
Occupation

9 responses



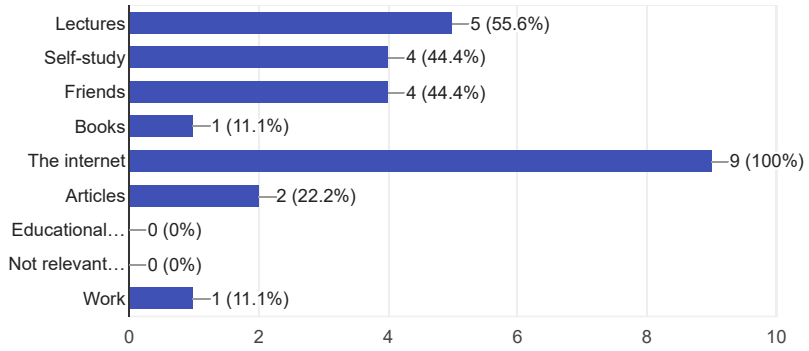
How familiar are you with Information Security Knowledge?

9 responses



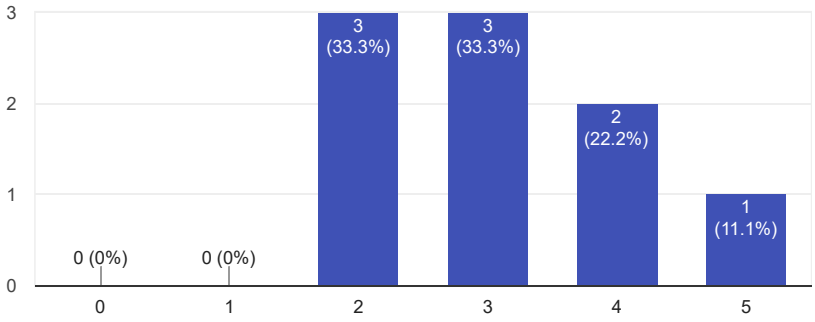
How have you learned what you currently know about IS?

9 responses



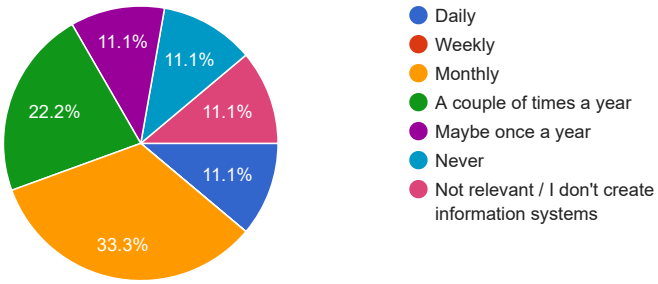
How familiar are you with programming information systems?

9 responses



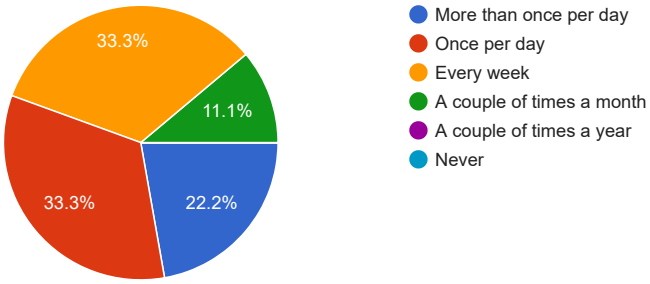
How often do you use Information Security Knowledge when programming an information system?

9 responses



How often do you play Computer Games?

9 responses

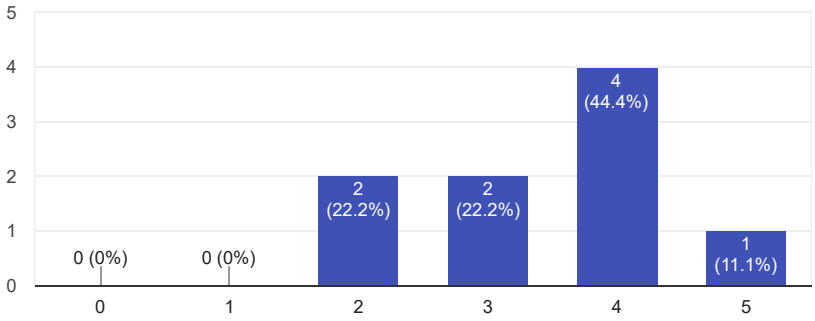


What types of games do you usually play?

9 responses

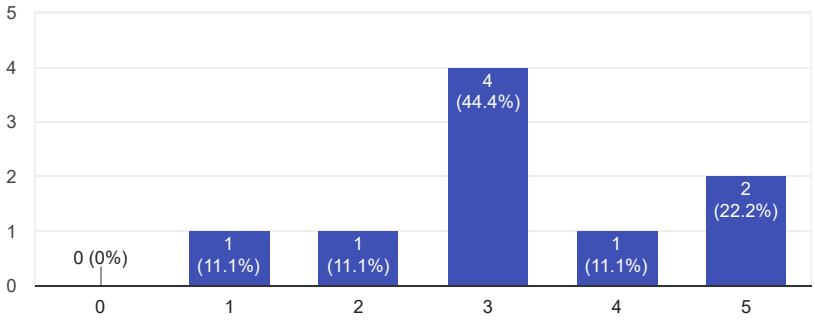
How familiar are you with Tower Defence games?

9 responses



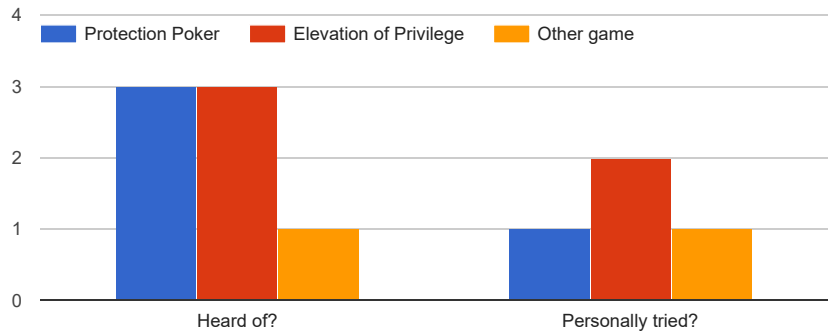
How well do you like Tower Defence games?

9 responses



Questions (2/3)

What current educational games to teach Information Security have you...



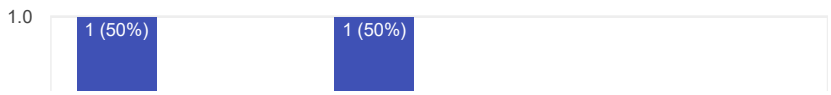
If you selected "Other game" above, please specify which one(s)

2 responses

OverTheWire
???

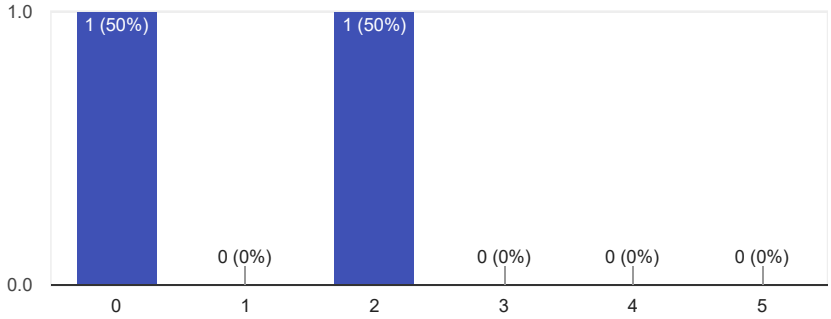
If you have tried Protection Poker, how well did you feel it taught you Information Security?

2 responses



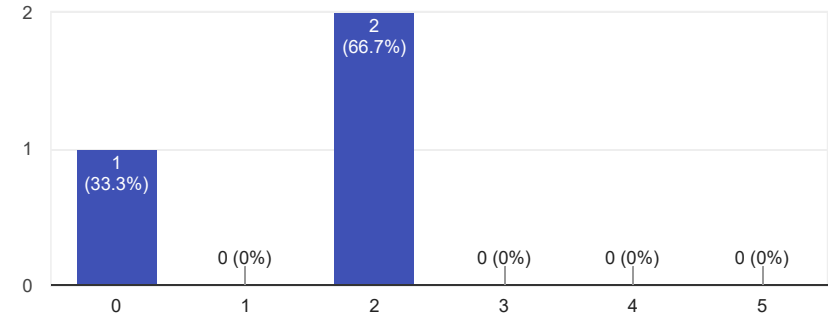
If you have tried Protection Poker, how fun was it?

2 responses



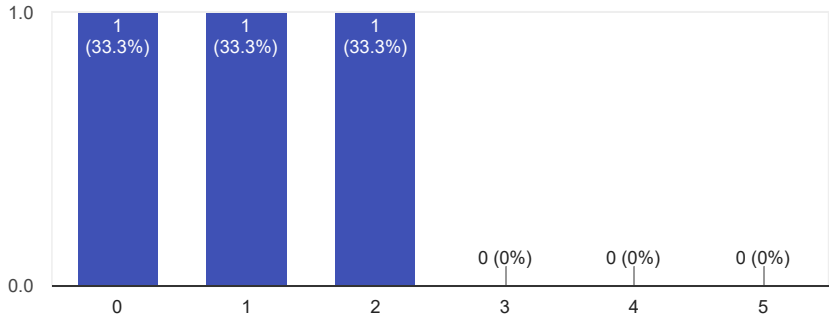
If you have tried Elevation of Privilege, how well did you feel it taught you Information Security?

3 responses



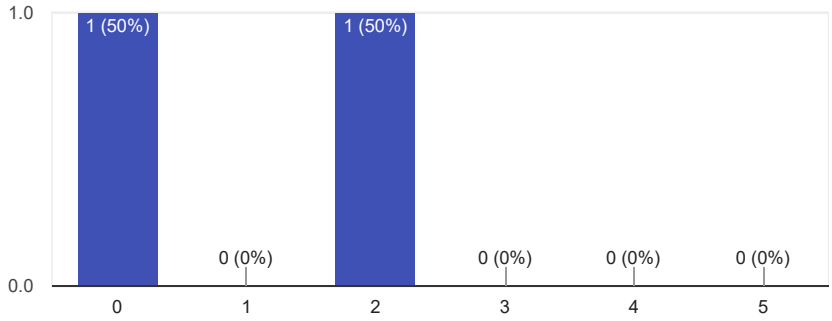
If you have tried Elevation of Privilege, how fun was it?

3 responses



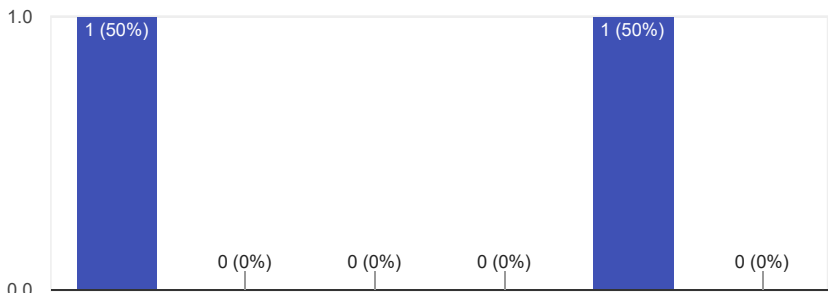
If you have tried another game, how well did you feel it taught you Information Security?

2 responses

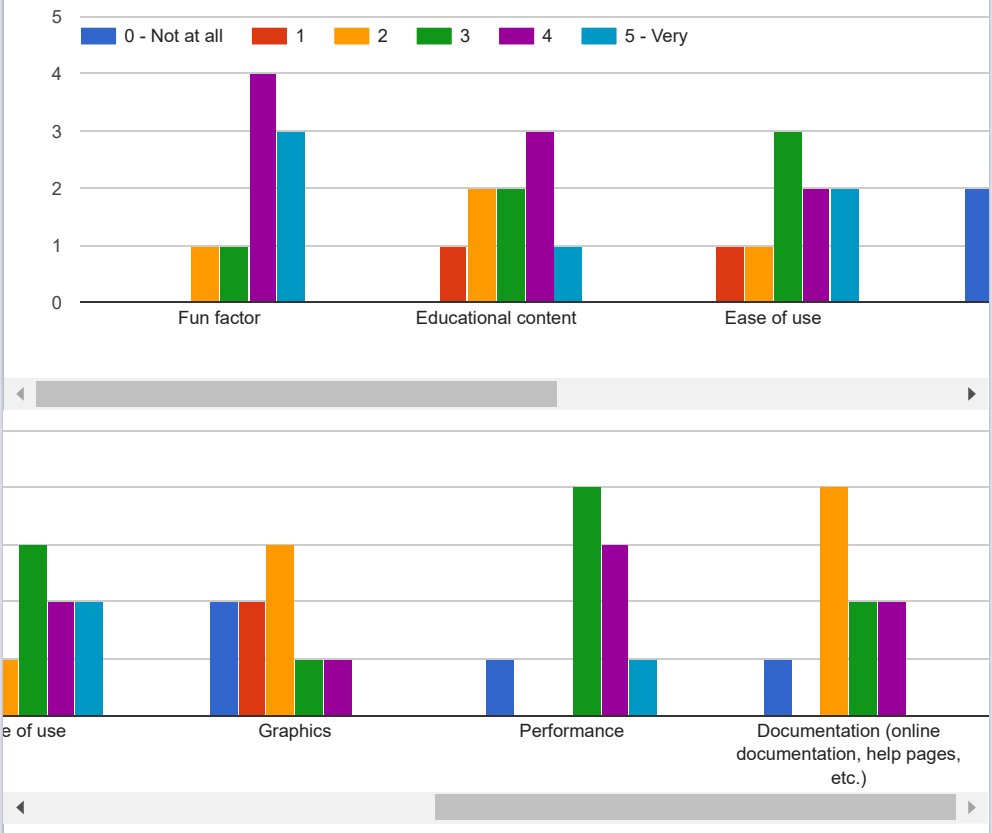


If you have tried another game, how fun was it?

2 responses

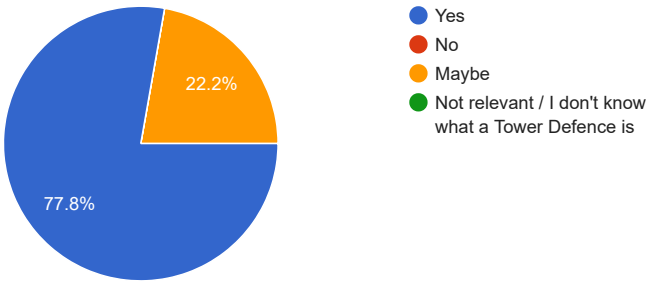


How important is this aspect for an educational game:



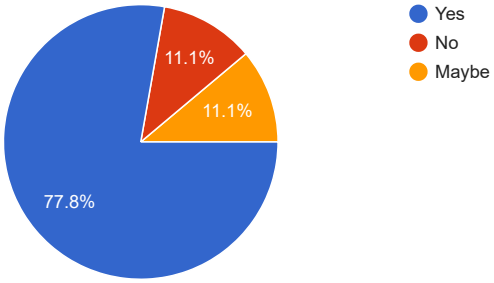
Do you think a Tower Defence sounds like a good choice for this game?

9 responses



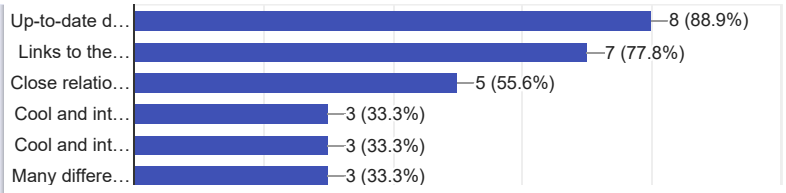
Would you try an educational game to gain more Information Security Knowledge?

9 responses



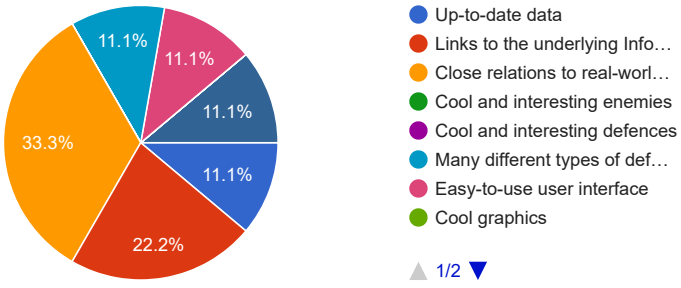
What would you like to see in an educational game about Information Security?

9 responses



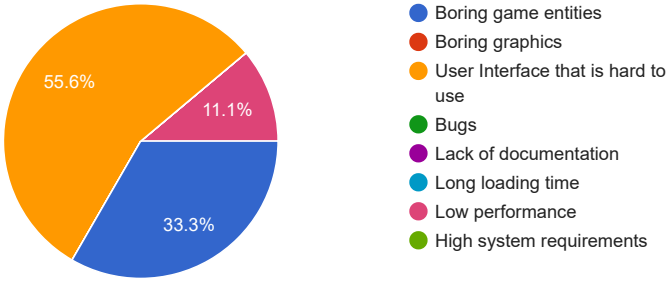
What would you MOST like to see in an educational game about Information Security?

9 responses



What would deter you MOST from trying an educational game about Information Security Knowledge?

9 responses



Optional Questions (3/3)

What are your thoughts on teaching Information Security through games?

7 responses

- Liker det
- Great way to teach the subject:)
- Good idea
- Most educational games I have tried force too much knowledge on the player. The most important thing is to make the game fun. If the game is fun and contains some educational material the player might learn something. If the game is not fun, people won't play it and won't learn anything. Gameplay needs to come first and educational aspects second.
- Seems experimental
- Seems like a great idea, might get younger people into IS.
- Alternative ways of learning are always welcome. Would have tried this if I were to study IS.

What are some faults with my game concept?

5 responses

- Kan starte lvl 1 med bare ett eller to entrypoints, unlocke entry point nr tre etterhvert??
- Unsure about how indepth the learning aspect is going to be
- I think the attacking entities should vary in seriousness based on how dangerous a security threat they are.
- How you go about visualizing the different security challenges, take stack smashing, how will you teach this concept accurately? What about side-channel attacks?
- Depends on your age range for this game, some older people might not take to games that easy if they were to use it. But as a teaching method it seems great!

What are some positive aspects of the concept?

5 responses

Det er en enkel og oversiktlig måte å visualisere hva som skjer, keep it simple

Sounds fun to play

A tower defense is a good choice for an educational game as it is a proven game mechanic. Players generally find it fun regardless of theme or topic.

There's potential for many different exploits and solutions, it's based on real-world lore, if you will.

I have limited knowledge about this, but TD seems like an interesting, creative and suitable way of educating TS.

How would you improve the concept I have described?

2 responses

Split attacks into groups based on seriousness. Maybe the player loses if just one if the really serious attacks get through? Seriousness could be based on whether a security threat would affect a single user or the whole system.

Add in an RPG element of a movable player (think of TRON).

Is there something specific I should maintain extra focus on when making this game?

4 responses

Keeping the right balance between learning and fun

Making sure that the gameplay is connected to the topic, as opposed to information security just being a theme

Focus more on making it a unique TD. There are so many TDs around. Graphics does not matter that much but gameplay and ease of use does.

It _must_ be fun.

How would YOU make the perfect educational game for teaching Information Security?

3 responses

¯_(ツ)_/¯

Spontaneously I think of something like Screeps but focused on penetration testing. Maybe players would write code to protect their 'system' as well as attack other players for access to more territory(bandwidth). Something like that

Basically a Factorio clone (since computers automate and Factorio is about automation), make defensive items exploit mitigations and the enemies and map cyber-themed (Tron).

Any other comments or feedback?

0 responses

No responses yet for this question.

Appendix B

User-testing

B.1 User-testing Questions

STIX and Stones User-testing

First off, thank you very much for participating in the user-testing of my game, STIX and Stones, which was created as my master thesis at NTNU.

Today, there are too few experts on information security, and too few developers are choosing to focus on or educate themselves in this field. With today's ever-changing cyber world, new threats are always appearing on the horizon. To be able to keep up with this development, it is important to have people with knowledge about it, and to have up-to-date information available to everyone. The sources for such knowledge exists around the internet, but they are often unstructured, enormous, and cumbersome to read. Especially for people not currently a security expert.

The idea behind this game is to create a more fun, engaging, and intuitive way for people, especially developers, to learn about information security. STIX and Stones aims to pull from the online sources continuously to stay up to date, and to deliver the information they contain in a more "edible" format, i.e. as a game, to the user. In this case: you.

To perform this user-testing, you need to have the game ready. It can be downloaded here: https://github.com/dagerikhl/ddsg-docs/raw/master/Data/STIX%20and%20Stones.zip

Please do not distribute these files without my permission.

The tasks shouldn't take more than 20 minutes. Again, thank you for participating, it's of great help!

* Required

Background

In this section, please fill out some information about your background.

1. **Age** *

2. **Gender** *
   Mark only one oval.
   - ◯ Female
   - ◯ Male
   - ◯ Prefer not to say

3. **Occupation** *
   Mark only one oval.
   - ◯ Student
   - ◯ Working with information security
   - ◯ Working within other field
   - ◯ None at the moment
   - ◯ Prefer not to say

4. **How familiar are you with computer games?** *
   Mark only one oval.

	0	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	◯	Expert

5. **How familiar are you with information security?** *
   Mark only one oval.

	0	1	2	3	4	5	
Not at all	◯	◯	◯	◯	◯	◯	Expert

Tasks

Below will follow a set of tasks that I would like you to complete with the game. Make sure you have the game executable ready. If you haven't downloaded it already, you can find it here: https://github.com/dagerikhl/ddsg-docs/raw/master/Data/STIX%20and%20Stones.zip

Please do not distribute these files without my permission.

Perform the tasks from top to bottom.

Should you not be able to complete a task, mark this in the related task question and move on to the next one.

Should the game get stuck because of a bug, because too much time has passed, or because of some other reason, please restart the game and proceed to the task you are currently on. If this happens, please state how and why the game got stuck (if you can) in the related task questions.

After each task, please answer the questions related to the task in this form.

Open the STIX And Stones application.

(If you downloaded the ZIP-archive, you need to extract it first.)

(If a window asking you to configure settings pops up, press "Play!".)

6. **How easy was it to complete?** *
   Mark only one oval.

	0	1	2	3	4	5	
Impossible	◯	◯	◯	◯	◯	◯	Trivial

7. **If impossible, what made it difficulty?**

Adjust the volume for music in the game to your preferred setting.

8. **How easy was it to complete?** *
Mark only one oval.

	0	1	2	3	4	5	
Impossible	◯	◯	◯	◯	◯	◯	Trivial

9. **If impossible, what made it difficulty?**

Start a game with 25 % difficulty level and 1x game speed.

10. **How easy was it to complete?** *
Mark only one oval.

	0	1	2	3	4	5	
Impossible	◯	◯	◯	◯	◯	◯	Trivial

11. **If impossible, what made it difficulty?**

Pause the game.

(Remember, if you need time to read the tasks or answer the questions during the subsequent tasks, you can always pause the game.)

12. **How easy was it to complete?** *
Mark only one oval.

	0	1	2	3	4	5	
Impossible	◯	◯	◯	◯	◯	◯	Trivial

13. **If impossible, what made it difficulty?**

Open the sources for the 2 incoming attack patterns in the next wave.

14. **How easy was it to complete?** *
Mark only one oval.

	0	1	2	3	4	5	
Impossible	◯	◯	◯	◯	◯	◯	Trivial

15. **If impossible, what made it difficulty?**

Move the camera to zoom in on the Server part of your system.

16. **How easy was it to complete?** *
Mark only one oval.

	0	1	2	3	4	5	
Impossible	◯	◯	◯	◯	◯	◯	Trivial

17. **If impossible, what made it difficulty?**

Select targeted asset of one of the 2 attack patterns and open its sources.

(This will require you to unpause the game).

18. **How easy was it to complete?** *
Mark only one oval.

	0	1	2	3	4	5	
Impossible	◯	◯	◯	◯	◯	◯	Trivial

19. **If impossible, what made it difficulty?**

Implement mitigations to mitigate the 2 incoming attack patterns along the correct injection vectors.

20. **How easy was it to complete?** *
Mark only one oval.

	0	1	2	3	4	5	
Impossible	◯	◯	◯	◯	◯	◯	Trivial

21. **If impossible, what made it difficulty?**

Check that all of your implemented mitigations have sufficient range to target incoming attacks.

22. **How easy was it to complete?** *
Mark only one oval.

	0	1	2	3	4	5	
Impossible	◯	◯	◯	◯	◯	◯	Trivial

23. **If impossible, what made it difficulty?**

Sell one of your implemented mitigations.

24. **How easy was it to complete?** *
Mark only one oval.

	0	1	2	3	4	5	
Impossible	◯	◯	◯	◯	◯	◯	Trivial

25. **If impossible, what made it difficulty?**

Finish the game, protecting your assets as best you can.

26. **How easy was it to complete?** *
Mark only one oval.

	0	1	2	3	4	5	
Impossible	◯	◯	◯	◯	◯	◯	Trivial

27. **If impossible, what made it difficulty?**

Check out your new highscore

(At least one game needs to be completed to perform this task.)

28. **If you had one, what was it?**

29. **How easy was it to complete?** *
Mark only one oval.

	0	1	2	3	4	5	
Impossible	◯	◯	◯	◯	◯	◯	Trivial

30. **If impossible, what made it difficulty?**

(Optional) Play the game again on whatever difficulty you want to see if you can beat your old highscore!

31. **Did you do this optional task?** *
Mark only one oval.

- Yes
- No

32. **If you did, what was you score?**

33. **How easy was it to complete?**
Mark only one oval.

	0	1	2	3	4	5	
Impossible							Trivial

34. **If impossible, what made it difficulty?**

(Optional) Play the game again on 100 % difficulty to see if you can beat the game at its hardest.

35. **Did you do this optional task?** *
Mark only one oval.

- Yes
- No

36. **If you did, what was you score?**

37. **How easy was it to complete?**
Mark only one oval.

	0	1	2	3	4	5	
Impossible							Trivial

38. **If impossible, what made it difficulty?**

Exit the game.

39. **How easy was it to complete?** *
Mark only one oval.

	0	1	2	3	4	5	
Impossible	◯	◯	◯	◯	◯	◯	Trivial

40. **If impossible, what made it difficulty?**

Evaluation

Here are some general questions about your experience when playing STIX and Stones.

This is to provide me with feedback about what was good, what could be better, and what was terrible about the game.

Please try to answer these questions as honestly and precisely as possible, thank you.

41. **Learning Value** *
How much did you feel the game improved your knowledge of the following subjects? (0 - Not at all, 5 - Enormously.)
Mark only one oval per row.

	0	1	2	3	4	5
Assets	◯	◯	◯	◯	◯	◯
Mitigations	◯	◯	◯	◯	◯	◯
Attack patterns	◯	◯	◯	◯	◯	◯
Information security in general	◯	◯	◯	◯	◯	◯
Computer games	◯	◯	◯	◯	◯	◯
Tower defence games	◯	◯	◯	◯	◯	◯

42. **Usability** *

To what degree do you agree with the following statements? (0 - Strongly disagree, 5 - Strongly agree.)
Mark only one oval per row.

	0	1	2	3	4	5
I think that I would like to use this system frequently						
I found the system unnecessarily complex						
I thought the system was easy to use						
I think that I would need the support of a technical person to be able to use this system						
I found the various functions in this system were well integrated						
I thought there was too much inconsistency in this system						
I would imagine that most people would learn to use this system very quickly						
I found the system very cumbersome to use						
I felt very confident using the system						
I needed to learn a lot of things before I could get going with this system						

43. **Assessment** *

To what degree do you agree with the following statements? (0 - Strongly disagree, 5 - Strongly agree.)
Mark only one oval per row.

	0	1	2	3	4	5
I thought this game was fun to play						
I found the purpose of the game confusing						
I feel more interested in information security after playing the game						
I thought there was too much information in the game						
I felt I had adequate control over how difficulty I wanted the game to be						
I found the game too challenging						
I want to play the game again						
I experienced many bugs while playing the game						
I want to see this game developed further						
I would rather learn about information security in a normal lecture than through this game						

Optional feedback

In case you have anything more to add, this is the place to let me know.

44. **I found this especially good about the game**

45. **This could be improved in the game**

46. **I found this especially bad about the game**

47. **General feedback**

B.2 User-testing Results

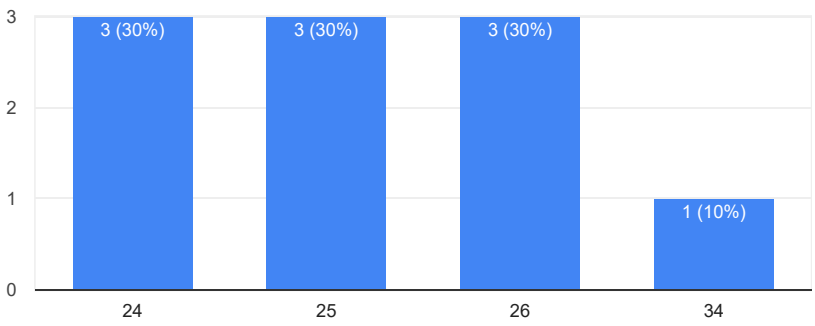
STIX and Stones User-testing

10 responses

Background

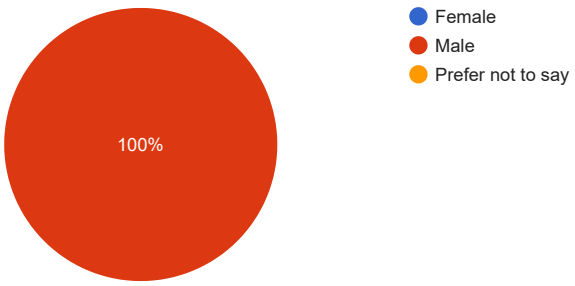
Age

10 responses



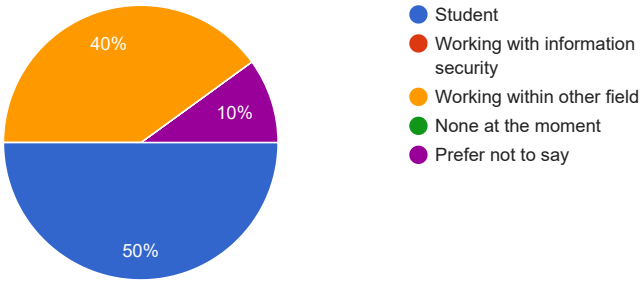
Gender

10 responses



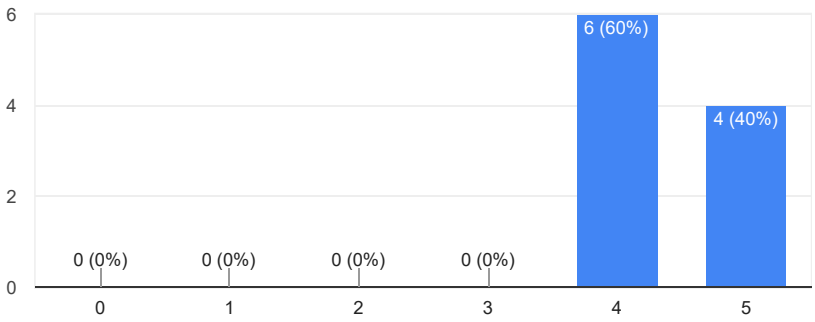
Occupation

10 responses



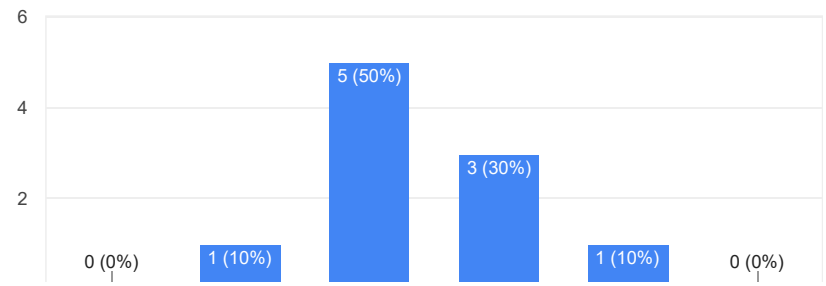
How familiar are you with computer games?

10 responses



How familiar are you with information security?

10 responses

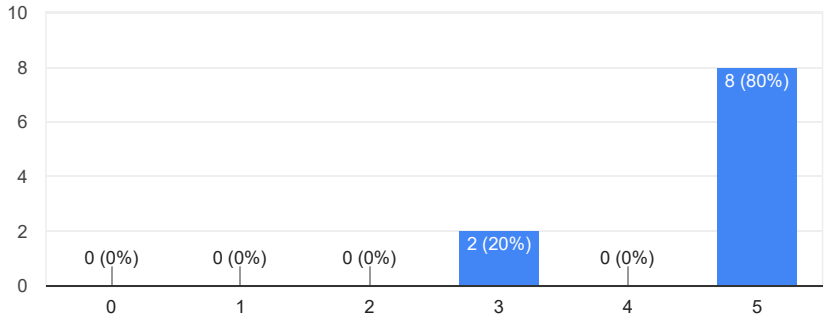


Tasks

Open the STIX And Stones application.

How easy was it to complete?

10 responses



If impossible, what made it difficulty?

2 responses

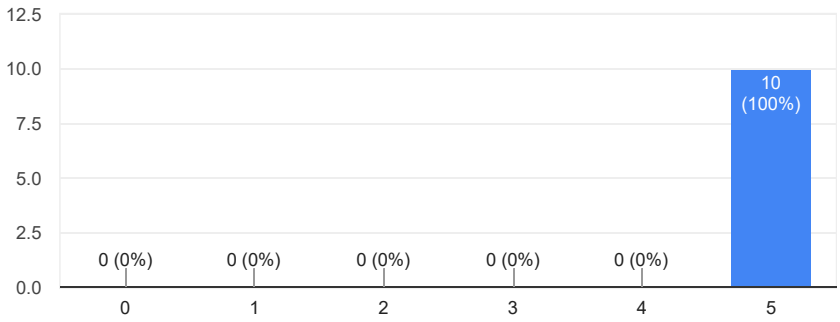
Anti-virus blocked it at start up, had to whitelist it to boot the game.

Spaces in filenames

Adjust the volume for music in the game to your preferred setting.

How easy was it to complete?

10 responses



If impossible, what made it difficulty?

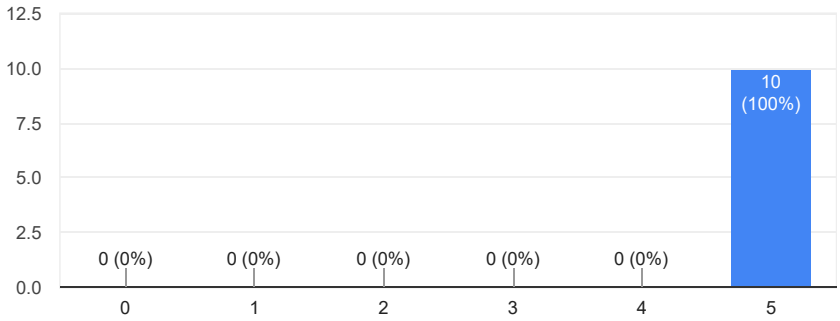
0 responses

No responses yet for this question.

Start a game with 25 % difficulty level and 1x game speed.

How easy was it to complete?

10 responses



If impossible, what made it difficulty?

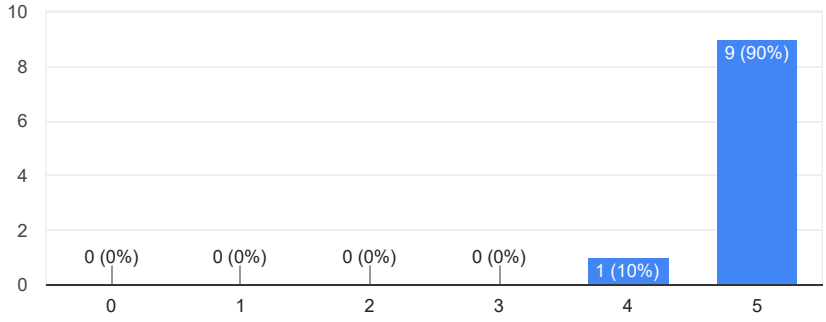
0 responses

No responses yet for this question.

Pause the game.

How easy was it to complete?

10 responses



If impossible, what made it difficulty?

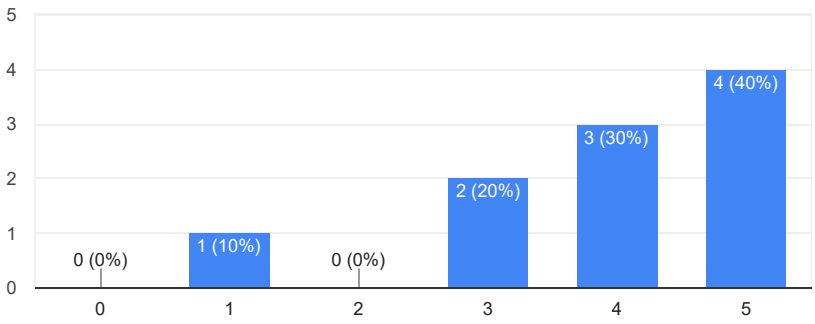
1 response

A bit annoying that you can't use the arrow keys to move the screen when the game is pause but you can use the mouse.

Open the sources for the 2 incoming attack patterns in the next wave.

How easy was it to complete?

10 responses



If impossible, what made it difficulty?

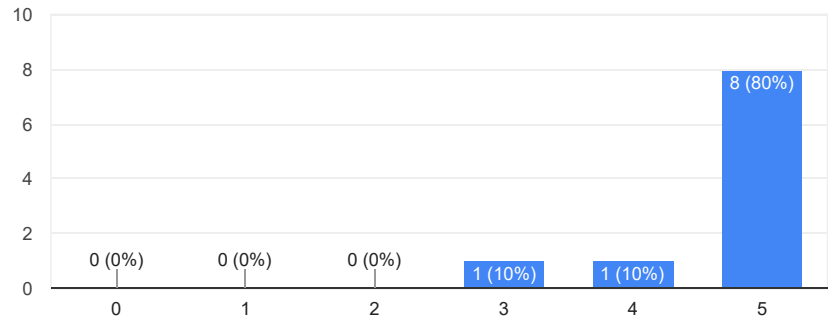
1 response

Hmm, hard to see "Right-click to open all in browser"

Move the camera to zoom in on the Server part of your system.

How easy was it to complete?

10 responses



If impossible, what made it difficulty?

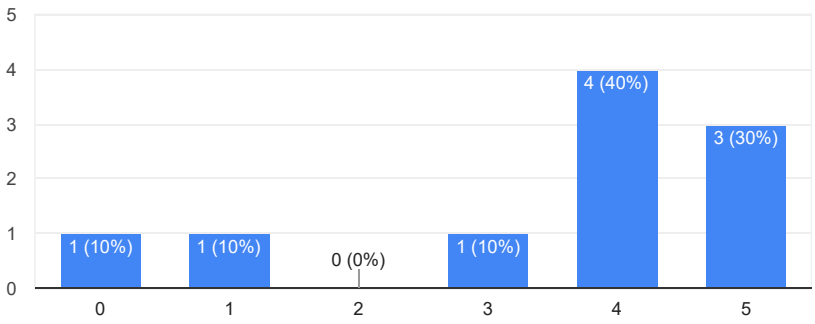
1 response

Can only move the camera position when the game is unpaused.

Select targeted asset of one of the 2 attack patterns and open its sources.

How easy was it to complete?

10 responses



If impossible, what made it difficulty?

4 responses

Right clicking is maybe not the best choice of key for this action as it is easy to missclick and get sent out of the game. Maybe something "ctrl+left mouse button" would be better?

It the source for the first attack it mitigated.

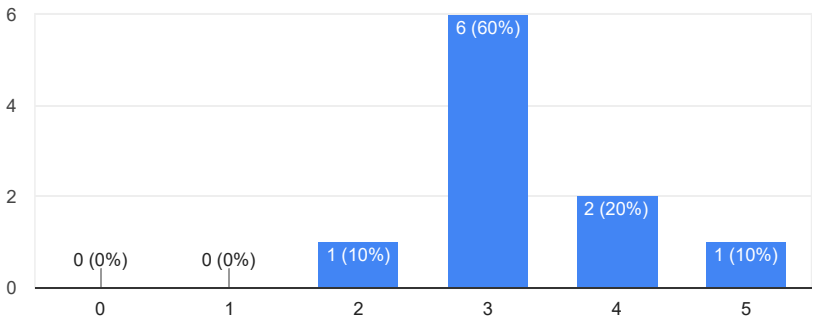
I thought "client" was the thing labeled "client"

I don't think it requires unpausing the game?

Implement mitigations to mitigate the 2 incoming attack patterns along the correct injection vectors.

How easy was it to complete?

10 responses



If impossible, what made it difficulty?

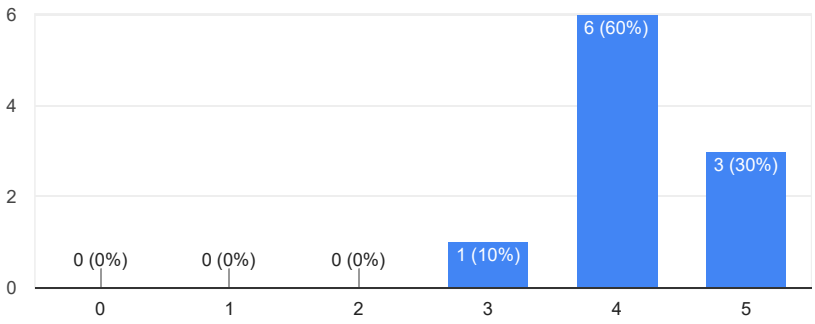
1 response

A bit difficult to figure out which lane will be attacked as there is no tutorial. The way the GUI is designed you have to move a lot between the attackers tooltip and the defences to figure out what you can use. This forces you to read a lot of the text which is good from a learnig perspective but quickly gets annoying from a gaming perspective. Perhaps some sort of grouping or colour coding of the attackers and defendors could improve on this.

Check that all of your implemented mitigations have sufficient range to target incoming attacks.

How easy was it to complete?

10 responses



If impossible, what made it difficulty?

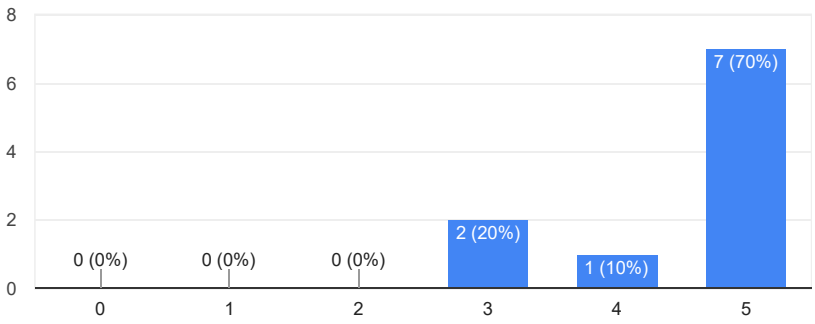
1 response

Makes it harder to be unable to do it while paused.

Sell one of your implemented mitigations.

How easy was it to complete?

10 responses



If impossible, what made it difficulty?

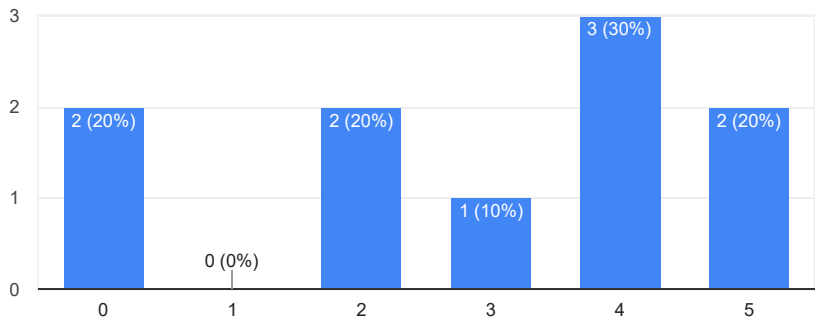
1 response

The "sell" and "open" buttons are on top of the info text. The open button feels unnecessary as you can right click to go to source.

Finish the game, protecting your assets as best you can.

How easy was it to complete?

10 responses



If impossible, what made it difficulty?

3 responses

It is very difficult. If you don't manage to stop the attackers you don't get any gold which means you can't add any more defenses. This means that it is nearly impossible to recover from one failed wave which detracts from the gameplay. This might however teach a valuable lesson in that big security flaws can quickly bankrupt your company.

To know how mutch DPS is needed

I'm terrible at tower defense

Check out your new highscore

If you had one, what was it?

7 responses

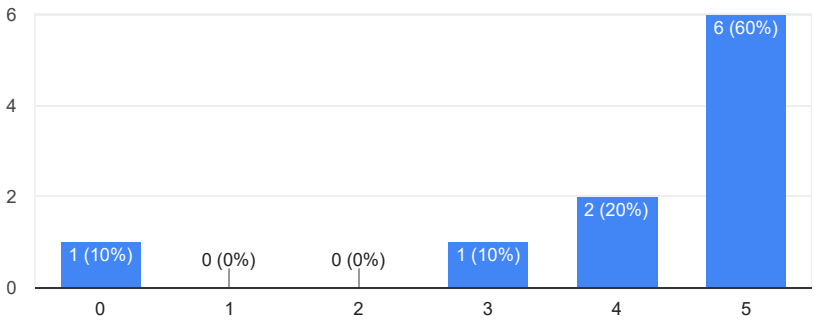
36250

213500

4500
98250
4000
5000
89750

How easy was it to complete?

10 responses



If impossible, what made it difficulty?

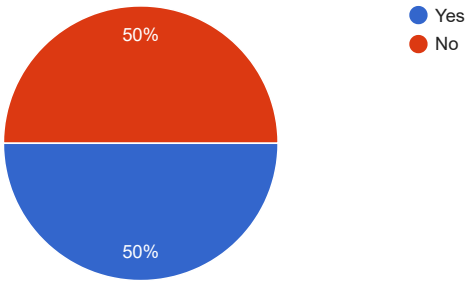
3 responses

Easy to check highscore but difficult to get one. Maybe you should get a highscore also when you fail to complete the game in order to motivate the player to keep trying?
played at 50% difficulty, not the give 25%
The score floated outside the screen

(Optional) Play the game again on whatever difficulty you want to see if you can beat your old highscore!

Did you do this optional task?

10 responses



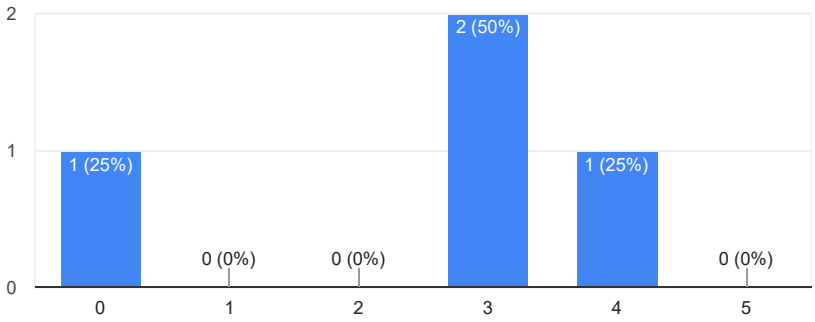
If you did, what was you score?

3 responses

3500
11500
148000

How easy was it to complete?

4 responses



If impossible, what made it difficulty?

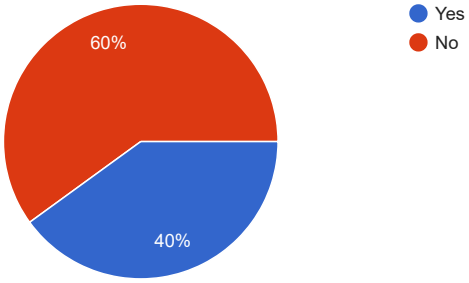
1 response

I tried it on 50%. I was not able to get past wave 3 in four attempts.

(Optional) Play the game again on 100 % difficulty to see if you can beat the game at its hardest.

Did you do this optional task?

10 responses



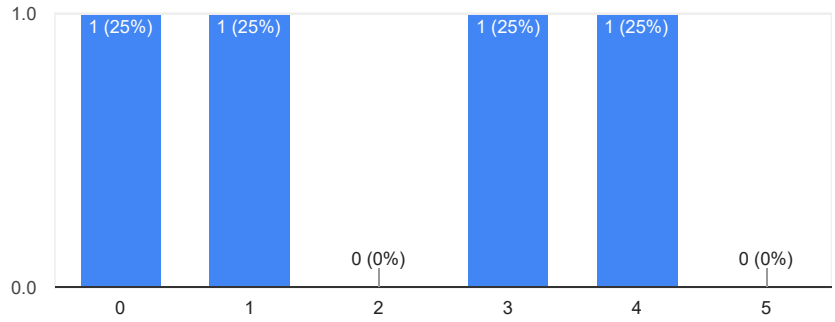
If you did, what was you score?

3 responses

120000
-2000
68000

How easy was it to complete?

4 responses



If impossible, what made it difficulty?

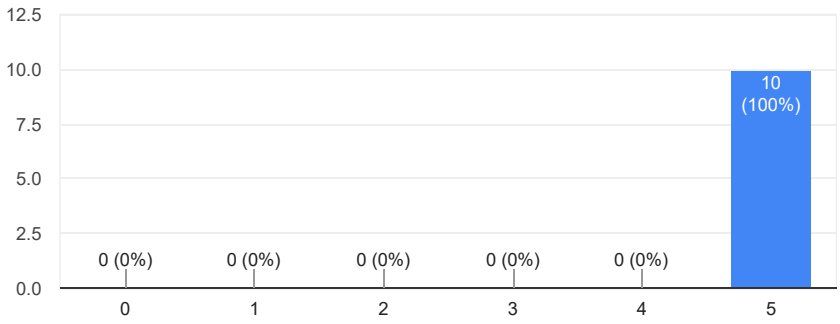
1 response

impossible. Not enough gold to build defenses

Exit the game.

How easy was it to complete?

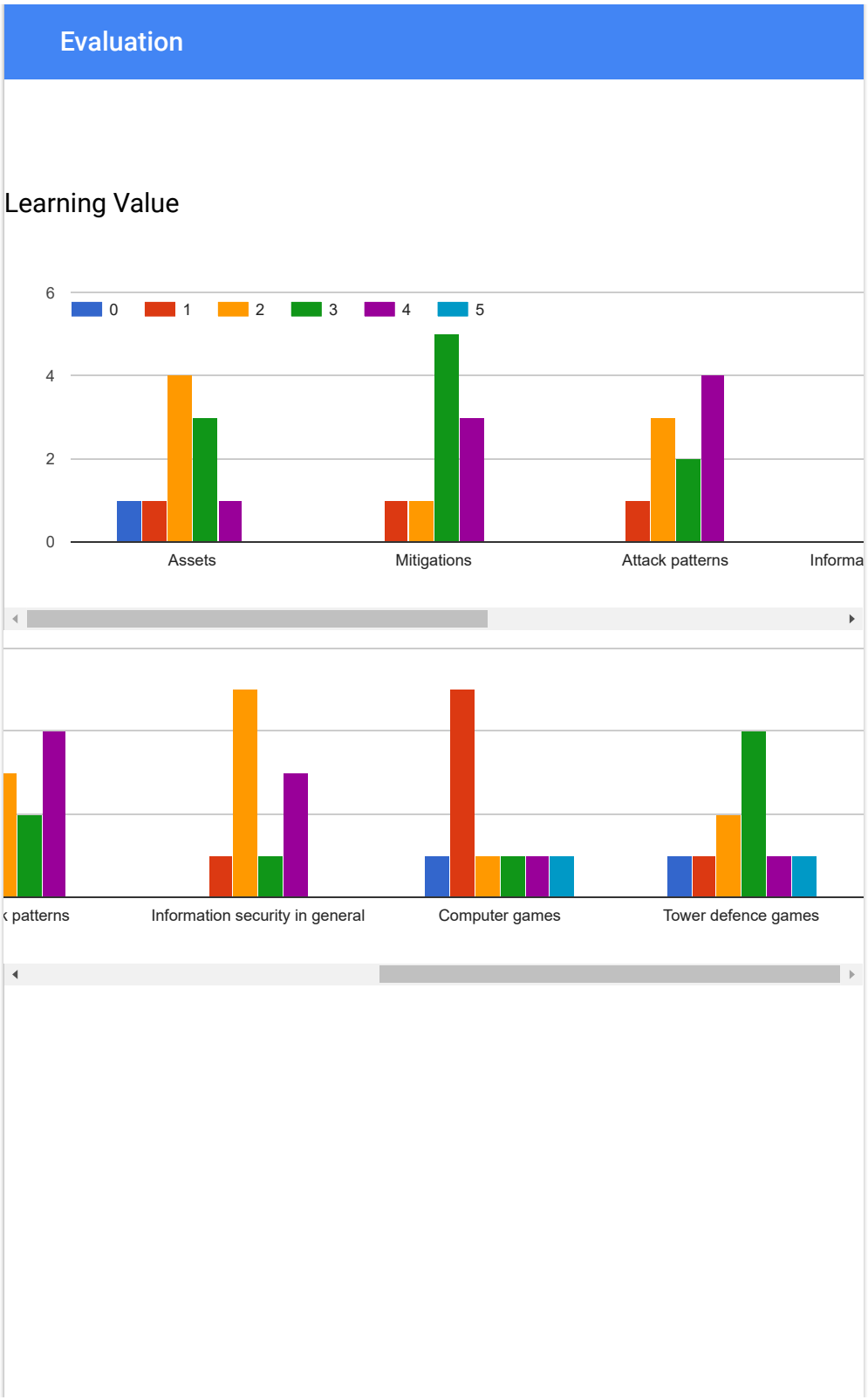
10 responses



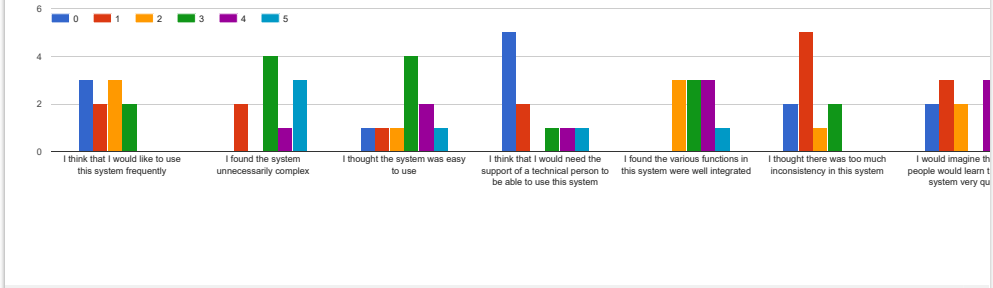
If impossible, what made it difficulty?

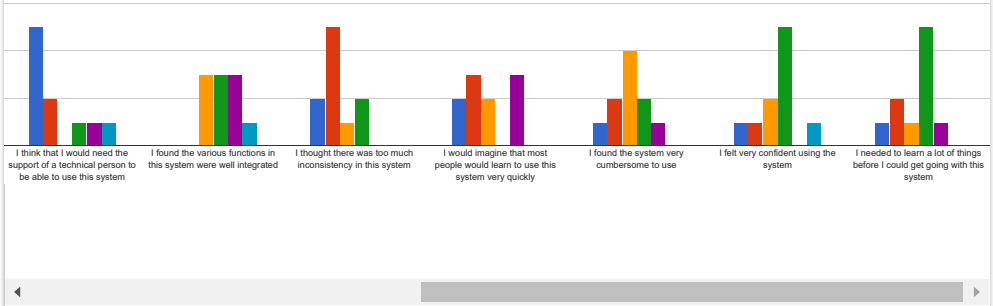
0 responses

No responses yet for this question.


Evaluation
Learning Value
6
4
2
0
0
1
2
3
4
5
Assets
Mitigations
Attack patterns
Informa
k patterns
Information security in general
Computer games
Tower defence games

Usability


6
4
2
0
0
1
2
3
4
5
I think that I would like to use this system frequently
I found the system unnecessarily complex
I thought the system was easy to use
I think that I would need the support of a technical person to be able to use this system
I found the various functions in this system were well integrated
I thought there was too much inconsistency in this system
I would imagine th people would learn t system very qu


I think that I would need the support of a technical person to be able to use this system
I found the various functions in this system were well integrated
I thought there was too much inconsistency in this system
I would imagine that most people would learn to use this system very quickly
I found the system very cumbersome to use
I felt very confident using the system
I needed to learn a lot of things before I could get going with this system

Assessment

Optional feedback

I found this especially good about the game

7 responses

1. The level design. The client, network and server lanes made a simple TD quite interesting.
2. There is a lot of content(choice of enemies and defenses) for a university project.
3. I experienced no game breaking bugs.
4. Most games for learning I have tried haven't really taught me anything. In this game however I found the learning part quite interesting. The game was so difficult that you had to read a lot of the text to stand a

chance. The theme of the game is highly relevant to my work. And finally what you learn from the game is very concrete.

I didn't encounter any breaking bugs, the visuals were aesthetically pleasing and it ran smooth on it's supposed resolution. The music worked and volume was good for it. The option for sources was intriguing and implemented in a good way with little to no problems in moving from the game to the source in a new window. Given my more competitive nature I filtered out most of the information given as it was way too much, in order to defend best possible.

You could easily click on the sources to learn more

Good tower defense game. Looks good. In-depth about information security

Tower defence ftw

Good UI, good concept, solid tower defense gameplay

Most of the game was well-developed, with a functional user interface and decent graphics.

This could be improved in the game

8 responses

1. The player should be given more gold for killing enemies and some gold between each round to enable the player to recover from a bad round.
2. A tutorial overlay that describes where to find what info at the start of the game would make it easier to get into.
3. The player was introduced to all of the content at once which makes it more difficult to get into. Maybe add a "campaign" mode which is more scripted and starts with fewer enemies and defenses to choose from.

To continue on the above point, the game could be greatly improved with some kind of tutorial giving the players a brief overview of each of the different parts; servers being attacked, paths for attacks to come from, different defenses and how they work against the attacks, for a learning experience having the information divvied up and given in smaller doses is also better and gives a sense of accomplishment for the user (resulting in a higher learning outcome). The randomness of waves and attacks is also something that can be improved, as it gives a feeling of luck of the draw with little balancing, this might be okay for end levels, but should not be the first one.
Tower targeting: the targeting of towers is off if placed in the middle of a wave, refusing to shoot on the attacker closest to server.
Name difference: some names are different in the "mitigated by" section and the defense section, namely 'privileged zone restrictor' as opposed to 'restrict privileged areas'

Icons for the assets so they are more easily recognized. Music got repetitive.

Polish (small bugs and UI problems). Needs to be easier to select correct mitigations. Needs a tutorial explaining what things mean. Labels above database so I dont have to hover over it. You should highlight the matching mitigation when I hover over an enemy/wave attack

The game was way too complex. Too many "Towers" and too little time to know whats what. Impossible to know what lane the enemies came from.

Tooltips from bought assets went out of the screen

A few UI bugs here and there. Could use a tutorial.

A few bugs, like the tooltips not scaling with the windowborder, ie. going through the top of the screen when selecting/hovering minions/towers.
It would be nice to be able to select and read the info about the creep waves/towers while paused.

I found this especially bad about the game

8 responses

The game sound, was too loud with headset on. Had to exit the game and keep my headphones up until the game booted up.

1. The GUI relating to the enemies and towers was a confusing. The positioning of the current/next wave made me believe that it had something to do with which lane they would arrive in. Could be mitigated with some form of colour coding maybe.

The information given for my first few games was just too much, the attacks dictated a wide array of defenses and needed a lot of thought and reading to process. The intention is on learning and for that to happen a tutorial level introducing aspects of the game would be good.
Bug: game doesn't end or give a victory screen when server is damaged but not broken (I think is what set it off)

Confused about if the wave would attack the client, network or server. Maybe right clicking on the icons aren't the best way for opening the source, I first tried to click on the link itself. In the future it would be best if the source (with the additional information) was opened in game as a pop-up. The different attacks should have different different styles.

Too much information at one time. Too slow (Boring to wait for one attack to walk from client to server). Names on mitigations did not match some places. Right click to open a link is not intuitive.

Too short. Should also have presented "towers" different (upgrades, clear categories etc.)

Lack of incentive to learn about the information security concepts beyond which components mitigates which attacks.

There was no tutorial, and can be considered overly challenging for novice players/developers.

General feedback

7 responses

Hard to see at first where the attacks first came, when you just jump into it. But when seeing where it was coming it became rather trivial to prepare for an attack. The pause feature is a must, but could be added a " Start next wave" button to make the gameplay abit faster when you've prepared.

I think the game can teach the player a lot but I think the gameplay needs to be improved in order to keep the player interested.

The game and its features are working well from a technical standpoint, it is the information and amount of it that feels difficult to process. The game runs smoothly, looks visually pleasing and is generally easy to control and navigate. Evolving the game with a tutorial and increasing fixed levels with the goal of learning is not that far off if needed. A solid foundation to build from. 5/7 not as good as bloons TD5

Needs a bit more polishing, but I can see the value of this game if more time is spent on improving it. I think the game itself is a good idea.

Impressive work. Took some time to understand the mechanics (maybe I didn't read survey thorough enough). Maybe add "how to play" in the game to make it it independent of this survey? To be honest I have little to none interest in information security. This game would be a nice introduction to IS if I were to study the subject.

I think it will be a great game if given more polish.

The game is well made. Depending on the players background, it may seem to be too complicated at the first glance, thus throwing people off, giving them less lust to play the game.
User-interface, graphics, sound is good. At times hard to place the towers where you want them, possibly related to the 3D-function of the game, often not found in TD games.
The game would benefit from a tutorial, and some sort of progression, eg. first set of games with less minion and tower combinations, to make the game seem less overwhelming.