Analysing Capture the Flag Competitions

MSc Cybersecurity

CMT400 - Dissertation

Author: Francesco Iulio [2101327]

Supervisor: Neetesh Saxena

5th November 2021

Abstract

While existing studies related to capture the flag events have interested mainly the organising and the educational benefits of such competitions, this research focuses on participants and their thinking process, valuing strategies, tools and procedures that can be extrapolated from the event's raw data available online.

The possibilities of improving future competitions and give the participants a better experience may increase thanks to a detailed analysis that will highlight statistics about attacks, procedures, common mistakes and system site capabilities, investigating the tools and the commands retrieved from competition's raw data sources to suggest a methodical procedure that can enable participants to understand the competitions, use the right tools and acquire the right mindset to collect flags.

This paper will first introduce concepts about Game Theory, Adversarial Thinking and Network Forensics, and then discuss suggestions to better face and adapt to challenges in capture the flag events, following an extensive analysis of the DEF CON 28 capture the flag final competition, called *Safe Mode* as it was held remotely for the first time in its history, due to the restrictions imposed in many states as a consequence of the start of the Covid-19 pandemic in 2020.

This thesis studies the network traffic and the game state data from the event, leveraging command-line tools to process information contained in large volumes of network traffic data, and manually correlating the results with the game state data, in order to identify the winning strategies that appeared to be employed by teams during the competition. The results show how strategies such as traffic monitoring, flag stealing, obfuscating communications, reusing exploits, backdooring files and deceiving adversaries can prove successful in a capture the flag environment.

Acknowledgment

It was never about the finish line rather the journey, and I am glad that still a long way is left before I reach my destination.

A heartfelt thank you to my family and my cousin Francesco, who gave me support during the tough lockdown we endured for over a year.

Furthermore, I want to extend my gratitude to Dr. Neetesh Saxeena for his support, availability and valuable constructive suggestions, always inspiring and motivating.

Thank you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The attacks to critical infrastructure [2], industrial control systems (ICS) [3] and supply-chains [4], together with the growing number of ransomware attacks occurring around the world [5, 6, 7], have made cyber warfare training particularly important to defend organizations worldwide. It would not be possible to practice offensive and defensive capabilities, and achieve tactical and strategic goals in a cyber conflict, with the lack of a dedicated environment to train in the advanced hacking skills necessary to efficiently respond to incidents. Capture the Flag (CTF) competitions are increasingly becoming the main proving field for hackers and computer security enthusiasts who want to test their skills on several types of cybersecurity related challenges. These events have been organized since 1993 [8], gaining an excellent reputation and being joined by increasingly more people over the years. Due to the fact that points during the competitions are collected through practical execution of tasks and commands over the network, often including several trial-and-error attempts made by participants, the analysis process could become a very daunting task, full of false positives and unrelevant data. In regards to analysing capture the flag competitions from a forensic point of view, the literature available is limited as there are many researches addressing the pedagogical and psychological aspects of such events, and a few others comparing some of the competitions held and their technicalities, however the only study in regards to analysing tactics and strategies in a capture the flag competition, parsing through the data produced by participants during the live action of the contest, was done by Yam in 2016 [9].

1.1 Research Motivation

This research is motivated by the willingness of exploring the strategies and tactics used during Attack-Defense competitions, that could be applied to real-time cyber conflicts, as well as the interest in covering techniques to extract valuable information from network packet captures. It is worth mentioning that while researchers have deeply studied the pedagogical benefits, compared challenges, and evaluated the character of the individuals more prone to participating to capture the flag contests, the actions and the decisions taken in the fast-paced and intensively stressing attack-defense environment, and endured for several days by players of such competitions, have not been assessed or sufficiently analysed. The research carried out in this paper has allowed to identify the strategies used by some of the best hacker teams in the world, who competed against each other in the DEF CON 28 CTF finals, which occurred between 7^{th} and 9^{th} August 2020. The analysis of the teams' tactics has brought to light some of the wrong and the successful decisions made by each team, when gaining them a valuable flag or making them lose control over their server. The dissertation will show how simple techniques as network sniffing can prove successful during a CTF, disclosing adversaries' flags and intents, allowing contestants to steal exploits and patches used for other services; or how stealth communications have been implemented to help participants to avoid having their traffic sniffed, though giving them only half point for successful exploitation or flag submission.

The analysis of the DEF CON CTF has given useful insights, that will allow future players to better prepare for cyber conflicts, as well as future analysts to better navigate the districated landscape of capture the flag competitions and their network analysis. This research attempts to expose the importance of analysing capture the flag events and to be of motivation for future research in this field, eventually easing the analysis process for past and future competitions.

1.2 Research Statement

This dissertation intends to bridge the existing gap between theory and practice in the analysis of capture the flag competitions for the purpose of identifying tactics, techniques and procedures employed during real-time computer conflicts. This paper offers knowledge for enhancing the analysis process of CTF data-sets, extracting meaningful information about the competition's participants. The research will explore the techniques used by capture the flag competitions' participants, through the analysis of available game data-sets, and list recommendations and platforms that are critical for real-time cyber conflicts training. The dissertation project is aimed to analyse and discuss the data sets collected at DEF CON 28, extracting relevant information and examining various aspects of the event, in terms of: type of challenges, system site, and type of participants, as well as techniques, tools, tactics and procedures utilised by contestants. Finally, the research will explore the currently most used hacking and capture the flag tools, leveraged by Attack-Defense CTF players, to quickly create counter-attacks, defend or deceive adversaries. This research aims to be a supporting document to develop the skills and mind-set necessary to successfully tackle future capture the flag competitions.

1.3 Research Aim and Objectives

The research and analysis process were carried out considering the following aims and objectives, established for a more focused and scoped in research.

Aim:

Analysing best practices, tools and tactics, and the correlation between dominant strategies and successful scoring during Capture the Flag competitions.

Objectives:

Analysing competitions and participant's tactics, techniques, tools and procedures used to discover and collect flags.
Using big data and network forensics analysis tools and procedures to parse game data and network traffic gathered during the contest.
Provide recommendations to approach future competitions.

1.4 Challenges

The research has presented several challenges when looking for thorough information in regards to analyzing CTF events from a forensic point of view. Examining payloads and extracting exploits becomes an impossible task without the identifying signatures, as well as extrapolating flags from the packet captures with simple regex strings results in an excessive amount of false positives, as also discussed by Yam in [9], where an analysis on the data extracted from DEF CON 22 has exposed similar difficulties.

- Scarce and outdated non-pedagogical literature about the analysis of capture the flag competition;
- Limited indications on the methodology to apply to analyse large volumes of network capture files;
- Large volumes of unreliable and unfiltered data to process;
- Literature mainly focused on the pedagogical aspects of capture the flag events;

Furthermore, the literature available on academic paper focuses mainly on the educational purposes of the competitions, and while these are widely recognized, researchers in these studies do not delve into the actions taken by the top teams to win the competitions. In fact, a technical analysis as the one presented here is quite a daunting task that cannot be automated and it is still hindered by the limited availability of tools that can treat large amounts of data. These challenges have been overcome with an in-depth study of the documentation of less known network forensics tools and an extended research through articles, blogs, writeups and videos that have shed light on an overlooked area of capture the flag competitions.

1.5 Dissertation Organisation

The dissertation has been divided into five chapters:

- Chapter 2 discusses the background and the literature review that was necessary to carry out the research.
- Chapter 3 describes the methodology used to dive into the data sources.
- Chapter 4 explores the practical procedure that resulted from the implementation of the methodology described in the previous chapter, and how to leverage big data and network forensics analysis tools to find useful information among large volumes of packet-capture files, using *deep packet analysis* techniques and packet-capture file manipulation, which allowed the data and network visualization contained in this document.
- Chapter 5 discusses the findings and the results of the analysis process, describing the successful strategies, tools and procedures as well as suggesting a few recommendations to successfully approach future competitions.

Finally, the Appendix contains a project management section as well as code snippets that were built during the analysis process, which will be helpful to future analysts wanting to dive into enormous amounts of packet capture files. The project management section will show how the project was carried out respecting the deadlines throughout the entire dissertation drafting process.

Chapter 2

Background and Literature Review

2.1 Overview

It is critical to specify and explore the foundational knowledge required by this project by reviewing the literature available and some fundamental concepts, which will allow a more efficient description of the problems encountered, as well as the thinking process applied by participants during the competition. This chapter discusses capture the flags and the type of challenges that can be found during contests, as well as describing game theory, adversarial thinking, network traffic analysis and the necessary background information related to the final competition of the DEF CON 28 capture the flag competition. The section related to the literature review will explore some of the research done in terms of pedagogy, scalability, games analysis, CTF comparisons and technical solutions developed to better analyze capture the flag competitions.

2.2 Capture the Flag Competitions

Capture the flag (CTF) competitions are simulated cyber conflicts joined by several opposing teams, competing to capture the largest amount of tokens (flags) in order to gain fame, distinction and respect among the hackers' community, and sometimes money prizes. The research has already established that such competitions are remarkably valuable in both academic and professional environments, and can be used as a comprehensive cybersecurity teaching tool, capable of delivering knowledge by taking advantage of computer security challenges intended as puzzles, which can be solved with solid cybersecurity foundations and advanced (re)searching skills.

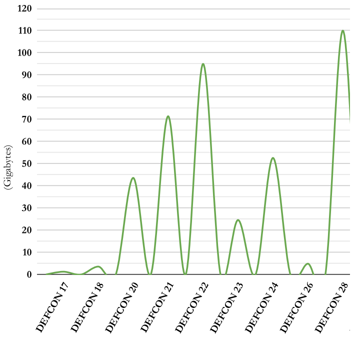


Figure 2.1: The amount of packets captured over the network during DEF CON finals.

CTF competitions at DEF CON have been running since 1996, making it the de-facto largest and most famous capture the flag event on the globe, attracting a massive pool of talented attackers [10]. For example, Figure 2.1 shows how the data exchanged over the DEF CON CTF network has increased throughout its editions, therefore demonstrating that the amount of the packets transmitted has grown, due to several factors: the increased number of players joining in the competition, the evolution of the games' infrastructure and the development of the attack-defense and king of the hill challenges over the years.

Flags are the reward for a successfully completed challenge or a service exploited on the opponent's server, a data file which grants points once submitted to the game system, and is discovered only when a working solution is found to a custom-made cybersecurity related problem. The challenge in the game may revolve around solving puzzles as in the case of *Jeopardy* style competitions, or attacking and defending vulnerable computer infrastructures as in the case of *Attack-Defense* style CTF. The challenges in the form of puzzles expect players to have a skill set that ranges from web hacking to cryptography, to digital forensics, as well as programming and reverse engineering. However, in the case of Attack-Defense events, a broader set of skills and prompt responsiveness to opponent's actions might be necessary. Attack-Defense competitions have proven how live real-time conflicts require a deeper knowledge in regards to offensive and defensive techniques in order to be played successfully: teams can steal adversaries' flags from the wire, copy and reuse exploits, monitor opponents' moves and utilize obfuscation and deceptive techniques to hinder and confuse enemies, while at the same time executing vulnerability assessments, instantly developing patches and deploying them to protect themselves from incoming cyber attacks.

Researchers in [11, 12, 13, 14] have recognised the importance of capture the flag events for future ethical hackers and information security specialists. In fact, such competitions have been defined as extremely pedagogical tools, which can greatly improve students' performance, motivating them to put more effort into their learning [9]. Furthermore, cybersecurity competitions have become an excellent recruiting tool for tech companies [15].

The popularity of CTF events has grown beyond educational institutions, and today we see CTF events organized by cyber security corporations and enthusiasts almost on a weekly basis [9].

Table 2.1: Number of capture the flag events that have been posted on CTFtime.org divided per year, format and location.

	CTF Format		*Location*		
Year	***Jeopardy***	***Attack-Defense***	***Remote***	***On-Site***	***Total***
2012	23	10	19	16	35
2013	41	13	35	20	55
2014	49	8	36	22	58
2015	65	12	48	31	79
2016	90	14	67	40	107
2017	125	14	102	39	141
2018	136	16	102	51	153
2019	175	20	145	53	198
2020	126	13	130	13	143
Total	830 (86%)	120 (12%)	684 (71%)	285 (29%)	969

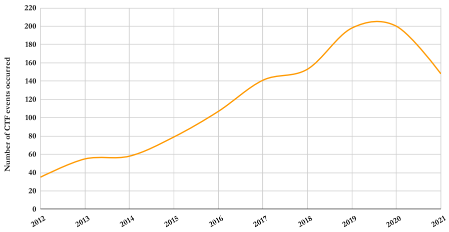


Figure 2.2: Number of CTF Events occurred from January 2012 to October 2021

Nowadays, CTF competitions can be adapted to be suitable also for non-traditional players, regardless of their cybersecurity knowledge, hacking skills or main career focus [16]. Table 2.1, together with Figure 2.2, show the number of Capture the Flag events occurred in the last nine years, divided by challenges' format and location, as per data provided by CTFTime.org [1]. The difficulty in analysing hundreds of gigabytes of raw network captured packets and the limited availability of automated tools aiding in such analysing it, as well as the advanced analytical and network forensics skills required to retrieve useful information, had created a substantial gap in the research of CTF teams' tactics, techniques and procedures (TTP). Nonetheless, a deep analysis into the mechanics and strategies adopted during a real capture the flag competition offers fresh solutions and insights for both, information security researchers and CTF players. The competitions that in 2020 weighted more in terms of reputation are shown in Figure 2.3.

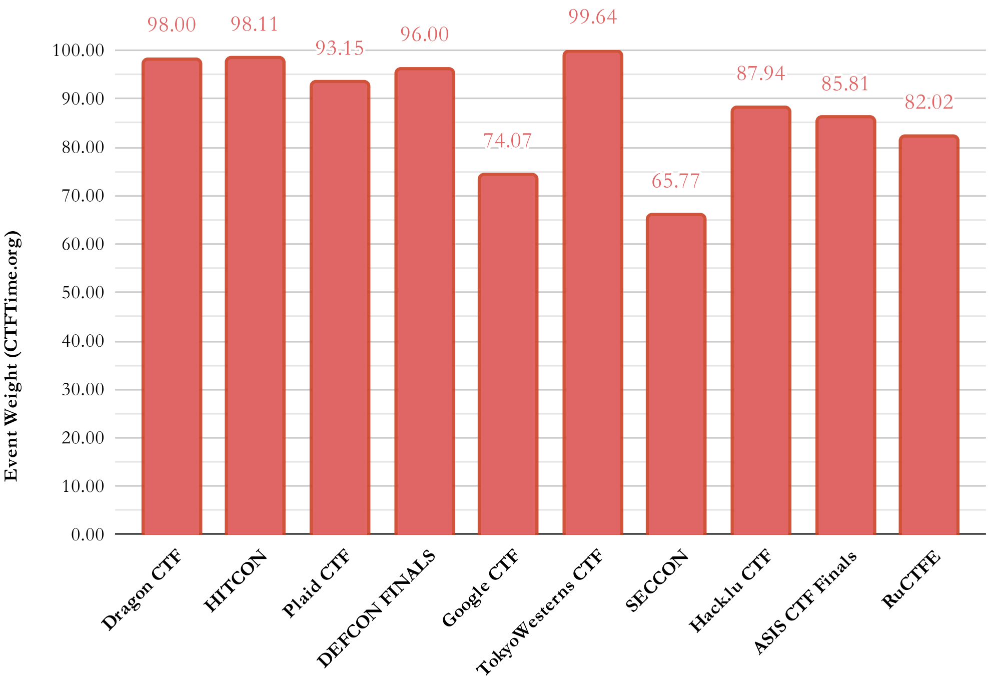


Figure 2.3: Capture The Flag competitions that are considered the most elitist showing a bigger weight according to CTFTime.org [1]

2.2.1 Jeopardy Style

It refers to a series of tailored hacking challenges that have been created to test participants on a variety of practical cybersecurity subjects. The challenges can revolve around different topics, such as: reverse engineering, digital forensics, web, cryptography, programming and many others. The critical property of a jeopardy style CTF is that once solved, the challenge should allow the participant to obtain one or more flags [17]. Jeopardy style CTF are typically hosted on a remote server, though the challenge could also include downloading a file locally that will contain the flag in itself. On the other hand, when the participants find themselves dealing with remote servers, the CTF usually requires an exploit to grant a flag. The more difficult is the challenge supposed to be for the teams, the more points a flag should grant. However, this also depends on the rules established by the event's organisers and all challenges may result in an equal share of points to be scored.
In addition to awarding points, some competitions award prizes to the first solutions of particular challenges; this is often done in conjunction with sponsor-provided challenges, who will provide a prize for the team that is first to solve the challenge. Each team's score is the sum of their awarded scores for each challenge, and the winning team is decided by the highest score at the end of the competition [18].

Following, are described the most common Jeopardy style type of challenges:

Table 2.2: Jeopardy Style CTF Challenge Types

Type	***Description***
pwn	Pwn challenges' goal is to "own" (compromise) a system, a piece of software or a target service to obtain a flag. The target is often vulnerable to a not well known exploitation procedure and the participants have to find the right way to break in and extract the flag. Such challenges are often solved by interacting with a remote server through a command line shell.
reverse engineering	Challenges that involve reverse engineering a binary to extract a flag. The executable is often downloaded from the server hosting the challenge and may require to look for a flag in the source code, or to understand the functionality of the software to cause it to output the flag.
digital forensics	Forensics challenges often entail investigating unknown data, possibly generated in unusual formats. The flag is often steganographically hidden inside the data and understanding the data format is often the first step to tackle the challenge.
web	These challenges are solved by executing web exploitation attacks against a real vulnerable web server. It usually entails connecting to an HTTP server and exploring the server to find vulnerable features. Often, the challenge involves some type of injection, a specific exploit execution or a request/response replay attack.
cryptography	Flags can also be obtained by reverse engineering a cryptographic protocol or a cryptographic system applied to a specific scenario or software.
programming	Increasingly often, CTF competitions include programming challenges, testing teams in their ability to write code to obtain a flag.
miscellaneous	In a competition, challenges that cannot be categorized, fall into the misc group. Typically such competition entail more gamified versions of cybersecurity problems that can be more accessible for beginners or teach foundational skills.

Furthermore, listed below are some advantages and disadvantages that could be related to jeopardy style capture the flag events:

Table 2.3: Advantages and disadvantages of Jeopardy Style CTF

Advantages	***Disadvantages***
Beginner Friendly	Often unrealistic challenges
Task Separation	Absence of adversarial response
Guided Challenges	
Optional participation	Often excessively gamified
Rewarding Experience at all levels	

2.2.2 Attack-Defense

As discussed by Cowan et al. in [10], this type of CTF game is *symmetric* as each team has both attackers and defenders. In attack-defense competitions, such as the ones organized at DEF CON, each team starts by assessing their own server (*jumpbox*), given to them by the CTF organisers. In fact, in a comprehensive study on capture the flag events made by ENISA, it is pointed out how in Attack-Defense competitions ‘teams may be expected to deploy specific patches to vulnerable software, which might range from updating off-the-shelf vulnerable software, through to writing and applying patches directly to custom services. They may also be expected to perform general network-hardening measures, such as updating firewall rules, resetting or strengthening passwords, and disabling unwanted or untrusted services or users’. Furthermore, in addition to hardening their infrastructure, teams have to attack and compromise their adversaries’ machines and vulnerable services. Points are awarded to teams that exploit opponents’ services, taking and maintaining control (persistence) over as many target hosts as possible [17] for a given amount of time (called *round*, or *tick*). The vulnerabilities may be based on real-world scenarios, CVE (*Common Vulnerabilities and Exposure*s) [19] recently discovered, or custom-made, with vulnerabilities purposely created by the organizers to evaluate individual responses and test specific abilities among the teams. Depending on the style of the CTF, flags can be awarded when successfully exploiting an adversary’s service, urging them to develop a patch in a timely fashion to avoid being further compromised, or by maintaining access on as many hosts as possible until the end of the round.

Typically competitions’ rules require services to stay always up to prevent teams from disabling vulnerable services instead of patching them, however some Attack-Defense CTF may involve extra regulations in regards to Service Level Agreement (*SLA*), making teams simply lose points in case their server goes offline. Organizers may also decide to grant extra points to teams successfully stealing flags by sniffing opponents’ traffic.

Often a Virtual Private Network (VPN) architecture is employed to host such events, so as to block Attack-Defense style competitions present the following characteristics:

Table 2.4: Attack-Defense CTF features

Features	***Description***
Symmetric real-time cyber conflict: expects teams to develop exploits and offensive strategies, as well as develop patches and defensive strategies	Players have to prepare to respond to imminent attacks on their servers and attempt to breach their rival's defence mechanisms in a timely fashion.
Each team has their vulnerable machine to defend from the adversaries	Each team is given their own virtual machine, to which they are connected through VPN.
Requires teams to have both, offensive and defensive competences	Each team has members that will dedicate their time in either attacking or defending.
Involves running exploits against vulnerable services	After discovering their opponents' vulnerabilities, teams have to develop exploits in order to reach the flag hidden in their adversaries' system.
Expects teams to keep their vulnerable services running	Teams cannot disable vulnerable services running on their servers. To stop services being exploited, they have to deploy valid patches.
Live Scoreboard	The points awarded to each team and the actual ranking at each round are shown on a real-time updated scoreboard.

In table 2.5 are listed some of the advantages and disadvantages that could be associated to Attack-Defence CTF games.

Table 2.5: Advantages and disadvantages of Attack-Defense CTF

Advantages	***Disadvantages***
Realistic cyber conflicts	Less Gamified CTF
Use of dominant strategies increases the chances of winning	Flags can be stolen by opponents
Introduces obfuscation techniques	Traffic can be sniffed by adversaries
Requires a broader range of cybersecurity skills: offensive and defensive	Not Beginner Friendly
Higher recognition	

The information discussed above demonstrate that Attack-Defense CTF events require participants to have excellent decision making abilities and be able to implement strategies that can make the difference during this type of cyber conflicts. An increasingly common variant often added to Attack-Defense competitions, is the King of the Hill (KoTH). Such types of challenges award points to teams that either defend their system the longest period of time in a round, or for submitting the best solution to a jeopardy style challenge available on the same network as the Attack-Defense game, as it happened in the case of DEF CON 28.

2.2.3 Game Theory, Adversarial Thinking and CIAAAN attributes

The successful development of strategies utilized to win the competitions comes from an in-depth review and use of *cyber-warfare* and *real-time computer conflicts* tactics, techniques and procedures, requiring concepts such as *game theory*, *adversarial thinking*, and *CIAAAN attributes* to be applied at the right time, to ensure that the maximum damage is delivered to the targeted opponent's infrastructure. Game theory is the study of strategies during conflicts and it is critical to understand action and reaction correspondence, or the best reaction for a given action in a conflict. Strategies that are different can interact with one another, making attack and defence capture-the-flag competitions more similar to a real conflict. It is through the study of adversarial and game theory that it will be easier to build strategies that are more efficient, can help detect attacks and develop techniques to efficiently counter the attacks [20]. Hackers competing in capture-the-flag events consistently outperform each other in terms of dominant strategies and, with the aggravating circumstance that the cybersecurity landscape is always evolving, it becomes difficult to establish the best procedure to win computer conflicts. A mind-set that always applies to CTF is Adversarial Thinking, which is the strategic ability to embody the technological capabilities, the unconventional perspectives, and the strategic reasoning necessary to anticipate the strategy actions of adversaries, including where, when, and how they might attack. This helps preserving the building block of information security, also known as the CIAAAN attributes, described in the following table:

Table 2.6: CIAAAN Attributes

Attributes	***Description***
Confidentiality	Ability to keep communications confidential
Integrity	Ability to ensure that the information has not been tampered with
Availability	Element that suggests the possibility of accessing resources or information
Authentication	Defines how to prove for the identity of the person accessing the information or resource
Authorization	Defines who can access the information or resource and what they can do with it
Non-repudiation	Ability to verify that an event has occurred

Each of these attributes are fundamental during attack and defence capture the flag competitions, and should never be overlooked during computer conflicts. It has been demonstrated that by removing CIAAAN attributes from the opponent (*information-based conflict*), it is possible to gain the opportunity to manipulate or expel them from the environment [21]. In fact, such attributes can help establish dominant strategies that naturally best the opponents [22] in CTF. The best response a participant could make in a given situation is studied in the analytic discipline called Game Theory, which can be applied every time players are required to make the best decisions among other competing players in order to win a conflict, or a CTF game. If all opponents respond with the best solution to an opponents' action, there would be a situation of perfect *Nash Equilibrium.*

2.3 Network Traffic Analysis

Network Traffic Analysis is the process of capturing and interpreting data that has flown across a network to collect information, establish root-cause analysis of a network event, analyse malware behaviour, monitoring networks' connectivity, or detect malicious attempts at breaching security mechanisms. Also, packet analysis can help with the following:

- Understanding network characteristics;
- Learning the systems active on a network;
- Determining network bandwidth users;
- Identifying peak network usage times;
- Detecting malicious activity;
- Help reconstruct payloads, files and sessions established over TCP;

The goal is to analyze the inbound and outbound network traffic in the system under observation. After the competitions, analysts can leverage network forensics to analyze the large volume of traffic exchanged, which is usually stored in PCAP format, readable by most of the network traffic analysis applications available, but often the data is partitioned into smaller capture files that are difficult to handle. Fortunately, it is possible to leverage scripts and tools to filter and sort out the raw data, making the analysis process smoother, reliable and highly rewarding when carried out using network traffic analysis and deep packet inspection techniques, which will allow the retrieval of useful information, aiding in the data correlation, identifying traffic anomalies, aggregating sources, destination IP addresses and other network information such as communicating port. Furthermore, it will be possible to enumerate services, search for specific strings inside the packet payloads and detect the use of exploits, through the use of the network traffic analysis tools. Furthermore, the teams playing in the CTF events also take advantage of network traffic analysis tools and techniques to steal flags or re-use exploits captured from their opponents, for example tools like tcpdump and Wireshark are the most used for this purpose.

NetFlow and IPFIX

NetFlow is a network traffic summarization standard developed by Cisco Systems, and it was initially employed to monitor the network bandwidth of users to bill them accordingly [23]. The use of this standard for network traffic analysis makes the accessing of high-value traffic information in a more compact and manageable format. NetFlow works with the concept of *flow* [23], which is an approximate reconstruction of a TCP session, assembled by comparing sequence numbers. In fact, a flow is a collection of packets that are closely grouped in time and are identically addressed. However, such standard would work mainly with Cisco developed solutions, therefore the IP Flow Information eXport (IPFIX) protocol was designed in an open way that easily allows using it with other protocols, interfaces, and applications [24]. The IPFIX format has been further extended with the development of the SiLK toolkit, in which the CERT Network Situational Awareness Group at Carnegie Mellon University developed additional fields aimed at improving information security analysis [23].

2.4 Literature Review

As it is evident from the literature review that follows, capture the flag competitions have become a huge topic, as they have the potential to become the next generation cyber-range, for hackers of all statures to leverage to understand and adopt cyberwarfare and real-time computer conflict tactics and strategies to prepare to future cyberattacks. Capture the flag competitions are still in a development phase, and while state-of-the-art competitions are being held every year, students and researchers are theorising the CTF events of the future.

Karagiannis and Magkos, in their paper from May 2021[25], highlighted how the use of capture the flag challenges as part of an engaging cybersecurity learning experience has made students feeling more confident about their skills, improving their engagement in the learning process, and showing positive outcomes in terms of technical skills and knowledge acquired

Goodman and Radu [26], have extensively documented the benefits of hackathons and capture the flag competitions, in a study made in 2020 in regards to the pedagogic theory that underpins them, utilizing a Learn-Apply- Reinforce/Share framework of learning. With the shift to a remote setting for work and study, these events are becoming increasingly prevalent allowing students to meet new people, feel welcome in a learning environment that is educational and enjoyable. The research also highlights the strong sense of community demonstrated in prompt support when needed, given from attendees as well as organizers, and the importance of feedback throughout the event's progression. Distance Learning recommendations and best practices were also taken into consideration by this research, which recognized how capture the flag competitions are more easily hosted and organized online, than hackathons.

In 2020, Karagiannis et al. [27] has explored and compared four popular open-source CTF platforms, which are *Facebook CTF*, *CTFd*, *Mellivora* and *Root the Box*, highlighting the distinct features for each, describing their advantages and disadvantages, and suggesting extra features to add to such platforms to improve them. Participants to the survey organized by the researchers have stated the main components that would make capture the flag events more appealing: *graphics and visualization*, *progress tracking*, *live score tracking*, an improved *rewarding system*, *storytelling elements*, *structured challenges* and *educational* *appropriateness.*

A report issued by the European Union Agency for Cybersecurity (ENISA) in 2021 [17] has addressed the contemporary use of capture the flag competitions around the world, providing a background about competitions' structure and variations, with an in-depth qualitative analysis of 22 notable CTF events, as well as a high-level statistical analysis of a 879 public events of all levels [17]. The analysis made by ENISA showed the variations in areas such as team size limits, challenge categories, scoring process, type of hosting platform, use of qualifier rounds and communication channels for media strategy. The competitions were organised mostly by governments, community groups and universities and were intended either for the general public or for students in tertiarty education. Furthermore, the study has highlighted the increasing presence of Attack-Defense formats in CTF events and the consistent grow of number of competitions all over the world.

Kucek and Leitner [28], in 2020, investigated the underlying infrastructure and CTF environments of 28 CTF infrastructures, focusing on 8 specific capture the flag competitions that were hosted on an open-source infrastructure, therefore making easier to inspect the CTF's code. The competitions studied by Kucek and Leitner are PicoCTF, FacebookCTF, HackTheArch, WrathCTF, Pedagogic-CTF, TootTheBox, CTFd and Mellivora. Their CTF challenges were studied to include several parameters such as: supported type of challenges, presence of hints, points awarded, challenge description and other information, such as participant's registration, scoring, registration details, number of players and so on. The study recognizes the value of open-source CTF environments, however it defers to the organizers the decision for the best type of infrastructure to host the competition.

Trickel et al. have studied in 2017 the importance of leveraging capture the flag competitions to tackle the shortage of cybersecurity professionals, recognizing the educational benefits of Attack-Defense events and building a framework to quickly configure an Attack-Defense CTF game on cloud, calling g it CTF-as-a-Service [29].

In 2017, Taylor et al. [30], analysed the competitions led by the Cyber Defender group over 8 years and described a novel framework that was designed according to the observations. Such a framework, called Catalyst, allows to run competitions in a more cost effective, extensible and flexible way. The solution can lead to improved CTF events, through better challenges and better data collection. The state-of-the-art CTF competitions assessed by Taylor et al. are mentioned in the table below, which lists 36 CTF implementations with information regarding whether the given engine supports static or dynamic content, whether the content supported contains policy-based problems, and whether the challenge engine and content is open-source, and is just a sample extracted from [30]:

Competition Name	***Static***	***Dynamic***	***Policy***	***Open Source***
DEF CON CTF Finals	No	Yes	No	No
RuCTF	No	Yes	No	No
UCSB iCTF	No	Yes	No	Yes
RuCTFE	No	Yes	No	No
DEF CON CTF Qualifiers	Yes	No	No	No
OpenCTF	Yes	No	No	No
CCDC	No	Yes	Yes	No
Panoply	No	Yes	No	No
PlaidCTF	Yes	No	No	No
PicoCTF	Yes	No	No	Yes
BackdoorCTF	Yes	No	No	No
Ghost in the Shellcode	Yes	No	No	No

Furthermore, Bashir et al. in 2017 has studied the personality, interests, culture, decision making and attachment styles of 588 participants to the Cybersecurity Awareness Week (CSAW), examining subgroups such as self-proclaimed hackers and non-hackers, males and females and cybersecurity employees and students [31]. The research has revealed that participants who displayed self-efficacy, rational decision-making style, and more investigative interests were more likely to declare an interest in a career in cybersecurity after the competition, suggesting to CTF events' organizers to target this type of demographic when attempting to attract new participants [31]. Furthermore, this research highlights how hackers attendees of various hacker conferences between 2005 and 2007, hacked for non-malicious reasons, with 31% reporting that they hack to solve interesting puzzles and challenges, and 22% reporting that they hack to advance network, software, and computer capabilities. In another survey, made to 216 hackers at a security convention on creativity, depression, and lifestyles, it was found that they were generally highly creative and had excellent stress management and multi-tasking skills (Bashir et al. 2017).

An in-depth analysis of tactics and strategies leveraged during capture the flag events was extensively done by Yam [9], who has initially collected data sets from the DEF CON 22 CTF Finals, and forensically analysed them to identify strategies and exploits used by the players. The research has revealed many of the tactics used during DEF CON 22, and discoveries include:

- The use of automated attacks to exploit other teams resources at regular intervals;
- The correlation between the number of flags discovered and the average time to develop an exploit, making the teams that developed an exploit faster the ones to accrue the most points;
- An approach purely was more effective than a more attack-defense balanced approach

However, such discoveries apply mainly to the DEF CON 22 CTF game structure and scoring rules, and are unique for that competition. Nonetheless, the study gives useful information to apply a successful methodology to study the data sets of capture the flag events.

The research carried out by Burns et al. [32] collected and analyzed the solutions of about 3600 Capture The Flag (CTF) challenges from 160 security competitions between 2012 and 2015, enumerating the security tools and techniques used by players. The CTF competitions evaluated were PicoCTF, OpenCTF, CTFd, FacebookCTF, TinyCTF, assessing jeopardy style challenges such as crypto, web, reverse, forensic, pwn and misc. Furthermore, the researchers have hosted their own CTF based on the open source PicoCTF infrastructure, discovering that challenges are beginner friendly and the majority of participants felt like their computer security concepts understanding was highly improved [32].

In 2016, Raj et al. [33], has discussed the implementation of containerization and container orchestration technologies such as Docker and Docker Swarm, to host more scalable and performant Attack-Defense CTF competitions, easing the process of setting up an efficient and cost effective CTF infrastructure that uses Docker instead of virtual machines. However, the solution proposed does not allow teams to capture exploits off the network and debug them to discover new vulnerabilities to patch on their server.

Nunes et al. [34] in 2015 has studied various classification techniques to attribute a cyberattack to its culprit, by examining the DEF CON 21 CTF data, discovering the deceptive techniques that accounted for the majority of misclassified samples, as well as exploring heuristics to mitigate the misclassification caused by deception. Nunes et al. used various machine learning approaches to parse the dataset, suggesting to use the same approach to identify advanced persistent threats and hacking groups in a network. The work was further improved in 2016 by building an advanced argumentation-based framework called Defeasible Logic Programming (DeLP) that could be employed to derive arguments as to who could have conducted a cyber-attack [35].

Burket et al. (2015), has studied a solution to automatically generate challenges for CTF competitions. Automatic problem generation (APG) allows a CTF challenge to create multiple automatically generated problem instances to allow an easier detection of stolen flags, similar problems with transferrable solutions and an improved user experience [36]. While generating problems automatically is not a new idea, APG has yet to be widely adopted in capture-the-flag competitions, and the studies carried out by Burket et al., confirm that APG has great potential to improve both, the effectiveness and integrity of computer security competitions, however the challenges related to the implementation of this technology into a live CTF event are yet to be explored [36].

2.5 The DEF CON 28 CTF Finals

DEF CON is an annual cyber security conference held in Las Vegas, NV. According to its official website, DEF CON was started in 1993 as a party for hacking enthusiasts, all of whom were part of an electronic bulletin board service network [12]. In 1996, DEF CON began holding formal annual CTF competitions, though prior to this, CTFs were also held, albeit in a less formal capacity. Since then, the DEF CON CTF has grown to be the most prestigious of all CTF events, and dubbed by CNBC as the “World Series of hacking” [13].

The DEF CON 28 CTF event was the 19th CTF event to be held at DEF CON. It was a three day event held August 7-9, 2020. The DEF CON CTF events throughout the years have all been attack-defense type CTFs. While this has not changed since its inception, the organizers, the scoring rules, the types of services, operating systems, and architectures have varied annually. The number of vulnerable services for each iteration of DEF CON CTF can range from five to about twenty teams. According to DEF CON CTF history by Dark Tangent [14], the services can range from "poorly configured crypto, SQL-injection, cross-site-scripting, buffer overflows, timing attacks, heap exploits, malformed network constructs and custom interpreters.” The organizers of this event (The Order of the Overflow) prepared two types of services for the competition: Attack/Defense and King of the Hill. The former format is familiar from decades of DEF CON CTF: exploiting other teams’ services to steal their flags, while protecting their own. King of the Hill works by making teams compete for the best solution, depending on the challenge in question. The figure below shows what was the competition each team participated in to compete at DEF CON 28 finals.

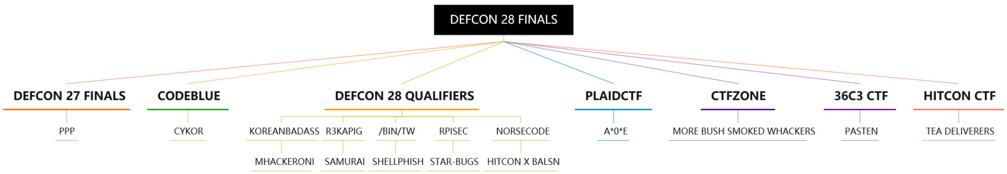


Figure 2.4: Competitions that allowed teams to participate to DEF CON 28 finals

For this competition, the organizers have decided to release the network traffic files for the teams at the end of each round, so that the teams could focus more on the competition rather than sniffing the network. Furthermore, they introduced *stealth ports* which is the concept of using a stealth communication channel to exploit services on target systems, in order to avoid opponents to sniff flags over the network or the exploit to reuse. The event was divided in 4 shifts of 8 hours, with 9 hours between shifts for the teams to rest.

2.5.1 System Site

The operating systems are usually Linux-based (with FreeBSD being used on some occasions) and the architectures vary widely between x86, x64, ARM, and embedded systems. The Order of the Overflow - organizers of the event - have built a customized server rack hosting the CTF game, to which teams had to connect through Virtual Private Network (VPN) built using WireGuard [37]. There was not much information available related to the custom server, willingly undisclosed by organizers to avoid risking to be attacked by participants, however the server hosting the CTF infrastructure was presenting the following features:

Table 2.7: CTF System site configuration

On-premise custom server rack
CTF Network accessible through VPN connection
Linux Operating Systems to power the jumpboxes
Network has been built with normal traffic and obfuscated traffic

The figure below shows an approximation of the type of network infrastructure that was deployed at DEF CON 28, however it is important to note that each team was connecting from their home location, as the competition was held remotely. To avoid cluttering the figure with too many images, only four teams were represented.

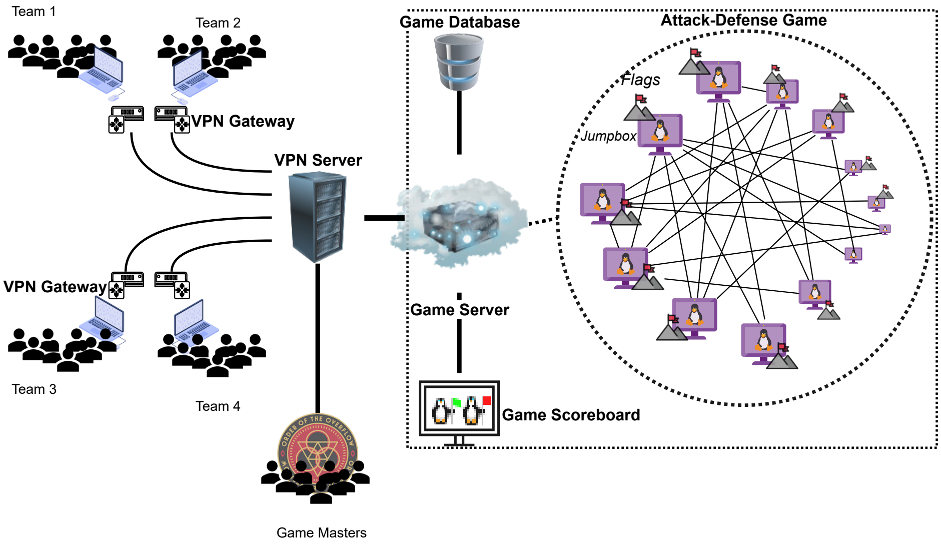


Figure 2.5: An approximate representation of the CTF infrastructure.

2.5.2 Team IP Assignments

Each team was assigned a server to defend from attacks from other participants, called jumpbox. Each team's jumpbox presented vulnerable services that would be activating and disabling after a number of rounds, called ticks. This system prevented teams from stealing an excessive amount of flags or exploits off the network, limiting the number of times that exploit or flag could be reused. In the following table there is an overview of each team's jumpbox network CIDR subnet and IP address.

Table 2.8: Teams and their IP Addresses

Teams	Assigned CIDR Subnet	Jumpbox Address
A*0*E	10.1.0.0/24	10.13.37.1
/bin/tw	10.2.0.0/24	10.13.37.2
CyKor	10.3.0.0/24	10.13.37.3
HITCON ⚔ Balsn	10.4.0.0/24	10.13.37.4
koreanbadass	10.5.0.0/24	10.13.37.5
mhackeroni	10.6.0.0/24	10.13.37.6
More Bush Smoked Whackers	10.7.0.0/24	10.13.37.7
NorseCode	10.8.0.0/24	10.13.37.8
pasten	10.9.0.0/24	10.13.37.9
PPP	10.10.0.0/24	10.13.37.10
r3kapig	10.11.0.0/24	10.13.37.11
RPISEC	10.12.0.0/24	10.13.37.12
Samurai侍	10.13.0.0/24	10.13.37.13
Shellphish	10.14.0.0/24	10.13.37.14
Star-Bugs	10.15.0.0/24	10.13.37.15
Tea Deliverers	10.16.0.0/24	10.13.37.16

2.5.3 Scoring

To score points during the competition the teams had to steal flags from other teams' attack-defense services (attack points), resist attacks against their infrastructure (defence points), and submit the best solutions for the King of the Hill challenges. The competition did not include 'SLA' or 'uptime' points to be detracted in case of vulnerable services not running on the team's system. Using the stealth ports to exploit a service and gain a flag would grant only half point to the team. This was a rule to make teams decide strategies about when it is appropriate to use stealth ports. Furthermore, successful exploits or successful defence of a service would grant 1 point. King of the Hill points depended on the quality of the solution, which was assessed by the organizers during the event. Each round, all teams tied for first place got 10 points. Rules were strict in regards to forbid DDOS against other teams and the creation of patches that adversarially passes the pre-deployment tests but brings down their team service. At the end of the competition, the teams with the greatest amount of points are the ones higher ranked in the scoreboard. An overview of the final scoreboard status is shown in the table below:

Table 2.9: Final Ranking at the end of DEF CON 28 CTF

Rank	Teams	Points
1	A*0*E	970
2	HITCON ⚔ Balsn	968
3	Tea Deliverers	841
4	More Bush Smoked Whackers	750
5	Samurai侍	635
6	Shellphish	570
7	CyKor	495
8	/bin/tw	435
9	NorseCode	409
10	Star-Bugs	394
11	PPP	352
12	koreanbadass	303
13	mhackeroni	273
14	r3kapig	260
15	RPISEC	211
16	pasten	77

2.6 Summary

This chapter has discussed the background of the dissertation and the literature review, describing capture the flag modalities, fundamental concepts necessary for developing strategies, the need of network traffic analysis for the in-depth study of the DEF CON 28 CTF competition as well as the literature review that was part of the extensive investigation done in regards to capture the flag events. Furthermore, the DEF CON Safe Mode event has been described in this chapter, with an overview of rules, restrictions, limitations, challenges, system site and scoring system, to give a better picture of the competition and what the participants have experienced during the three days of CTF. In the following chapter, the methodology of choice applied for the analysis of the DEF CON 28 capture the flag finals dataset will be discussed.

Chapter 3

Methodology

3.1 Overview

The methodology developed in this research has been drawn by different resources available online. As explained in the previous chapter, Yam [9] has efficiently performed the analysis by using a similar methodology, which motivated the procedure that was followed to analyse the data sets in this document, as shown in Figure 3.1. Another important research, that inspired this project's methodology, is the one carried out by Nunes et al. [34] to discover the strategies employed by the competition's participating teams, however [34] leverage machine learning to predict attacks and detect deception, as mentioned in the previous chapter.

The methodology that was established to perform the analysis of the DEF CON 28 CTF finals data set, consisted of six fundamental steps that allowed a cost effective, scalable and reliable analysis of the competition, regardless of the size of the data set or the amount of corrupted data contained in it. The steps taken were as follows: *data collection, data extraction, data analysis, environment setup, correlation and interpretation of data* and *conclusion drawing.*

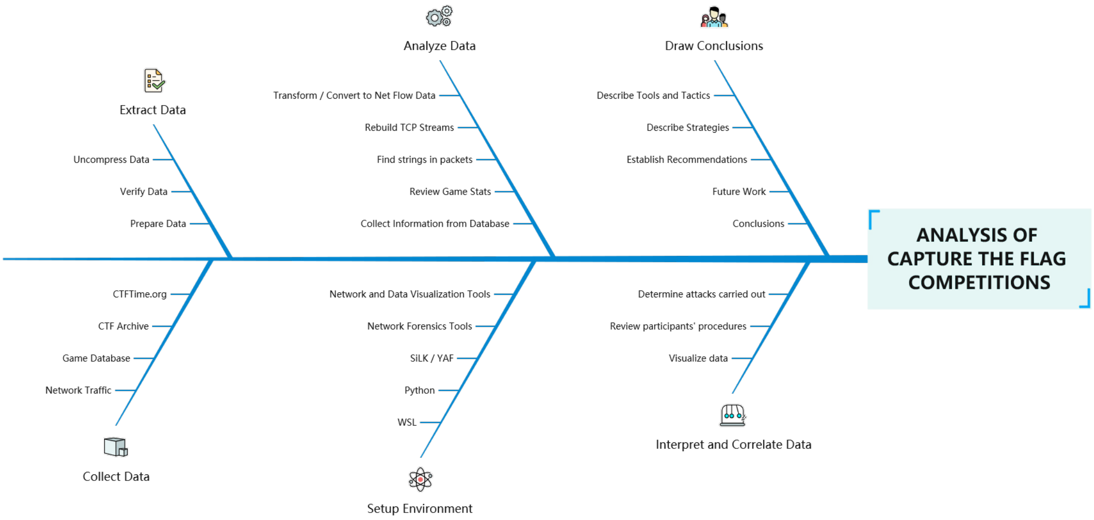


Figure 3.1: Methodology overview

3.2 Collecting Data

The fist step of the analysis is the collection of the data sets from the web. The data consists in network traffic, game database, competitions archive and relevant information collected from other websites that discuss the topic and the competition assessed. The data collection step has to be thorough, as all the data has to be stored locally to be subsequently processed and analysed.

3.3 Pre-processing Data

Packets captured off the network during the competitions, are raw unfiltered data, that includes background noise from network scans and a multitude of failed attempts at exploiting or patching services. Therefore, it is critical to transform the data into a format that can be used for analysis. The network capture files will be processed to generate a streamlined version of the packets, removing payloads from the packets and making the packets containing only network data such as: *bytes exchanged*, *timestamp*, *protocol*, *source* and *destination* addresses and ports. Once the data has gone through the pre-processing phase, it will be easier to analyse it with network forensics tools and create graphs and charts through automated mechanisms that can read through the processed information contained in the network traffic.

3.4 Analysing Data

The data analysis part of the methodology is one of the most critical, given that missed data may lead to missed information. The analysis takes advantage of several tools to parse through the data and output interesting information that could then be fed into a visualization tool or become an important detail to correlate to game state data.

3.5 Interpreting and Correlating Data

An increased aid to this analysis was the possibility to visualise data and filter out unnecessary information, therefore narrowing the search and making sense of the millions of packets captured during the competition. Often visualizing information allows one to come up with insights that would otherwise be impossible. Especially when it comes to searching for anomalies, visualizing data may be a helpful technique. For example, it is possible to check whether teams automate their attacks by visualizing network traffic trends, and spot possible instances of automation from the presence of regular spikes in network activity. In addition, it is important to know whether teams attempted to exploit the clients of other teams. This could be determined by observing a network graph that captured host-to-host conversations and looking for instances where clients of one team connected to clients of another team [9].

3.6 Drawing Conclusions

This is the part of the analysis where all the information that was successfully collected, processed, interpreted and correlated does finally find a meaning. The discoveries made throughout the analysis process are further explained at this stage, and they will help form the recommendations suggested in the final part of this document.

3.7 Summary

An overview of the methodology that has been followed to parse through the hundreds of gigabytes of network and game data, has been given in this chapter. Subsequently, the design and implementation of this methodology is discussed.

Chapter 4

Design and Implementation

4.1 Overview

The analysis of DEF CON 28 CTF data has to be treated differently from typical network analysis, as the data presents two critical differences: the first being that the traffic generated during the event is entirely malicious traffic, the second is that it is not possible to detect anomalous traffic, as most of the traffic generated by the teams presents adversarial behaviour. Furthermore, the vulnerable services available during the competition were custom-made, together with the exploits used to collect the flags, making it impossible to identify a type of exploit through its public signature [1]. Therefore, it was important to establish a different set of considerations to analyse the DEF CON 28 CTF network data.

4.2 Planning the analysis process

The DEF CON 28 Safe Mode CTF was the event of choice because of the availability and reliability of the information and data stored online. The data sources that were collected and used for the analysis are the following:

- Network Traffic of the DEF CON 28 CTF competition;
- Game data from the DEF CON 28 CTF database dump;
- Competition's information gathered on unrelated websites;

However, the massive size of the data collected during the competition required higher processing power and advanced network forensics techniques. The following methodology was adopted to analyse the large amount of DEF CON 28 CTF packet capture files available:

1. Convert the data to a streamlined version to allow a more efficient statistical analysis of the network traffic;
2. Feed the data to visualization engines;
3. Narrow down areas of interest based on timestamps, exploits and flags collected by teams;

As well as, implementing the use of the following technologies and procedures to extract relevant information from the data gathered:

Table 4.1: Technologies and procedures used to extract relevant information

Raw data sources have to be processed and packets stripped from payloads: to facilitate the analysis it is necessary to remove the actual content of the packets exchanged in the network.
Application of tools necessary to gather statistics and visualize relevant network information.
Manual procedure necessary to correlate data between Game Stats and Network Data, and interpret teams' behaviour (their strategies and techniques).
Workstation capable of processing large amounts of network data: laptop running Windows 10, x64 architecture with 16 GB Ram and 3 TB of Hard Drive space available.

The network traffic that presents a large volume will be converted to network flow records, a lightweight version of the combined network packets, however containing less information. Once the analysis process on the lighter data set is completed, it will possible to further down the analysis with the information gathered previously, preventing the overload of limited computing resources but still performing a comprehensive analysis.

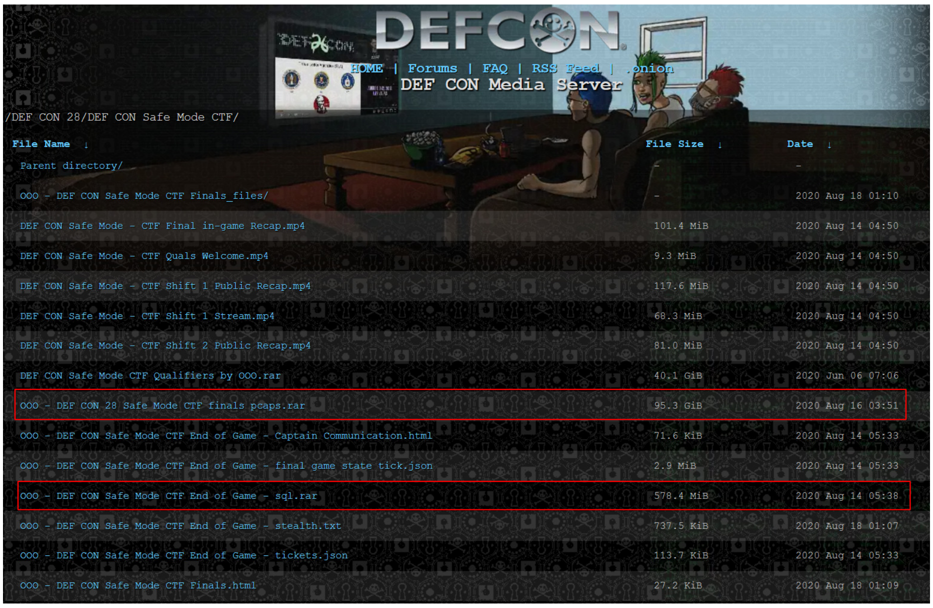
Deep packet analysis is critical to understand tools and tactics leveraged by the teams and, in order to be carried out, a mix of tools had to be used. Deep packet inspection allows then to singularly pick the interesting packets and deepen the analysis by extracting the payload contained inside the packet. The payload may reveal a particular string of ASCII characters, namely *a flag*, or a fully compiled exploit written in C, or the netcat command used to capture a reverse shell. Therefore, by taking advantage of the open-source projects available on the web, the Network Traffic Analysis was carried out with the use of the following tools:

Table 4.2: Tools used for analysis

Network Analysis Tools	***Description***
SiLK	SiLK, the System for Internet-Level Knowledge, is a collection of traffic analysis tools developed by the CERT Network Situational Awareness Team (CERT NetSA) to facilitate security analysis of large networks. The SiLK tool suite supports the efficient collection, storage, and analysis of network flow data, enabling network security analysts to rapidly query large historical traffic data sets. SiLK is ideally suited for analysing traffic on the backbone or border of a large, distributed enterprise or mid-sized ISP
Tcpflow	Tcpflow is a program that captures data transmitted as part of TCP connections (flows), and stores the data in a way that is convenient for protocol analysis and debugging. Each TCP flow is stored in its own file. Thus, the typical TCP flow will be stored in two files, one for each direction. tcpflow can also process stored 'tcpdump' packet flows.
Ngrep	Ngrep is similar to tcpdump, but it has the ability to look for a regular expression in the payload of the packet, and show the matching packets on a screen or console. It allows users to see all unencrypted traffic being passed over the network, by putting the network interface into promiscuous mode.
Mergecap	Mergecap is a program that combines multiple saved capture files into a single output file.
Tshark	TShark is a network protocol analyzer. It lets you capture packet data from a live network, or read packets from a previously saved capture file, either printing a decoded form of those packets to the standard output or writing the packets to a file.
Brassfork	Brassfork helps in visualizing network traffic. It does that by reading PCAP files and outputting files that can be read by graphing applications.
Flowplotter	Generates visualizations from the output of flow tools such as SiLK.
Gephi	Gephi is an open-source network analysis and visualization software package written in Java.
Python Pandas	Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.

4.3 Data Collection

The data sets collected for analysis have been downloaded from the DEF CON Archive [38] and consisted of the game database and the game network traffic.



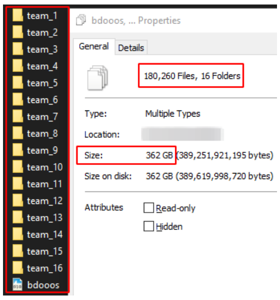
DEF CON Media Server

/DEF CON 28/DEF CON Safe Mode CTF/

File Name ↓	File Size ↓	Date ↓
Parent directory/	-	-
OOO - DEF CON Safe Mode CTF Finals_files/	-	2020 Aug 18 01:10
DEF CON Safe Mode - CTF Final in-game Recap.mp4	101.4 MiB	2020 Aug 14 04:50
DEF CON Safe Mode - CTF Quals Welcome.mp4	9.3 MiB	2020 Aug 14 04:50
DEF CON Safe Mode - CTF Shift 1 Public Recap.mp4	117.6 MiB	2020 Aug 14 04:50
DEF CON Safe Mode - CTF Shift 1 Stream.mp4	68.3 MiB	2020 Aug 14 04:50
DEF CON Safe Mode - CTF Shift 2 Public Recap.mp4	81.0 MiB	2020 Aug 14 04:50
DEF CON Safe Mode CTF Qualifiers by OOO.rar	40.1 GiB	2020 Jun 06 07:06
OOO - DEF CON 28 Safe Mode CTF finals pcaps.rar	95.3 GiB	2020 Aug 16 03:51
OOO - DEF CON Safe Mode CTF End of Game - Captain Communication.html	71.6 KiB	2020 Aug 14 05:33
OOO - DEF CON Safe Mode CTF End of Game - final game state tick.json	2.9 MiB	2020 Aug 14 05:33
OOO - DEF CON Safe Mode CTF End of Game - sql.rar	578.4 MiB	2020 Aug 14 05:38
OOO - DEF CON Safe Mode CTF End of Game - stealth.txt	737.5 KiB	2020 Aug 18 01:07
OOO - DEF CON Safe Mode CTF End of Game - tickets.json	113.7 KiB	2020 Aug 14 05:33
OOO - DEF CON Safe Mode CTF Finals.html	27.2 KiB	2020 Aug 18 01:09

Figure 4.1: The DEF CON archive containing all past CTF data sets.

The data presented is a set made of 180,260 packet capture files, with a size of 362 GB on disk once decompressed on the analysis system. Furthermore, each folder contained the average of 11,000 PCAP files, that would have been impossible to sort through manually - or with any other open-source traffic analyser software available (i.e. Wireshark [39], Tshark [40], CapLoader [41], Bruteshark [42]). The data available in regards to the DEF CON 28 CTF Finals are full packet capture files, containing packet headers, full encapsulated TCP messages together with their payload. Traffic captures as such are extremely large and could contain traffic that is not directly related to the analysis that is being carried out. For example, the data capture included traffic accidentally routed into the CTF network from the internet, multicast packets, broken TCP sessions and background noise from other unidentified sources. Therefore, it was necessary to filter out all the noise and convert the packets to a more streamlined version, resulting in a more efficient representation of the traffic, as well as maintaining



180,260 Files, 16 Folders

Type: Multiple Types
Location: [redacted]
Size: 362 GB (389,251,921,195 bytes)
Size on disk: 362 GB (389,619,998,720 bytes)
Attributes ☐ Read-only
☐ Hidden

Figure 4.2: Number of Files and Volume of the traffic generated at DEF CON 28 CTF Finals

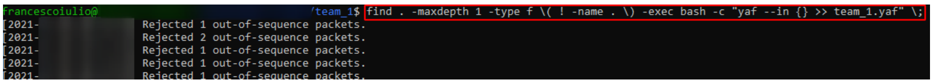
the reliability of the information contained inside the data sets.

To tackle this problem, the entire network traffic was processed with a tool called YAF (Yet Another Flowmeter) [43] which parses PCAP dump-files as generated during the CTF competition, exporting them to the IPFIX (IP Flow Information Export) [44, 45] file format, which is a universal format to work with flow records.

4.4 Pre-processing

YAF is a reliable tool to convert PCAP files to IPFIX flow records. The steps taken to convert the data sets are shown in the following Figure (4.4,4.4). In order to allow the command to be executed iterating through all the files contained in the folder, a short bash script had to be used. The command that was executed to include multiple PCAP files contained in a single team folder, is the following:

```
find . -maxdepth 1 -type f \( ! -name . \) -exec bash -c "yaf --in {} >> team_1.yaf" \;
```

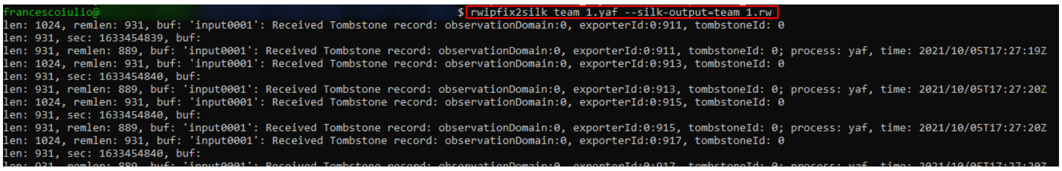


```
francescoiulio@[redacted]/team_1$ find . -maxdepth 1 -type f \( ! -name . \) -exec bash -c "yaf --in {} >> team_1.yaf" \;
[2021-[redacted]  Rejected 1 out-of-sequence packets.
[2021-            Rejected 2 out-of-sequence packets.
[2021-            Rejected 1 out-of-sequence packets.
[2021-            Rejected 1 out-of-sequence packets.
[2021-            Rejected 1 out-of-sequence packets.
```

Figure 4.3: The command to convert the PCAP files in a folder to one single IPFIX flow record.

This command was repeated for every folder inside the network traffic dump gathered from the DEF CON archive, resulting in 16 different flow records (1 for each team). Converting the data to an IPFIX format first, allows to clean the PCAP files and filter them out from corrupted packets, broken TCP flows and discard the packet's payloads. The remaining data is the information about: source and destination IP addresses and ports, protocols, identified network application (based on customizable signatures), communication timestamps, packet transfer's start date and end date and other network traffic information. Subsequently, the records have to be converted to a format readable by the SiLK tool-suite so as to ease the analysis of the flow record files. The command used to convert a single IPFIX flow record to a SiLK flow record is the following:

```
rwipfix2silk flow_record.yaf --silk-output=flow_record.rw
```

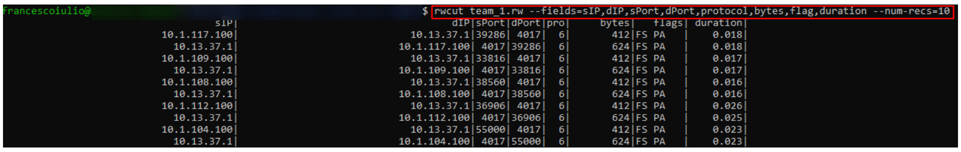


```
francescoiulio@[redacted]$ rwipfix2silk team 1.yaf --silk-output=team 1.rw
len: 1024, remlen: 931, buf: 'input0001': Received Tombstone record: observationDomain:0, exporterId:0:911, tombstoneId: 0
len: 931, sec: 1633454839, buf:
len: 931, remlen: 889, buf: 'input0001': Received Tombstone record: observationDomain:0, exporterId:0:911, tombstoneId: 0; process: yaf, time: 2021/10/05T17:27:19Z
len: 1024, remlen: 931, buf: 'input0001': Received Tombstone record: observationDomain:0, exporterId:0:913, tombstoneId: 0
len: 931, sec: 1633454840, buf:
len: 931, remlen: 889, buf: 'input0001': Received Tombstone record: observationDomain:0, exporterId:0:913, tombstoneId: 0; process: yaf, time: 2021/10/05T17:27:20Z
len: 1024, remlen: 931, buf: 'input0001': Received Tombstone record: observationDomain:0, exporterId:0:915, tombstoneId: 0
len: 931, sec: 1633454840, buf:
len: 931, remlen: 889, buf: 'input0001': Received Tombstone record: observationDomain:0, exporterId:0:915, tombstoneId: 0; process: yaf, time: 2021/10/05T17:27:20Z
len: 1024, remlen: 931, buf: 'input0001': Received Tombstone record: observationDomain:0, exporterId:0:917, tombstoneId: 0
len: 931, sec: 1633454840, buf:
len: 931, remlen: 889, buf: 'input0001': Received Tombstone record: observationDomain:0, exporterId:0:917, tombstoneId: 0; process: yaf, time: 2021/10/05T17:27:20Z
```

Figure 4.4: Conversion from IPFIX to SiLK flow records using the rwipfix2silk command.

Once the command is executed for each IPFIX file, it will be possible to leverage SiLK commands to look for interesting data. For example, the SiLK command *rwcut* shows the first ten packets in the flow record.

```
rwcut team_1.rw --fields=sIP,dIP,sPort,dPort,protocol,bytes,flag,duration --num-recs=10
```

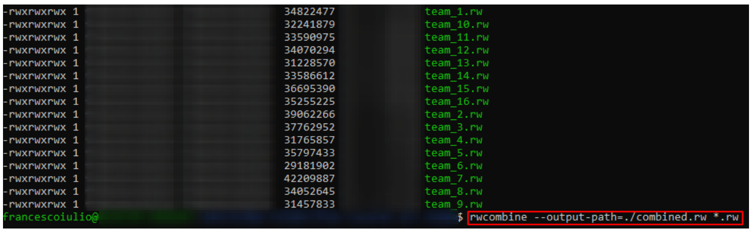


```
francescoiulio@[redacted] $ rwcut team_1.rw --fields=sIP,dIP,sPort,dPort,protocol,bytes,flag,duration --num-recs=10
            sIP|            dIP|sPort|dPort|pro|     bytes|   flags| duration|
  10.1.117.100|     10.13.37.1|39286| 4017|  6|       412|FS PA   |    0.018|
    10.13.37.1|   10.1.117.100| 4017|39286|  6|       624|FS PA   |    0.018|
  10.1.109.100|     10.13.37.1|33816| 4017|  6|       412|FS PA   |    0.017|
    10.13.37.1|   10.1.109.100| 4017|33816|  6|       624|FS PA   |    0.017|
  10.1.108.100|     10.13.37.1|38560| 4017|  6|       412|FS PA   |    0.016|
    10.13.37.1|   10.1.108.100| 4017|38560|  6|       624|FS PA   |    0.016|
  10.1.112.100|     10.13.37.1|36906| 4017|  6|       412|FS PA   |    0.026|
    10.13.37.1|   10.1.112.100| 4017|36906|  6|       624|FS PA   |    0.025|
  10.1.104.100|     10.13.37.1|55000| 4017|  6|       412|FS PA   |    0.023|
    10.13.37.1|   10.1.104.100| 4017|55000|  6|       624|FS PA   |    0.023|
```

Figure 4.5: rwcut showing the first 10 packets.

It is possible to further merge all the flow records to contain all the teams' network traffic in a single file. This can be done taking advantage of the *rw-combine* tool available in SiLK. In a folder containing the flow records of each team, it is possible to run the following command to merge them into a single SiLK flow record:

```
rwcombine --output-path=./combined.rw *.rw
```

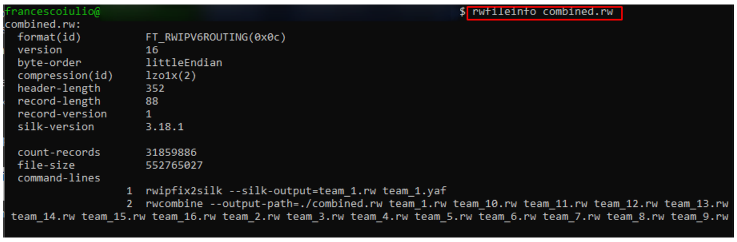


```
-rwxrwxrwx 1 [redacted] 34822477 team_1.rw
-rwxrwxrwx 1            32241879 team_10.rw
-rwxrwxrwx 1            33590975 team_11.rw
-rwxrwxrwx 1            34070294 team_12.rw
-rwxrwxrwx 1            31228570 team_13.rw
-rwxrwxrwx 1            33586612 team_14.rw
-rwxrwxrwx 1            36695390 team_15.rw
-rwxrwxrwx 1            35255225 team_16.rw
-rwxrwxrwx 1            39062266 team_2.rw
-rwxrwxrwx 1            37762952 team_3.rw
-rwxrwxrwx 1            31765857 team_4.rw
-rwxrwxrwx 1            35797433 team_5.rw
-rwxrwxrwx 1            29181902 team_6.rw
-rwxrwxrwx 1            42209887 team_7.rw
-rwxrwxrwx 1            34052645 team_8.rw
-rwxrwxrwx 1            31457833 team_9.rw
francescoiulio@[redacted] $ rwcombine --output-path=./combined.rw *.rw
```

Figure 4.6: rwcombine to merge all teams' flow records

4.5 Analysis of Network Traffic Data

Once the data has been converted to the SiLK format, there are several commands that can be leveraged to gain information about the data contained in the network traffic. One of the first commands to run, needed to have an overview of the file to analyse is *rwfileinfo*

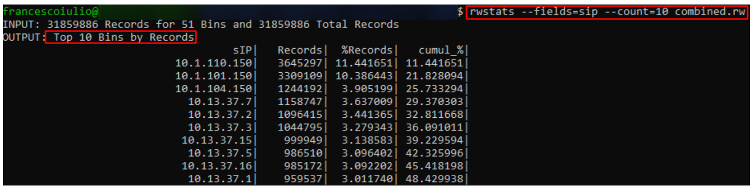


```
francescoiulio@                                         $ rwfileinfo combined.rw
combined.rw:
  format(id)          FT_RWIPV6ROUTING(0x0c)
  version             16
  byte-order          littleEndian
  compression(id)     lzo1x(2)
  header-length       352
  record-length       88
  record-version      1
  silk-version        3.18.1

  count-records       31859886
  file-size           552765027
  command-lines
                   1  rwipfix2silk --silk-output=team_1.rw team_1.yaf
                   2  rwcombine --output-path=./combined.rw team_1.rw team_10.rw team_11.rw team_12.rw team_13.rw
 team_14.rw team_15.rw team_16.rw team_2.rw team_3.rw team_4.rw team_5.rw team_6.rw team_7.rw team_8.rw team_9.rw
```

Figure 4.7: using rwfileinfo command

Another useful command is *rwstats*, that allows analysts to quickly see statistics related to the packets. For example, to see the 10 IP addresses that sent the most packets during the entire competition, it is possible to leverage the following command:

```
rwstats --fields=sip --count=10 ./flowrecord.rw
```



```
francescoiulio@                                         $ rwstats --fields=sip --count=10 combined.rw
INPUT: 31859886 Records for 51 Bins and 31859886 Total Records
OUTPUT: Top 10 Bins by Records
                          sIP|   Records|  %Records|   cumul_%|
                 10.1.110.150|   3645297| 11.441651| 11.441651|
                 10.1.101.150|   3309109| 10.386443| 21.828094|
                 10.1.104.150|   1244192|  3.905199| 25.733294|
                  10.13.37.7|   1158747|  3.637009| 29.370303|
                  10.13.37.2|   1096415|  3.441365| 32.811668|
                  10.13.37.3|   1044795|  3.279343| 36.091011|
                 10.13.37.15|    999949|  3.138583| 39.229594|
                  10.13.37.5|    986510|  3.096402| 42.325996|
                 10.13.37.16|    985172|  3.092202| 45.418198|
                  10.13.37.1|    959537|  3.011740| 48.429938|
```

Figure 4.8: using rwstats to see the major attackers

Data Visualization

The data visualization tools used to visualize network traffic trends and conversation between hosts, were flowplotter, gephi and google charts. Interesting areas of the analysis can be narrowed down by feeding the SiLK flow records to visualization tools such as *Flowplotter*. The following procedure shows how to pipe the content of the SiLK flow record into the visualization tool.

First step is to install flowplotter using git clone to download the project's repository. Subsequently, from the flowplotter's root folder the following command can be executed to generate a line chart, divided per hour, showing the network traffic bytes exchanged:

```
cat team_flow_record.rw | ./flowplotter.sh linechart 3600 bytes > line-chart.html
```

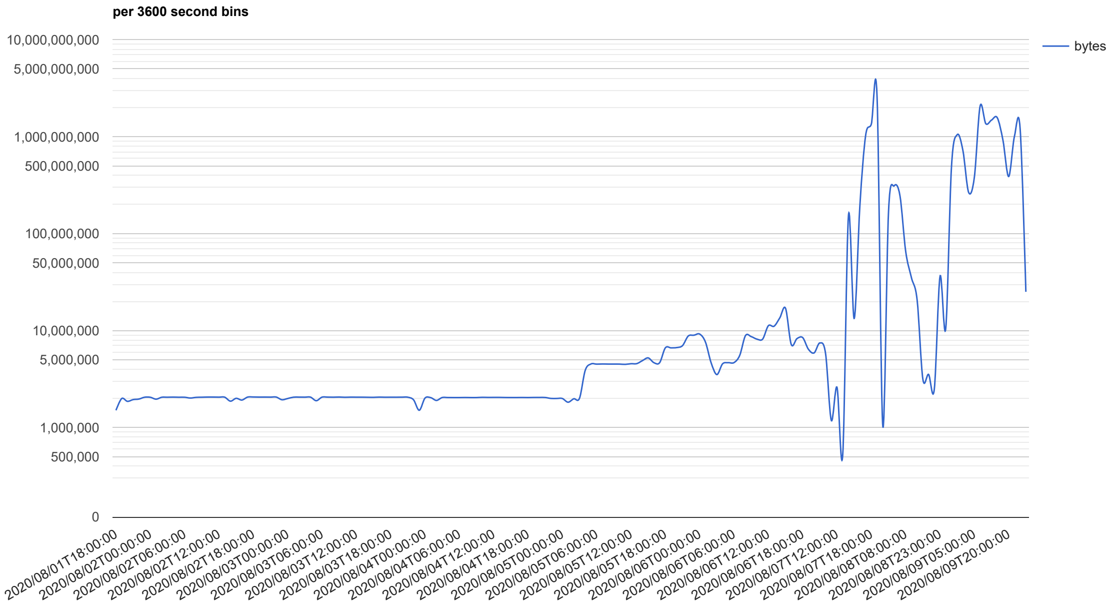
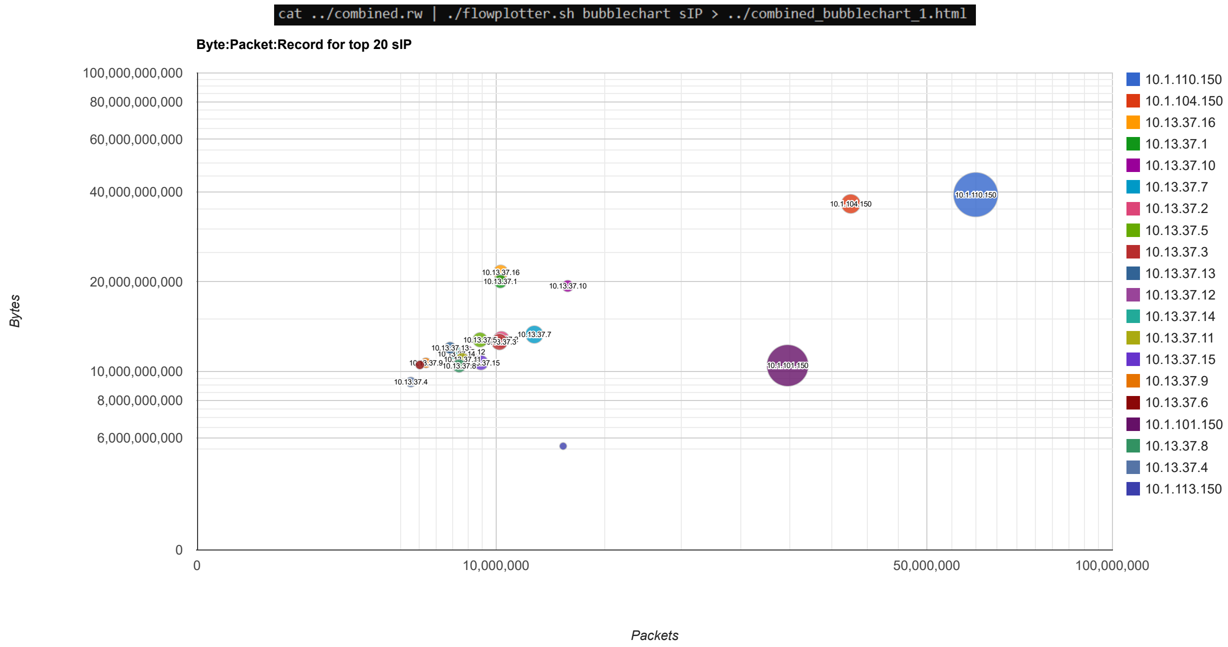


Figure 4.9: linechart produced by the flowplotter command above (piping in team 10 flow records)

The graphs generated from this tool can show several statistics, based on the type of parameters and options passed to the program. Flowplotter should be able to leverage the same functions as the rwstats command from the SiLK toolkit. As mentione before, it is possible to visualize different type of charts depending on the parameters passed to the flowplotter command, as shown in the following images:

```
cat ../combined.rw | ./flowplotter.sh bubblechart sIP > ../combined_bubblechart_1.html
```

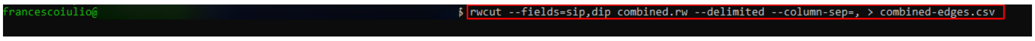


```
cat ../combined.rw | ./flowplotter.sh bubblechart sIP > ../combined_bubblechart_1.html
```

Figure 4.10: a bubble chart showing the top 20 source IP addresses with more records

Finally, it is possible to convert the SiLK flow record into a textual form and use sed to transform the output of rwcut to a dot-file format that can be displayed with Gephi.

```
rwcut --fields=sip,dip combined.rw --delimited --column-sep=, > combined-edges.csv
```



```
francescoiulio@[redacted] $ rwcut --fields=sip,dip combined.rw --delimited --column-sep=, > combined-edges.csv
```

Figure 4.11: the rwcut command that transforms the flow record to a CSV file to import into Gephi

Once the previous command has finished executing, the CSV file generated can be imported into Gephi for further analysis:

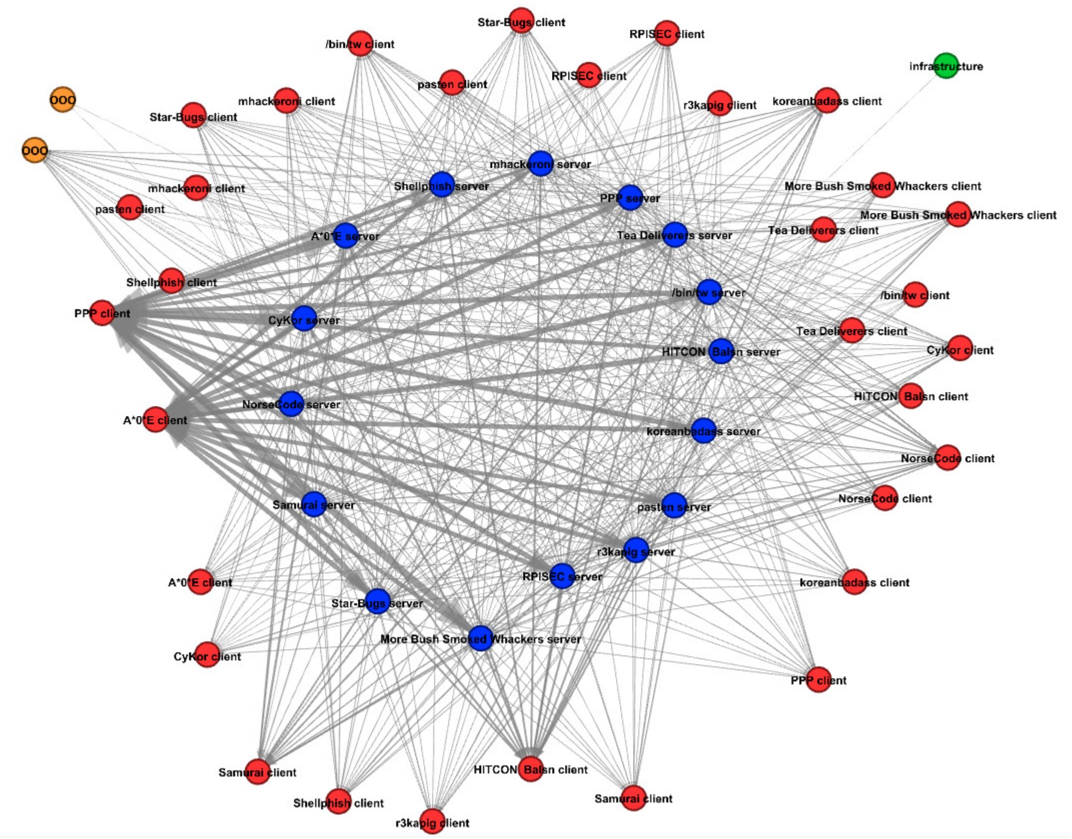


Figure 4.12: Gephi graph obtained from flow records

Due to the large amount of data the data-set included, it would have been impossible to build such a graph using the PCAP files. Using Gephi it is possible to highlight how the attacks went, and what team has attempted to exploit the other.

4.5.1 Database Analysis

The database containing all the information about the game that was played is a 30 GB MySQL file. Unfortunately, there are no open-source systems that can handle a database of this size. Therefore, the only available and cost-efficient way to inspect the content of the dump-file was to open it using a *word processor*. Using a word processor such as *Sublime* it was possible to investigate the contents stored on the game database. This included:

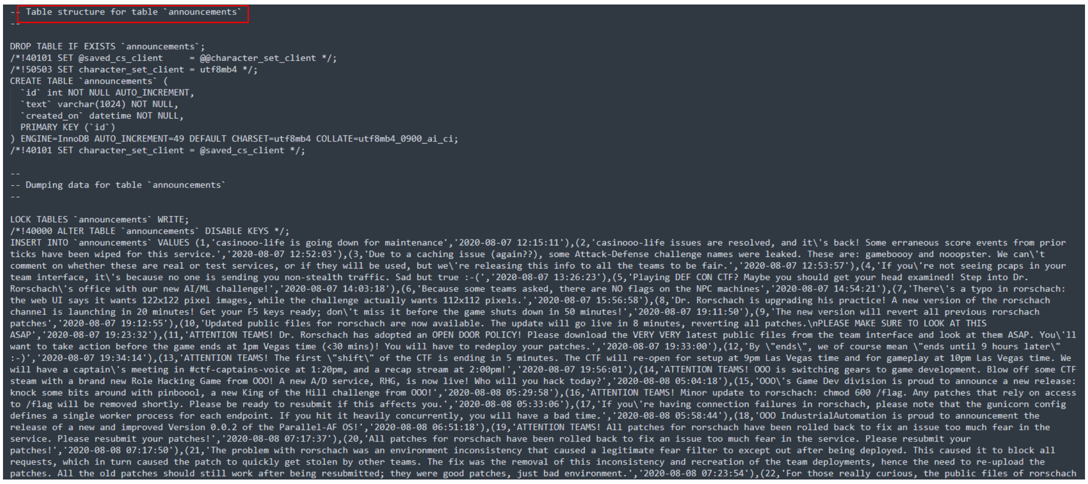
1. The timestamp of the flags correctly submitted, with the related team ID;

2. The content of the patches and the exploits developed during the competition;

3. The flags that were stolen, with the ID of the stealing team, and the team it was stolen from;

4. The game rankings throughout the competition

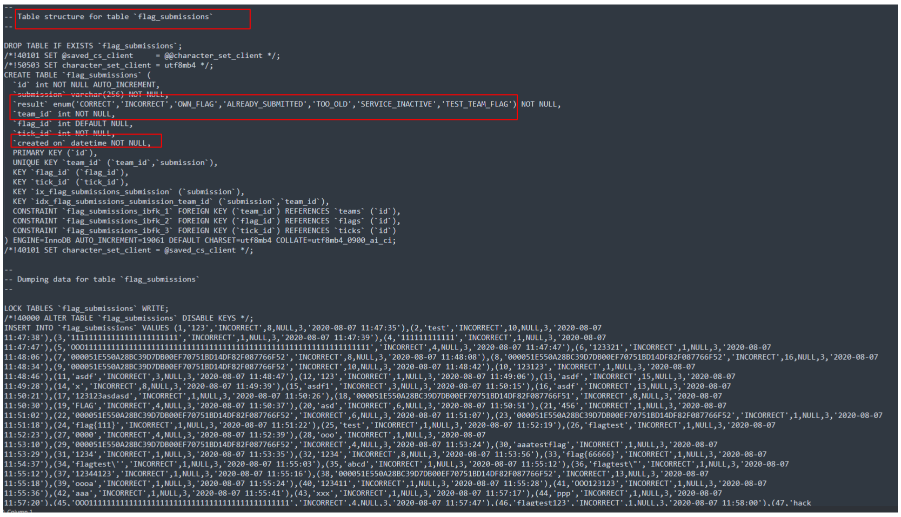
The announcements table found in the database had a list of every communication that was released by the organizers during the competition.



```
-- Table structure for table `announcements`
--

DROP TABLE IF EXISTS `announcements`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `announcements` (
  `id` int NOT NULL AUTO_INCREMENT,
  `text` varchar(1024) NOT NULL,
  `created_on` datetime NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=49 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `announcements`
--

LOCK TABLES `announcements` WRITE;
/*!40000 ALTER TABLE `announcements` DISABLE KEYS */;
INSERT INTO `announcements` VALUES (1,'casinooo-life is going down for maintenance','2020-08-07 12:15:11'),(2,'casinooo-life issues are resolved, and it\'s back! Some erraneous score events from prior
ticks have been wiped for this service.','2020-08-07 12:52:03'),(3,'Due to a caching issue (again??), some Attack-Defense challenge names were leaked. These are: gameboooy and nooopster. We can\'t
comment on whether these are real or test services, or if they will be used, but we\'re releasing this info to all the teams to be fair.','2020-08-07 12:53:57'),(4,'If you\'re not seeing pcaps in your
team interface, it\'s because no one is sending you non-stealth traffic. Sad but true :-(','2020-08-07 13:26:23'),(5,'Playing DEF CON CTF? Maybe you should get your head examined! Step into Dr.
Rorschach\'s office with our new AI/ML challenge!','2020-08-07 14:03:18'),(6,'Because some teams asked, there are NO flags on the NPC machines','2020-08-07 14:54:21'),(7,'There\'s a typo in rorschach:
the web UI says it wants 122x122 pixel images, while the challenge actually wants 112x112 pixels.','2020-08-07 15:56:58'),(8,'Dr. Rorschach is upgrading his practice! A new version of the rorschach
channel is launching in 20 minutes! Get your F5 keys ready; don\'t miss it before the game shuts down in 50 minutes!','2020-08-07 19:11:50'),(9,'The new version will revert all previous rorschach
patches','2020-08-07 19:12:55'),(10,'Updated public files for rorschach are now available. The update will go live in 8 minutes, reverting all patches.\nPLEASE MAKE SURE TO LOOK AT THIS
ASAP','2020-08-07 19:23:32'),(11,'ATTENTION TEAMS! Dr. Rorschach has adopted an OPEN DOOR POLICY! Please download the VERY VERY latest public files from the team interface and look at them ASAP. You\'ll
want to take action before the game ends at 1pm Vegas time (<30 mins)! You will have to redeploy your patches.','2020-08-07 19:33:00'),(12,'By \"ends\", we of course mean \"ends until 9 hours later\"
:-)','2020-08-07 19:34:14'),(13,'ATTENTION TEAMS! The first \"shift\" of the CTF is ending in 5 minutes. The CTF will re-open for setup at 9pm Las Vegas time and for gameplay at 10pm Las Vegas time. We
will have a captain\'s meeting in #ctf-captains-voice at 1:20pm, and a recap stream at 2:00pm!','2020-08-07 19:56:01'),(14,'ATTENTION TEAMS! OOO is switching gears to game development. Blow off some CTF
steam with a brand new Role Hacking Game from OOO! A new A/D service, RHG, is now live! Who will you hack today?','2020-08-08 05:04:18'),(15,'OOO\'s Game Dev division is proud to announce a new release:
knock some bits around with pinboool, a new King of the Hill challenge from OOO!','2020-08-08 05:29:58'),(16,'ATTENTION TEAMS! Minor update to rorschach: chmod 600 /flag. Any patches that rely on access
to /flag will be removed shortly. Please be ready to resubmit if this affects you.','2020-08-08 05:33:06'),(17,'If you\'re having connection failures in rorschach, please note that the gunicorn config
defines a single worker process for each endpoint. If you hit it heavily concurrently, you will have a bad time.','2020-08-08 05:58:44'),(18,'OOO IndustrialAutomation is proud to announcement the
release of a new and improved Version 0.0.2 of the Parallel-AF OS!','2020-08-08 06:51:18'),(19,'ATTENTION TEAMS! All patches for rorschach have been rolled back to fix an issue too much fear in the
service. Please resubmit your patches!','2020-08-08 07:17:37'),(20,'All patches for rorschach have been rolled back to fix an issue too much fear in the service. Please resubmit your
patches!','2020-08-08 07:17:50'),(21,'The problem with rorschach was an environment inconsistency that caused a legitimate fear filter to except out after being deployed. This caused it to block all
requests, which in turn caused the patch to quickly get stolen by other teams. The fix was the removal of this inconsistency and recreation of the team deployments, hence the need to re-upload the
patches. All the old patches should still work after being resubmitted; they were good patches, just bad environment.','2020-08-08 07:23:54'),(22,'For those really curious, the public files of rorschach
```

Figure 4.13: database announcement table

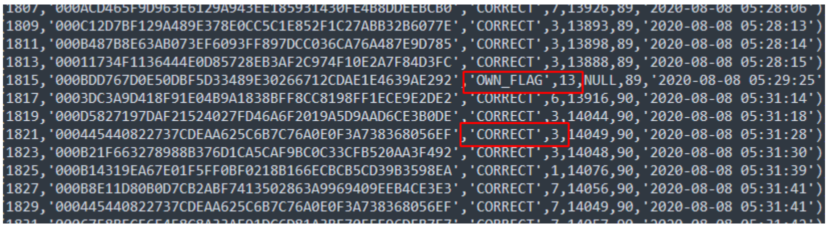
The 'flag submission' table contained all of the flags that were submitted in the game system. In the figure below, it is possible to notice how many flags initially submitted, were vain attempts at assessing the flag submission system.



```
--
-- Table structure for table `flag_submissions`
--

DROP TABLE IF EXISTS `flag_submissions`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `flag_submissions` (
  `id` int NOT NULL AUTO_INCREMENT,
  `submission` varchar(256) NOT NULL,
  `result` enum('CORRECT','INCORRECT','OWN_FLAG','ALREADY_SUBMITTED','TOO_OLD','SERVICE_INACTIVE','TEST_TEAM_FLAG') NOT NULL,
  `team_id` int NOT NULL,
  `flag_id` int DEFAULT NULL,
  `tick_id` int NOT NULL,
  `created_on` datetime NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `team_id` (`team_id`,`submission`),
  KEY `flag_id` (`flag_id`),
  KEY `tick_id` (`tick_id`),
  KEY `ix_flag_submissions_submission` (`submission`),
  KEY `idx_flag_submissions_submission_team_id` (`submission`,`team_id`),
  CONSTRAINT `flag_submissions_ibfk_1` FOREIGN KEY (`team_id`) REFERENCES `teams` (`id`),
  CONSTRAINT `flag_submissions_ibfk_2` FOREIGN KEY (`flag_id`) REFERENCES `flags` (`id`),
  CONSTRAINT `flag_submissions_ibfk_3` FOREIGN KEY (`tick_id`) REFERENCES `ticks` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=19061 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `flag_submissions`
--

LOCK TABLES `flag_submissions` WRITE;
/*!40000 ALTER TABLE `flag_submissions` DISABLE KEYS */;
INSERT INTO `flag_submissions` VALUES (1,'123','INCORRECT',8,NULL,3,'2020-08-07 11:47:35'),(2,'test','INCORRECT',10,NULL,3,'2020-08-07
11:47:38'),(3,'1111111111111111111111111','INCORRECT',1,NULL,3,'2020-08-07 11:47:39'),(4,'111111111111','INCORRECT',1,NULL,3,'2020-08-07
11:47:47'),(5,'OOO1111111111111111111111111111111111111111111111111111111111111111','INCORRECT',4,NULL,3,'2020-08-07 11:47:47'),(6,'123321','INCORRECT',1,NULL,3,'2020-08-07
11:48:06'),(7,'000051E550A28BC39D7DB00EF70751BD14DF82F087766F52','INCORRECT',8,NULL,3,'2020-08-07 11:48:08'),(8,'000051E550A28BC39D7DB00EF70751BD14DF82F087766F52','INCORRECT',16,NULL,3,'2020-08-07
11:48:34'),(9,'000051E550A28BC39D7DB00EF70751BD14DF82F087766F52','INCORRECT',10,NULL,3,'2020-08-07 11:48:42'),(10,'123123','INCORRECT',1,NULL,3,'2020-08-07
11:48:46'),(11,'asdf','INCORRECT',3,NULL,3,'2020-08-07 11:48:47'),(12,'123','INCORRECT',1,NULL,3,'2020-08-07 11:49:06'),(13,'asdf','INCORRECT',15,NULL,3,'2020-08-07
11:49:28'),(14,'x','INCORRECT',8,NULL,3,'2020-08-07 11:49:39'),(15,'asdf1','INCORRECT',3,NULL,3,'2020-08-07 11:50:15'),(16,'asdf','INCORRECT',13,NULL,3,'2020-08-07
11:50:21'),(17,'123123asdasd','INCORRECT',1,NULL,3,'2020-08-07 11:50:26'),(18,'000051E550A28BC39D7DB00EF70751BD14DF82F087766F51','INCORRECT',8,NULL,3,'2020-08-07
11:50:30'),(19,'FLAG','INCORRECT',4,NULL,3,'2020-08-07 11:50:37'),(20,'asd','INCORRECT',6,NULL,3,'2020-08-07 11:50:51'),(21,'456','INCORRECT',1,NULL,3,'2020-08-07
11:51:02'),(22,'000051E550A28BC39D7DB00EF70751BD14DF82F087766F52','INCORRECT',6,NULL,3,'2020-08-07 11:51:07'),(23,'000051E550A28BC39D7DB00EF70751BD14DF82F087766F52','INCORRECT',1,NULL,3,'2020-08-07
11:51:18'),(24,'flag{111}','INCORRECT',1,NULL,3,'2020-08-07 11:51:22'),(25,'test','INCORRECT',1,NULL,3,'2020-08-07 11:52:19'),(26,'flagtest','INCORRECT',1,NULL,3,'2020-08-07
11:52:23'),(27,'0000','INCORRECT',4,NULL,3,'2020-08-07 11:52:39'),(28,'ooo','INCORRECT',1,NULL,3,'2020-08-07
11:53:10'),(29,'000051E550A28BC39D7DB00EF70751BD14DF82F087766F52','INCORRECT',4,NULL,3,'2020-08-07 11:53:24'),(30,'aaatestflag','INCORRECT',1,NULL,3,'2020-08-07
11:53:29'),(31,'1234','INCORRECT',1,NULL,3,'2020-08-07 11:53:35'),(32,'1234','INCORRECT',8,NULL,3,'2020-08-07 11:53:56'),(33,'flag{66666}','INCORRECT',1,NULL,3,'2020-08-07
11:54:37'),(34,'flagtest\'','INCORRECT',1,NULL,3,'2020-08-07 11:55:03'),(35,'abcd','INCORRECT',1,NULL,3,'2020-08-07 11:55:12'),(36,'flagtest\"','INCORRECT',1,NULL,3,'2020-08-07
11:55:12'),(37,'12344123','INCORRECT',1,NULL,3,'2020-08-07 11:55:16'),(38,'000051E550A28BC39D7DB00EF70751BD14DF82F087766F52','INCORRECT',13,NULL,3,'2020-08-07
11:55:18'),(39,'oooa','INCORRECT',1,NULL,3,'2020-08-07 11:55:24'),(40,'123411','INCORRECT',1,NULL,3,'2020-08-07 11:55:28'),(41,'OOO123123','INCORRECT',1,NULL,3,'2020-08-07
11:55:36'),(42,'aaa','INCORRECT',1,NULL,3,'2020-08-07 11:55:41'),(43,'xxx','INCORRECT',1,NULL,3,'2020-08-07 11:57:17'),(44,'ppp','INCORRECT',1,NULL,3,'2020-08-07
11:57:20'),(45,'OOO11111111111111111111111111111111111111111111111','INCORRECT',4,NULL,3,'2020-08-07 11:57:47'),(46,'flagtest123','INCORRECT',1,NULL,3,'2020-08-07 11:58:00'),(47,'hack
```

Figure 4.14: flag submission table

Subsequently, only valid flags are found within the database, as all teams start to successfully exploit services and collect flags.



```
1807,'000ACD465F9D963E6129A943EE185931430FE4B8DDEEBCB0','CORRECT',7,13926,89,'2020-08-08 05:28:06')
1809,'000C12D7BF129A489E378E0CC5C1E852F1C27ABB32B6077E','CORRECT',3,13893,89,'2020-08-08 05:28:13')
1811,'000B487B8E63AB073EF6093FF897DCC036CA76A487E9D785','CORRECT',3,13898,89,'2020-08-08 05:28:14')
1813,'00011734F1136444E0D85728EB3AF2C974F10E2A7F84D3FC','CORRECT',3,13888,89,'2020-08-08 05:28:15')
1815,'000BDD767D0E50DBF5D33489E30266712CDAE1E4639AE292','OWN_FLAG',13,NULL,89,'2020-08-08 05:29:25'
1817,'0003DC3A9D418F91E04B9A1838BFF8CC8198FF1ECE9E2DE2','CORRECT',6,13916,90,'2020-08-08 05:31:14')
1819,'000D5827197DAF21524027FD46A6F2019A5D9AAD6CE3B0DE','CORRECT',3,14044,90,'2020-08-08 05:31:18')
1821,'000445440822737CDEAA625C6B7C76A0E0F3A738368056EF','CORRECT',3,14049,90,'2020-08-08 05:31:28')
1823,'000B21F663278988B376D1CA5CAF9BC0C33CFB520AA3F492','CORRECT',3,14048,90,'2020-08-08 05:31:30')
1825,'000B14319EA67E01F5FF0BF0218B166ECBCB5CD39B3598EA','CORRECT',1,14076,90,'2020-08-08 05:31:39')
1827,'000B8E11D80B0D7CB2ABF7413502863A9969409EEB4CE3E3','CORRECT',7,14056,90,'2020-08-08 05:31:41')
1829,'000445440822737CDEAA625C6B7C76A0E0F3A738368056EF','CORRECT',7,14049,90,'2020-08-08 05:31:41')
```

Figure 4.15: the presence of correct flags inside the database.

The game database has a lot of information that can be useful for the analysis, however the fact that it cannot be open with other type of database management systems, makes the parsing of this information way more harder than expected, as text editors are not the indicated tool for reading from a database.

4.5.2 Ngrep String Search

When stealth communication is not being used by the players, flags are easily detectable on the network, as they are a unique 48 characters string submitted to the game server. An example of a flag is shown in the following picture:

```
'000445440822737CDEAA625C6B7C76A0E0F3A738368056EF'
```

Figure 4.16: a flag of DEF CON 28 finals.

The information found inside the database dump-file has helped define strings that could be used to identify packets with specific content. Using *ngrep*, it is possible to parse through the PCAP file to output the name of the file containing a specific string that was declared in the command.

```
sudo ngrep -q "000E0CBF0F8887A42CF66880ADF4B5DDCC4BA2B0B1EF1EEA" -I merged.pcap
```

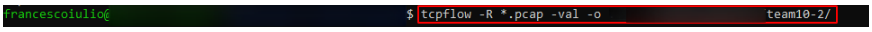
However, this type of research is only possible with one PCAP at a time, and merging all the files in a single PCAP has not given satisfying results. Merging all PCAP files in a folder is possible with the following command:

```
find ./ -type f -maxdepth 1 \ | xargs -I"{}" reordercap "{}" "{}" \ | xargs mergecap -w merged.pcap
```

4.5.3 Tcpflow Analysis

At this point, to try to identify additional information, and hopefully interesting files, out of the network traffic. Using *tcpflow* it was possible to reconstruct the TCP sessions contained in the series of PCAP files that were available from the DEF CON archive. The transmission of files or HTTP requests and responses over the network can happen with the exchange of several TCP packets over time. Tcpflow automatically reads the files and generates a report if anything is found. The use of Tcpflow can potentially extract relevant information such as payloads, exploits, credentials and HTTP sessions. Tcpflow can be executed with the network files grouped in a folder with the following command:

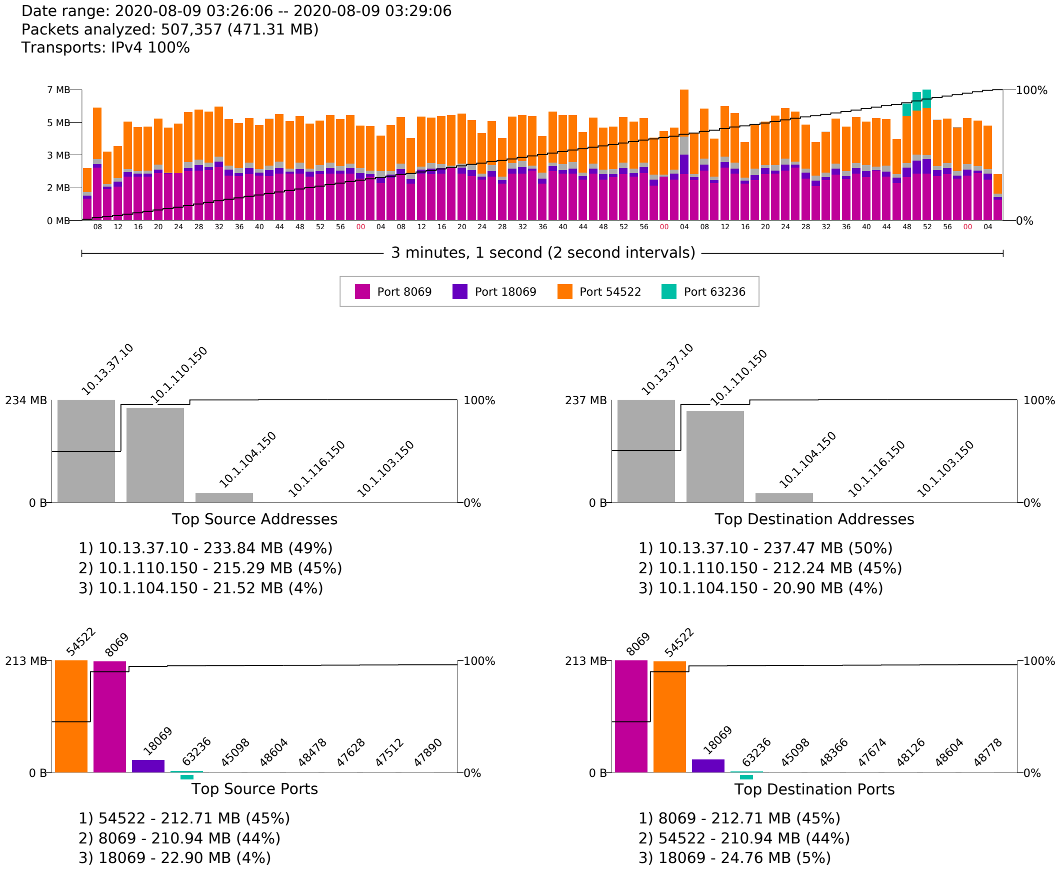
```
tcpflow -R *.pcap -val -o team-x-results/team_x_tcpflow
```



```
francescoiulio@[redacted] $ tcpflow -R *.pcap -val -o [redacted]team10-2/
```

Figure 4.17: command to execute tcpflow in a folder containing PCAP files.

If successful, Tcpflow should then generate a report with a summary of the findings, as seen in the following image:



Date range: 2020-08-09 03:26:06 -- 2020-08-09 03:29:06
Packets analyzed: 507,357 (471.31 MB)
Transports: IPv4 100%

Top Source Addresses

1) 10.13.37.10 - 233.84 MB (49%)
2) 10.1.110.150 - 215.29 MB (45%)
3) 10.1.104.150 - 21.52 MB (4%)

Top Destination Addresses

1) 10.13.37.10 - 237.47 MB (50%)
2) 10.1.110.150 - 212.24 MB (45%)
3) 10.1.104.150 - 20.90 MB (4%)

Top Source Ports

1) 54522 - 212.71 MB (45%)
2) 8069 - 210.94 MB (44%)
3) 18069 - 22.90 MB (4%)

Top Destination Ports

1) 8069 - 212.71 MB (45%)
2) 54522 - 210.94 MB (44%)
3) 18069 - 24.76 MB (5%)

Figure 4.18: report generated by tcpflow when analyzing part of the PPP team PCAP files

4.6 Summary

This chapter has shown how the analysis was carried out to extract relevant information out of the data sets collected. The successful conversion of the network traffic data, together with the correlation of the details contained in the competition's database, have given unexpected insights about the used during the cyber conflict. The possibility to leverage tools like Ngrep, tcpflow and Gephi has made the analysis process smoother. Finally, in the next chapter the paper will discuss the results and suggest recommendations that can be used to improve preparedness and boost situational awareness in future competitions.

Chapter 5

Results and Evaluation

5.1 Overview

This section describes the interesting discoveries resulting from our analysis. The methodology used to make these discoveries involved visualizing the network traffic data to spot interesting or anomalous artifacts along with a manual analysis of the captured network packets.

5.2 Discoveries

In the following section, some of the tactics and tools discovered through the qualitative and quantitative research performed for this dissertation are described.

5.2.1 Tactics Discovered

The qualitative research done in regards to capture the flag competition has highlighted a few strategies and tactics that are among the most used in competitions. These include, but are not limited to: *obfuscating traffic, teasing adversaries, exploit sensing, exploit recycling, network monitoring, flags detection, high-performance computing* through *resources pooling,* as well as prohibited techniques as: *ddos-ing adversaries, attacking cloud providers, attacking provider's supply-chain, exploiting hosting infrastructure.* Listed below, are some of the main strategies and tactics used by participating teams:

- Obfuscating traffic: traffic obfuscation is a valid technique that can prevent opponents from capturing and reusing exploits, as when analysed they would appear as they are encrypted, to hide flags that have just been stolen, or to make the cyber operations carried out increasingly stealth. There are many tools that can be used to obfuscate exploits, and multi-stage exploits are often used to make reverse engineering more difficult.

- Teasing adversaries: it may include several other techniques, such as *flag* tampering, *backdooring files*, using *ip rules to disrupt traffic*, or leaving a *zip bomb* for adversaries to open on their system.

  - Flags tampering, consists in changing the value of the text file containing the flags, making the game system reject the flag once it is stolen and subsequently submitted by one of the adversaries.
  - Backdooring files is the strategy that is often used to embed malicious code inside files that could be appealing for the opponents. For example, an exploit could be developed with a backdoor to execute a reverse shell connection to the creator of the exploit every time this is used by others elsewhere.
  - IP rules can be used to drop, reroute or slow down traffic, making reverse shell connections and opponents' control over a system highly unstable.
  - Zip bombs [46] are files that when unzipped make the system they are extracted on, unusable. A successful technique can be to remove all archiving and extraction tools off the system to defend, and store a zip bomb file called 'credentials', 'password' or another name that could appeal to other teams. Once the file is moved over to their machine and extracted, it will make their system run out of memory and shutdown.

- Network monitoring was the preferred technique to steal flags, exploits and achieve a better situational awareness. Some teams deploy Intrusion Detection System instances in an attempt to detect attacks in the moment they are launched, however often in such competitions exploits are custom-made and cannot be identified through the use of signatures. On the other hand, flags can be easily detected over the network, therefore making exfiltration of flags a good indicator of compromise for teams that are actively monitoring their infrastructure.

- Exploit recycling is done by monitoring the network, collecting adversaries exploits and reuse them to exploit the same target service. Teams have proved to be able to capture exploits used by other teams and leverage them for their own objectives. At times, detecting an exploit could be an impossible task, since the DEF CON 28 CTF has provided *stealth* channels of communications that could be used by teams to encrypt the data exchanged with the target servers (i.e. flags, exploits, patches, etc.).

- High-performance computing through resource pooling is a strategy that can be leveraged today due to the increased availability of several cloud server instances readily available in minutes. Teams can take advantage of external computing resources and pool them together to carry out brute-forcing, password spraying or dictionary attacks in a timely fashion.

- Attacking the teams' clients, used to connect to the jumpbox, could also be a successful tactic that would allow full control over the opponents machine, as well as the possibility to gather all the information the victim has collected about the other participants.

Other techniques, that have turned out to be successful when employed in a capture the flag competition are:

- Honey Tricks, consisting in the use of *Honey Tokens* and *Honeypots* to trick adversaries into thinking they are looking at sensitive information, or exploiting a real service, making them delay their operations.
- Vulnerability assessment, system's hardening measures and patch deployment to be carried out first in Attack-Defense capture the flag contests.
- Deceiving Attackers
- Teams may establish secret agreements between them to work in a mutually beneficial way, for example by sharing information about the adversaries, flags the patches, or exploits developed.
- Requesting external support from fellow CTF players and hackers in forums and chat rooms to help with solving the challenges.

In the first half of the competition, almost all attacks were stealthed [47]. However, combined with the per-service flag limits (discussed next), this represented a serious reduction in the total number of points a team would be able to get from a service. Later in the game, teams made different stealth decisions based (seemingly) on whether the victim team was ahead or behind them in score, whether the service would be retired soon, or whether they felt that the victim team had the network analysis capabilities to identify the attack [47]. One outcome from stealth attacks is that it made it extremely difficult for teams to defend against difficult-to-find but easy-to-exploit bugs [47].

5.2.2 Tools Discovered

The preferred tool employed for exploit development was Python; from the SQL database dump of the competitions, it was possible to retrieve the name of a few exploits used by the teams. Python, together with Bash, was used to automate tasks, sniff networks, deploy patches as well as exploiting adversaries' machines services. Furthermore, the database has also revealed the execution of a few ELF binaries, however a further analysis of such binaries was not possible as their code was compiled and it was not readable. Obviously, the teams could have employed all sorts of tools and programming languages to address the challenges, however these were not directly recognizable from the database dump or the network traffic capture. Other useful type of tools these teams might have employed, include:

- **Disassemblers:** the vulnerabilities inside the services can be statistically analyzed by disassembling the code related to services to assembly code.
- **Debuggers**: with debuggers it is possible to analyze the behaviour of the services while they are running.
- **Exploitation Frameworks:** are commonly used by participants of CTF competitions to develop exploits, payloads and obfuscate their operations. Exploitation frameworks commonly used include the Metasploit Framework, the Empire Framework and Cobalt Strike, which all leverage a library of exploit codes and shellcodes to exploit vulnerabilities and control the exploited targets. While the exploits used in exploitation frameworks are targeting known software and services, the vulnerabilities created in

capture the flag games are often tailored for the competition, therefore the exploits contained in the frameworks may not always successfully work in these environmments.

- **Scripting and progamming languages:** the use of bash and python was extensive during the competition, as seen from the database dump-file. Teams have used these scripting languages to automate certain tasks, for example alerting in case a new user connects to their machine, or automate the launching of attacks and exploits. Together with Bash and Python, also Perl and Golang are often useful for this purpose.
- **Network traffic analysis tools:** tools such as Wireshark, Tshark, Tcpdump and Tcpflow (as well as many others) allow teams to analyze the network and to re-use exploits captured from other teams. The network traffic captured will show all the inbound and outbound data in their network, allowing them to search for exploits and/or exfiltrated flags.

5.2.3 Visualising Statistics off packet captures

The possibility to easily see the traffic generated during the competition (Figure 5.1) has allowed the research to focus on specific areas of the competition.

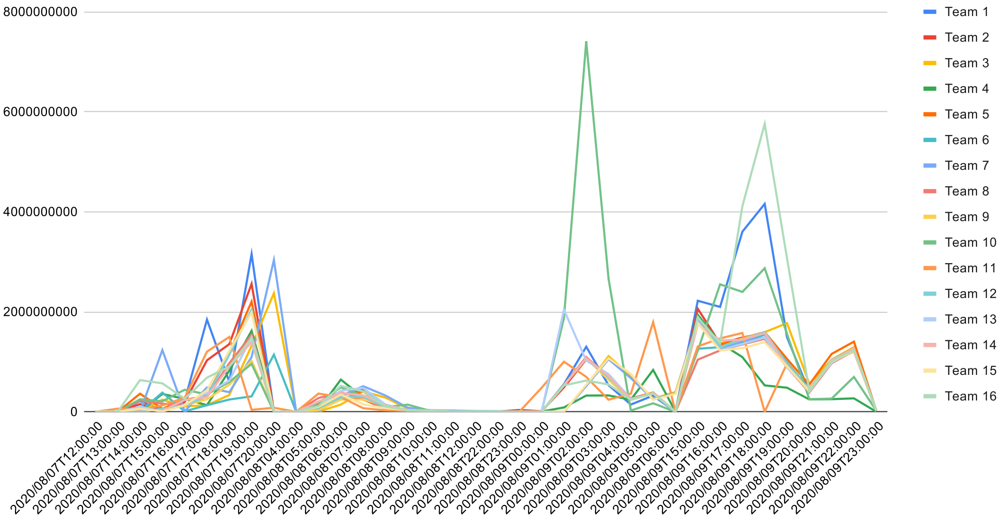


Figure 5.1: traffic generated during the competition

Figure 5.2, shows a *force directed link graph* that was effortlessly generated leveraging *SiLK Flow Records* and *Flowplotter*. Flowplotter has read the information contained inside the flow record, discarding the corrupted TCP flows and establishing the correct information to display in the chart, according to the filter rules and parameters passed to the flowplotter command and the type of graph that was selected. In Figure 5.2 the results is a map showing the attacks that have been carried out in the attack-defense game of DEF CON 28 CTF finals between team members' clients (red dots) and teams' vulnerable servers (green dots).

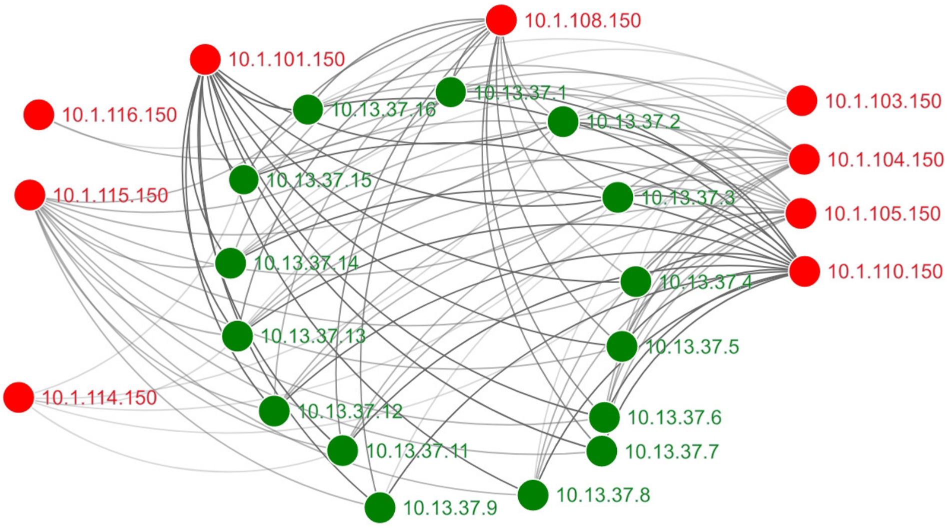


Figure 5.2: Force Directed Link Graph generated with SiLK output and FlowPlotter.

Furthermore, through the graphical aids generated by flowplotter is possible to analyse the most active teams during the competition. For instance, by visualizing the teams that have sent the greatest amount of packets during the CTF, it is possible to state that Tea Deliverers, A*0*E and Plaid Parliament of Pawning were the teams that were more actively exploiting services and attacking opponents.

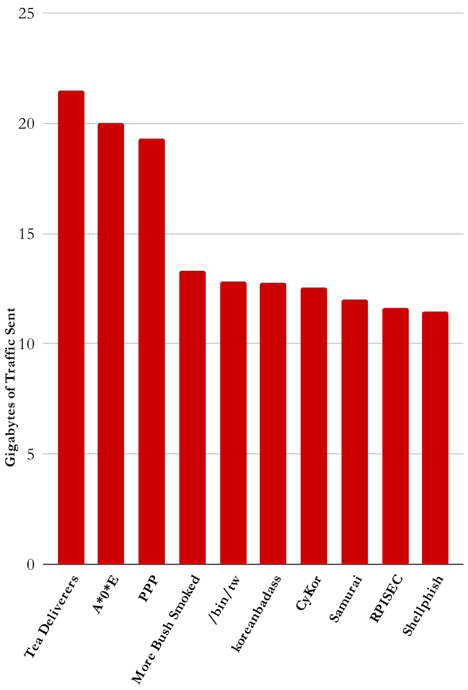


Figure 5.3: The ten more active attacking teams during DEF CON 28 CTF.

The same visualization technique comes in handy when attempting to discover which were the most attacked teams. In fact, the infrastructure of the teams shown in the figure on the right, are among the ones that endure the most attacks from the opponents, during the event.

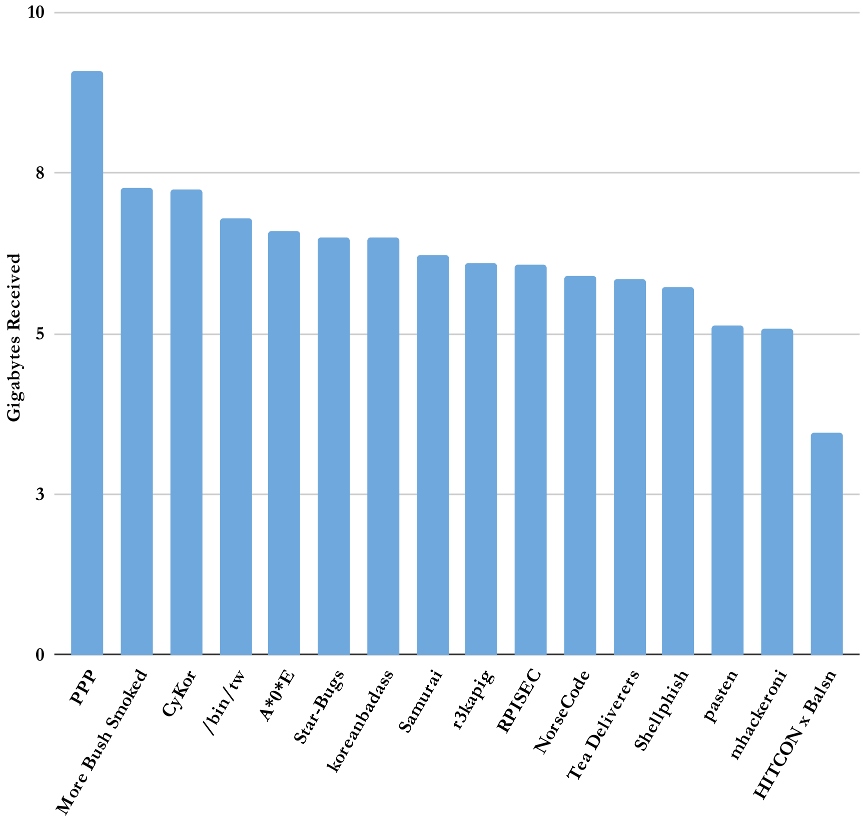


Figure 5.4: The column chart in this figure shows the teams that have received the larger amount of attacks to their infrastructure.

Further information can be obtained from the Gephi graph that was produced during the analysis, by converting the flow records to CSV format. For example, it is possible to highlight the attacks suffered by A*0*E server, with attacks heavily carried out by PPP mainly, as well as HITCON X Balsn and Samurai.

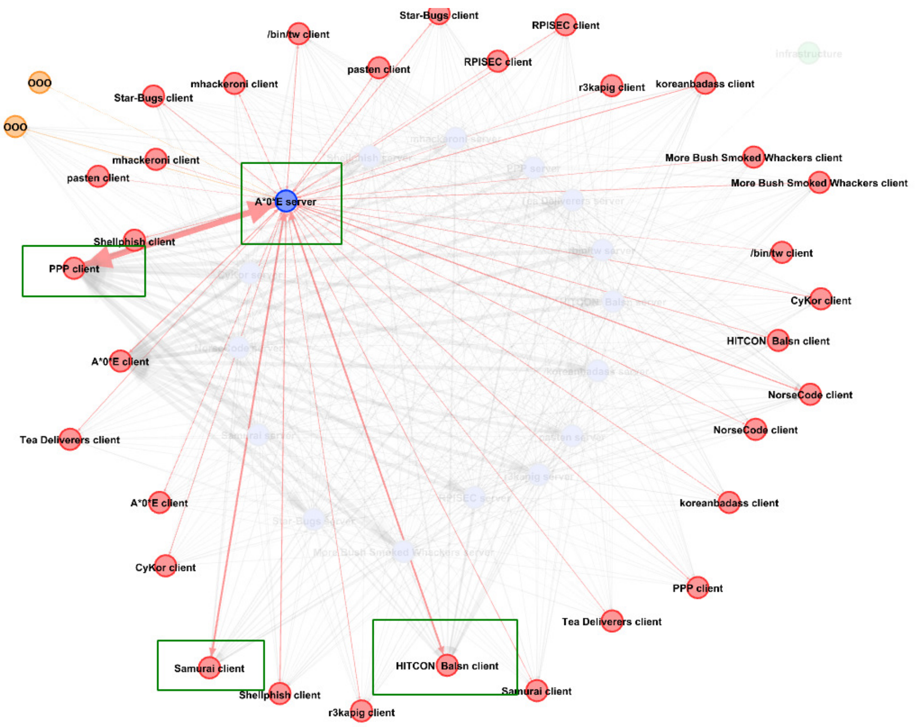


Figure 5.5: The attacks suffered by A*0*E.

The servers attacked by Tea Deliverers are easily spotted through Gephi.

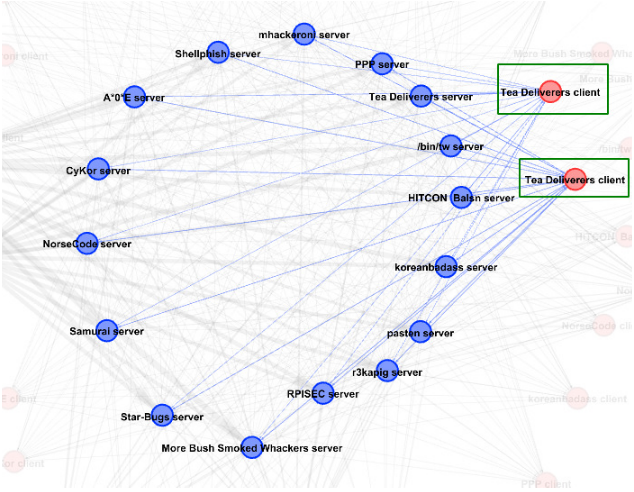


Figure 5.6: The attacks carried out by the Tea Deliverers team.

5.3 Evaluation

At the end of this research, the amount of data collected in regards to capture-the-flag competitions and Attack-Defense strategies and tactics presented valuable information, and a satisfactory level of findings emerged from an in-depth analysis of the network traffic. When inspecting the network traffic two main challenges were encountered:

1. The amount of traffic produced during the competition was impossible to manually sort through with the amount of time and the computing power available.
2. Exploits could not be carved out from the network traffic as no known signature could be used to reference the data and identify the actual exploit code.

The findings are a mix of qualitative and quantitative research, and the thorough inspection of a critical series of resources was necessary to complete the research despite the limited computing power available and the time constraints related to the project. The use of the following sources was fundamental for this research:

- Collection of packet capture files related to the DEF CON 28 CTF;
- Game Stats collected from DEF CON 28 CTF SQL Database dump;
- Academic papers related to the subject of Capture the Flag competitions;
- Research about the methodologies and procedures to analyze large volumes of network traffic;
- On-line videos, blogs and articles related to the topic of big data analysis and analysis of network capture files;
- Network Analysis and digital forensics tools required for deep packet inspection and statistical analysis of network data;

The information derived from this analysis can conclude that the teams that have leveraged offensive techniques the most and stealth communication channels were leveraged for the first part of the competition to obfuscate flag exfiltration and exploits. Furthermore, the teams have used scripting languages to automate their processes and actively monitored the network traffic to steal exploits and flags. Subsequently, with the use of regular expressions and tools such as ngrep, the strings contained in the database could be sprayed against the network traffic PCAP files to help identify packets that contained flags or exploits. However, this was only in theory and often this kind of search gave unsatisfying results. In many cases, teams obfuscated their exploits, as this was allowed by the game itself and many teams found the feature to be useful for at least the first half of the game. Also, the teams were not using publicly available or proof-of-concept exploit repositories, otherwise it would have been visible from the network traffic scans.

5.4 Recommendations

The following section was developed throughout the dissertation research, by encountering new information about the TTP and strategies employed in competitions and taking note of these to formulate suggestions that can prove to be very useful during a computer conflict.

5.4.1 Tools Used during competitions

Below are listed some of the tools that can be used during an attack-defense CTF competition:

Table 5.1: Tools to use during competitions - part 1

Category	***Tool***	***Description***
Red Teaming	SharpCollection [48]	Builds of common C# offensive tools, fresh from their respective master branches built and released in a CDI fashion.
	sqlmap [49]	Sqlmap is an open source penetration testing tool that automates the process of detecting and exploiting SQL injection flaws and taking over of database servers.
	Metasploit Framework [50]	Framework for exploit development, command and control and easy management of exploitation and post-exploitation phases.
	Empire [51]	A post-exploitation framework that includes a pure-PowerShell Windows agents, Python 3.x Linux/OS X agents, and C# agents.
	Obfuscator [52]	The program is designed to obfuscate the shellcode. Currently the tool supports 2 encryption.
	The Backdoor Factory (BDF) [53]	It patches executable binaries with user desired shellcode and continue normal execution of the prepatched state.
	Prism backdoor [54]	The tool creates user space stealth reverse shell backdoors.
	icmpdoor [55]	Icmpdoor is a covert ICMP reverse shell written in Python 3 and scapy.
Blue Teaming	unhide [56]	Forensic tool to find hidden processes.
	res·pound·er [57]	A tool that detects presence of a Responder in the network and identifies compromised machines before hackers run away with the loot (hashes, flags, etc.).
	Artillery [58]	Artillery is a combination of a honeypot, monitoring tool, and alerting system.
Network Analysis	Bluespawn [59]	Helps blue teams monitor systems in real-time against active attackers by detecting anomalous activity.
Tricking Adversaries	ZipBomb [60, 46]	Malicious archive file designed to crash or render useless the program or system [61].
	Portspoof [62]	Camouflage technique that configure the network so that attackers' port scans will become entirely mangled and meaningless (i.e. opens all TCP ports to make network scans useless)
	Deploy-Deception [63]	Deploy-Deception is a PowerShell module to deploy active directory decoy objects.

Table 5.2: Tools to use during competitions - part 2

Category	***Tool***	***Description***
Honeypots	Honey Tokens [64]	Allow to track files, or data executed without permission. Can be given a name to trick attackers into stealing or open the file.
	T-Pot [65]	Is a collection of several honeypots that can be deployed in a network to graphically visualize the attacks occurring towards the system.
	LaBrea [66]	A program that creates a honeypot taking over unused IP addresses on a network and creating "virtual machines" that answer to connection attempts. LaBrea answers those connection attempts in a way that causes the machine at the other end to get "stuck", sometimes for a very long time.
Decompilers and Debuggers	IDA Pro [67]	A powerful disassembler and a versatile debugger.
	GDB [68]	Portable debugger that runs on many Unix-like systems and works for many programming languages.
	strace [69]	strace is a diagnostic, debugging and instructional userspace utility for Linux. It is used to monitor and tamper with interactions between processes and the Linux kernel, which include system calls, signal deliveries, and changes of process state [70].
	binwalk [71]	A fast, easy to use tool for analyzing, reverse engineering, and extracting firmware images.
Crypto	Cyberchef [72, 73]	A web app for encryption, encoding, compression and data analysis.
Communication and Collaboration	Rocket.chat [74]	Open-source fully customizable communications platform developed in JavaScript for organizations with high standards of data protection.
	Discord [75]	An open-surce chat engine that is easy to set up.
	Etherpad [76]	Etherpad is a real-time collaborative editor scalable to thousands of simultaneous real time users. It provides full data export capabilities, and runs on your server, under your control.

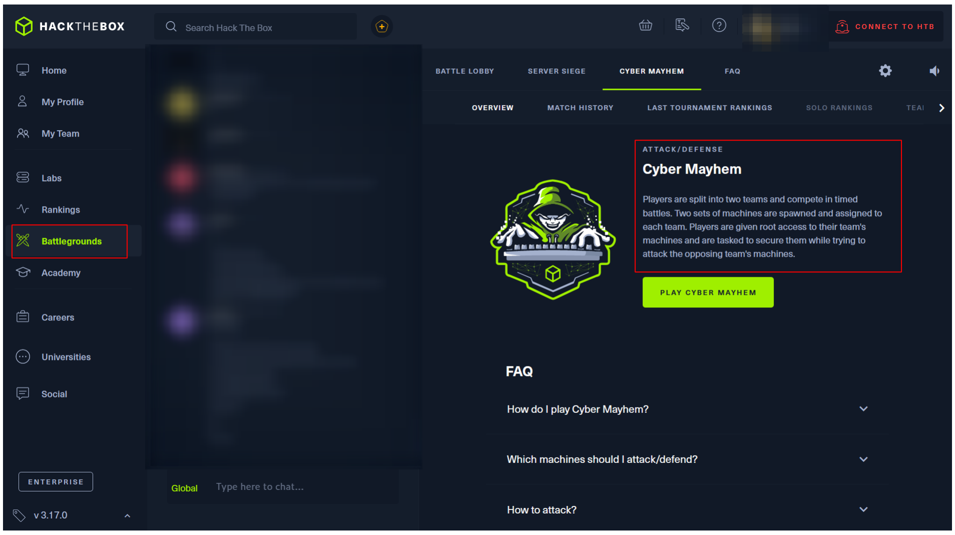
5.4.2 Ethical Hacking Training Platforms

In this section have been discussed few options available when willing to practice for capture the flag competitions, for both jeopardy and attack-defense style games. In fact, the platforms mentioned in Table 5.1 are among the best providers of vulnerable virtual environments that are remotely accessible, safe and intended for individuals who want to improve their cybersecurity skills. Also important to note is the very helpful community that actively participates in the development of the platforms and by helping each other in case of vulnerable machines that are found hard to exploit. As in CTF competitions, the goal on most of these platform is to collect as many user and root flags as possible, thus demonstrating that administrative privileges where achieved on such system and therefore receiving points that increase the score in the platform's ranking.

Table 5.3: Platforms helpful for capture the flag practice and training

Training Platform	Description
OFFENSIVE security [77]	Proving Grounds presents machines that are similar to the ones available on Vulnhub [78], however it allows easy practice without setting up a lab environment (as required by Vulnhub) and simply connecting via VPN to their network infrastructure. These machines are especially indicated for individuals who want to test a more realistic environment.
Hack The Box PEN-TESTING LABS [79]	Hack the Box has recently made radical changes to the what they present and manage their gamified cybersecurity training platform. There is the possibility to gradually develop skills through learning paths, academies, jeopardy challenges, vulnerable virtual machines and Attack-Defense games. The platform has a lot of content in terms of CTF and cybersecurity training, and allows to track the ranking among hackers playing worldwide.
Try Hack Me [80]	Try Hack Me is another platform that allows enthusiasts from all levels to join the CTF world. There are tons of labs and virtual machines, and every task has its own hints and guided execution. The platform really attempts to make fundamental concepts very well understood before progressing to more advanced concepts.

Other platforms include over-the-wire, gh0st, hackthissite and virtual hacking labs. While Try Hack Me and Proving Ground all offer machines containing real-life vulnerable services, jeopardy challenges and a lot of educational content, Hack The Box has recently also introduced an attack-defence environment in their infrastructure. For example, is possible to practice Attack-Defense CTF formats on Hack The Box from *the Battlegrounds* section of their website, which allows players to compete in timed battles [81].



Cyber Mayhem

Players are split into two teams and compete in timed battles. Two sets of machines are spawned and assigned to each team. Players are given root access to their team's machines and are tasked to secure them while trying to attack the opposing team's machines.

FAQ

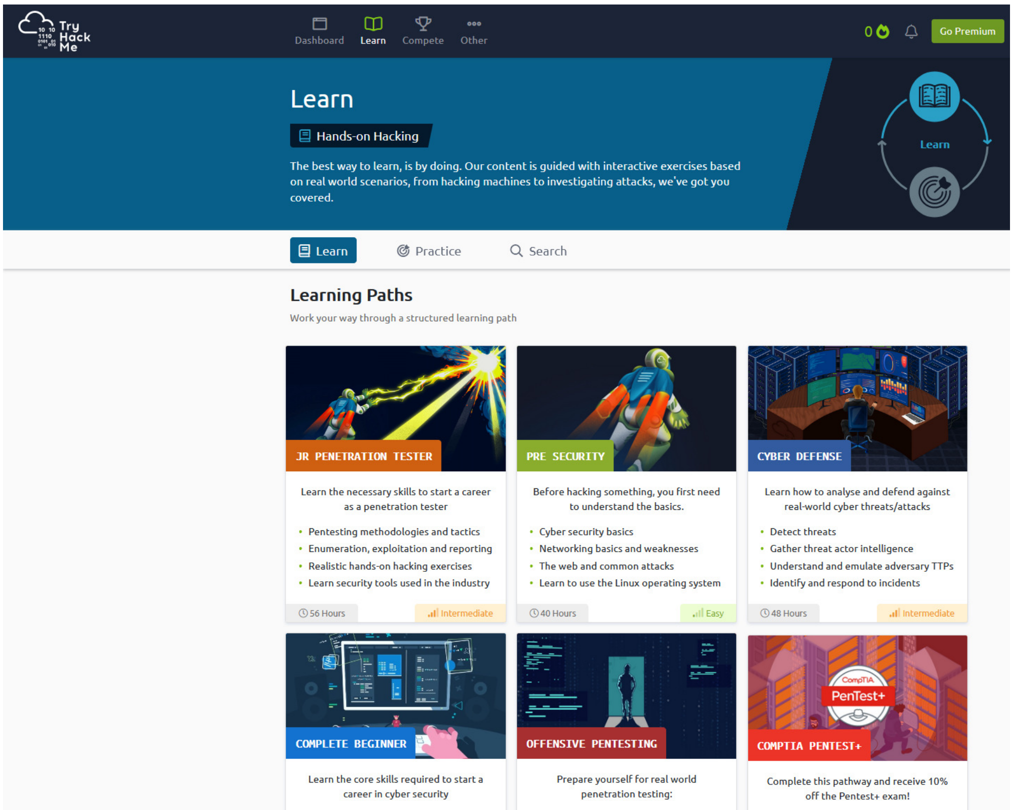
How do I play Cyber Mayhem?

Which machines should I attack/defend?

How to attack?

Figure 5.7: Cyber Mayhem real-time cyber conflict on Hack the Box.

Try Hack Me is another excellent platform that allows beginners to gradually develop their skills, through guided labs and learning paths that really explore every aspect of the offensive and defensive concepts, tools and procedures to



Learn

Hands-on Hacking

The best way to learn, is by doing. Our content is guided with interactive exercises based on real world scenarios, from hacking machines to investigating attacks, we've got you covered.

Learning Paths

Work your way through a structured learning path

JR PENETRATION TESTER

Learn the necessary skills to start a career as a penetration tester

- Pentesting methodologies and tactics
- Enumeration, exploitation and reporting
- Realistic hands-on hacking exercises
- Learn security tools used in the industry

56 Hours · Intermediate

PRE SECURITY

Before hacking something, you first need to understand the basics.

- Cyber security basics
- Networking basics and weaknesses
- The web and common attacks
- Learn to use the Linux operating system

40 Hours · Easy

CYBER DEFENSE

Learn how to analyse and defend against real-world cyber threats/attacks

- Detect threats
- Gather threat actor intelligence
- Understand and emulate adversary TTPs
- Identify and respond to incidents

48 Hours · Intermediate

COMPLETE BEGINNER

Learn the core skills required to start a career in cyber security

OFFENSIVE PENTESTING

Prepare yourself for real world penetration testing:

COMPTIA PENTEST+

Complete this pathway and receive 10% off the Pentest+ exam!

Figure 5.8: Some of the learning paths available on Try Hack Me.

The features of these platforms can help every aspirant capture the flag player to efficiently develop the strategies in preparation to real competitions, and help spread more advanced concepts in the CTF players community, introducing the essential defensive skill-set in the offensive dominated world of computer service exploitation.

5.4.3 Improving Future Competitions

As also mentioned in the literature review, several studies that have studied how to improve future capture the flag competitions. The use of cloud computing and containers has already been adopted by the majority of the CTF platforms and organizers, as also suggested by [33] however competitions still present issues when subject to thousands of megabytes of exploits executing and a literal cyber warfare is happening on the network. With the shift to all computing infrastructures to the cloud environment, it will be possible to leverage container orchestration systems such as Kubernetes [82] to automate the deployment and management of containerized applications, such as vulnerable services and capture-the-flag jeopardy challenges, improving performance and scalability. Such solutions are often seen in capture the flag platforms such as the ones discussed previously (Pentester Labs, Try Hack Me, Hack The Box, Proving Grounds, etc.), however they are rarely implemented in competitive capture the flag events. Furthermore, the automated generation of CTF challenges as discussed in [36], should be further developed and customized to be implemented for a CTF event. The organizers of CTF competitions should aim at improving the post-mortem analysis, create better organized and structured walkthroughs, and develop new infrastructures, one that is more scalable and performant, and hopefully with the advent of new technology and renewable energy sources this will also become gradually and increasingly more cost-effective.

5.5 Future Work and Conclusions

This research has demonstrated the procedure to perform the analysis of a capture the flag competition using network forensics tools, and attempt to derive statistical information, as well as describing some of the tactics and tools used by the participants of CTF Attack-Defense events. The analysis process has produced information coming directly from the network traffic data sets of DEF CON 28 CTF finals, and through the use of specific network forensic tools, such as YAF and SiLK, it was possible to feed visualization software with network data to derive areas of interest to further investigate. Subsequently, an evaluation of the information extracted by the correlation of the qualitative and quantitative research made the discussion about the strategies employed by competitions' teams possible. Further studies may be directed at analyzing the data-sets utilizing machine learning approaches, as was done in the work done for cyber attribution during computer conflicts. The world of cybersecurity and capture the flag competitions revolves around hacking and exploiting services, therefore an improved methodology to identify and reassemble custom exploits, through a heuristic analysis of the obfuscation techniques and malicious code used by exploit developers, could greatly benefit the studies in regards to tactics and techniques used by CTF players. Subsequently, an automated process for converting and analysing network data and flow records could be created to better manage large volumes of network traffic data. Capture the flag competitions are essential for the cybersecurity industry and community, and as the researchers have demonstrated, have inspired thousands of students and enthusiasts to pursue the knowledge of the noble art of defending computer systems, including those devices that are daily used in our lives, making our personal space slightly more vulnerable to external attacks.

Bibliography

[1] CTFtime.org / All about CTF (Capture The Flag), 2021. URL: https://ctftime.org/event/list/?year=2020&online=-1&format=0&restrictions=-1.

[2] James Spiro. Cyberattacks on critical infrastructure jump by 41% in first half of 2021 - CTech, 2021. URL: https://www.calcalistech.com/ctech/articles/0,7340,L-3915536,00.html.

[3] Mutsuo NOGUCHI and Hirofumi UEDA. An Analysis of the Actual Status of Recent Cyberattacks on Critical Infrastructures : NEC Technical Journal, 2021. URL: https://www.nec.com/en/global/techrep/journal/g17/n02/170204.html.

[4] European Union Agency for Cybersecurity. *ENISA threat landscape for supply chain attacks.* Publications Office, LU, 2021. URL: https://data.europa.eu/doi/10.2824/168593.

[5] Brenda Robb. The State of Ransomware in 2021, 2021. Section: Ransomware. URL: https://www.blackfog.com/the-state-of-ransomware-in-2021/.

[6] Davey Winder. The five most important ransomware attacks of 2021, September 2021. URL: https://www.raconteur.net/technology/the-five-most-important-ransomware-attacks-of-2021/.

[7] Michael Sentonas. Council Post: Ransomware: Double The Trouble In 2021, 2021. Section: Innovation. URL: https://www.forbes.com/sites/forbestechcouncil/2021/09/24/ransomware-double-the-trouble-in-2021/.

[8] Makoto Nakaya, Masahiro Okawa, Masahiro Nakajima, and Hiroyuki Tominaga. A Support Environment and a Trial Practice of Hacking Contest with Attack and Defense Style on a Game Website. In *2017 21st International Conference Information Visualisation (IV)*, pages 360–365, London, July 2017. IEEE. URL: http://ieeexplore.ieee.org/document/8107997/, doi:10.1109/iV.2017.78.

[9] Wye Kede Jerel Yam. Strategies used in capture-the-flag events contributing to team performance. page 116, 2016.

[10] C. Cowan, S. Arnold, S. Beattie, C. Wright, and J. Viega. Defcon Capture the Flag: defending vulnerable code from intense attack.

In *Proceedings DARPA Information Survivability Conference and Exposition*, pages 120–129, Washington, DC, USA, 2003. IEEE Comput. Soc. URL: `http://ieeexplore.ieee.org/document/1194878/`, `doi:10.1109/DISCEX.2003.1194878`.

[11] Menelaos Katsantonis, Panayotis Fouliras, and Ioannis Mavridis. Conceptual analysis of cyber security education based on live competitions. In *2017 IEEE Global Engineering Education Conference (EDUCON)*, pages 771–779, Athens, Greece, April 2017. IEEE. URL: `http://ieeexplore.ieee.org/document/7942934/`, `doi:10.1109/EDUCON.2017.7942934`.

[12] Llanos Tobarra, Antonio Perez Trapero, Rafael Pastor, Antonio Robles-Gomez, Roberto Hernandez, Andres Duque, and Jesus Cano. Game-based Learning Approach to Cybersecurity. In *2020 IEEE Global Engineering Education Conference (EDUCON)*, pages 1125–1132, Porto, Portugal, April 2020. IEEE. URL: `https://ieeexplore.ieee.org/document/9125202/`, `doi:10.1109/EDUCON45650.2020.9125202`.

[13] Valdemar Švábenský, Pavel Čeleda, Jan Vykopal, and Silvia Brišáková. Cybersecurity knowledge and skills taught in capture the flag challenges. *Computers & Security*, 102:102154, March 2021. URL: `https://linkinghub.elsevier.com/retrieve/pii/S0167404820304272`, `doi:10.1016/j.cose.2020.102154`.

[14] Kevin Chung. Lowering the Barriers to Capture The Flag Administration and Participation. page 6, 2017.

[15] Thomas A Augustine, Lori L DeLooze, Justin C Monroe, and Christopher G Wheeler. Cyber competitions as a computer science recruiting tool. page 8, 2010.

[16] Ann-Marie Horcher. Shall We Play a Game? : Building Capture the Flag Games for Non-Traditional Players. In *2020 Research on Equity and Sustained Participation in Engineering, Computing, and Technology (RESPECT)*, pages 1–2, Portland, OR, USA, March 2020. IEEE. URL: `https://ieeexplore.ieee.org/document/9272410/`, `doi:10.1109/RESPECT49803.2020.9272410`.

[17] European Union Agency for Cybersecurity. *CTF events: contemporary practices and state of the art in capture the flag competitions.* Publications Office, LU, 2021. URL: `https://data.europa.eu/doi/10.2824/313553`.

[18] Alastair Janse van Rensburg and Richard Baker. CTF events: contemporary practices and state of the art in capture the flag competitions. Technical report, European Union Agency for Cybersecurity - Publications Office, LU, 2021. URL: `https://data.europa.eu/doi/10.2824/313553`.

[19] CVE - CVE, 2021. URL: `https://cve.mitre.org/`.

[20] Dan Borges. *Adversarial Tradecraft in Cybersecurity.* Packt, 1 edition, June 2021.

[21] Best response, August 2021. Page Version ID: 1039962487. URL: `https://en.wikipedia.org/w/index.php?title=Best_response&oldid=1039962487`.

[22] Erich Prisner. *Game Theory through Examples.* American Mathematical Society, Providence, Rhode Island, 2014. URL: `https://www.ams.org/clrm/046`, `doi:10.5948/9781614441151`.

[23] Michael Collins. *Network Security Through Data Analysis Building Situational Awareness Michael Collins.* O'Reilly, 1 edition, 2014.

[24] J. Quittek, S. Bryant, B. Claise, P. Aitken, NEC, Inc. Cisco Systems, J. Meyer, and Paypal. Information Model for IP Flow Information Export, 2008. URL: `https://www.ietf.org/rfc/rfc5102.txt`.

[25] Stylianos Karagiannis and Emmanouil Magkos. Adapting CTF challenges into virtual cybersecurity learning environments. *Information & Computer Security*, 29(1):105–132, May 2021. URL: `https://www.emerald.com/insight/content/doi/10.1108/ICS-04-2019-0050/full/html`, `doi:10.1108/ICS-04-2019-0050`.

[26] Tom Goodman and Andreea Ina Radu. Learn-Apply-Reinforce/Share Learning: Hackathons and CTFs as General Pedagogic Tools in Higher Education, and Their Applicability to Distance Learning (A Preprint). 2020. Publisher: Unpublished. URL: `http://rgdoi.net/10.13140/RG.2.2.21810.43205`, `doi:10.13140/RG.2.2.21810.43205`.

[27] Stylianos Karagiannis, Elpidoforos Maragkos-Belmpas, and Emmanouil Magkos. An Analysis and Evaluation of Open Source Capture the Flag Platforms as Cybersecurity e-Learning Tools. In Lynette Drevin, Suné Von Solms, and Marianthi Theocharidou, editors, *Information Security Education. Information Security in Action*, volume 579, pages 61–77. Springer International Publishing, Cham, 2020. Series Title: IFIP Advances in Information and Communication Technology. URL: `https://link.springer.com/10.1007/978-3-030-59291-2_5`, `doi:10.1007/978-3-030-59291-2_5`.

[28] Stela Kucek and Maria Leitner. An Empirical Survey of Functions and Configurations of Open-Source Capture the Flag (CTF) Environments. *Journal of Network and Computer Applications*, 151:102470, February 2020. URL: `https://linkinghub.elsevier.com/retrieve/pii/S1084804519303303`, `doi:10.1016/j.jnca.2019.102470`.

[29] Erik Trickel, Francesco Disperati, Eric Gustafson, Faezeh Kalantari, Mike Mabey, Naveen Tiwari, Yeganeh Safaei, Adam Doupe, and Giovanni Vigna. Shell We Play A Game? CTF-as-a-service for Security Education. page 11, August 2017.

[30] Clark Taylor, Pablo Arias, Jim Klopchic, Celeste Matarazzo, and Evi Dube. CTF: State-of-the-Art and Building the Next Generation. page 11, 2017.

[31] Masooda Bashir, Colin Wee, Nasir Memon, and Boyi Guo. Profiling cybersecurity competition participants: Self-efficacy, decision-making and interests predict effectiveness of competitions as a recruitment tool. *Computers & Security*, 65:153–165, March 2017. URL: `https://linkinghub.elsevier.com/retrieve/pii/S0167404816301389`, `doi:10.1016/j.cose.2016.10.007`.

[32] Tanner J Burns, Samuel C Rios, Thomas K Jordan, Qijun Gu, and Trevor Underwood. Analysis and Exercises for Engaging Beginners in Online CTF Competitions for Security Education. page 9, 2016.

[33] Arvind S Raj, Bithin Alangot, Seshagiri Prabhu, and Krishnashree Achuthan. Scalable and lightweight CTF infrastructures using application containers. page 9, 2016.

[34] Eric Nunes, Nimish Kulkarni, Paulo Shakarian, Andrew Ruef, and Jay Little. Cyber-Deception and Attribution in Capture-the-Flag Exercises. *arXiv:1507.01922 [cs]*, July 2015. arXiv: 1507.01922. URL: `http://arxiv.org/abs/1507.01922`.

[35] Eric Nunes, Paulo Shakarian, Gerardo I. Simari, and Andrew Ruef. Argumentation Models for Cyber Attribution. *arXiv:1607.02171 [cs]*, July 2016. arXiv: 1607.02171. URL: `http://arxiv.org/abs/1607.02171`.

[36] Jonathan Burket, Peter Chapman, and Tim Becker. Automatic Problem Generation for Capture-the-Flag Competitions. page 9, 2015.

[37] Jason A. Donenfeld. WireGuard: fast, modern, secure VPN tunnel, 2021. URL: `https://www.wireguard.com/`.

[38] The Dark Tangent. media.defcon.org, 2021. URL: `https://media.defcon.org/`.

[39] Wireshark · Go Deep., 2021. URL: `https://www.wireshark.org/`.

[40] tshark - The Wireshark Network Analyzer 3.4.9, 2021. URL: `https://www.wireshark.org/docs/man-pages/tshark.html`.

[41] CapLoader - Handles Big Data PCAP files, 2021. URL: `https://www.netresec.com/?page=CapLoader`.

[42] Oded Shimon. About, November 2021. original-date: 2020-02-16T20:58:59Z. URL: `https://github.com/odedshimon/BruteShark`.

[43] YAF, 2020. URL: `https://tools.netsa.cert.org/yaf/`.

[44] B. Claise, B. Trammell, and P. Aitken. Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information. Technical Report RFC7011, RFC Editor, September 2013. URL: `https://www.rfc-editor.org/info/rfc7011`, `doi:10.17487/rfc7011`.

[45] Sam Kumarsamy. IP Flow Information Export - IPFIX vs. NetFlow, September 2019. URL: `https://blog.gigamon.com/2019/09/17/ipfix-vs-netflow/`.

[46] David Fifield. A better zip bomb, 2019. URL: https://www.bamsoftware.com/hacks/zipbomb/.

[47] Order of the Overflow. DC 28 CTF Postmortem, 2021. URL: https://oooverflow.io/dc-ctf-2020-finals/postmortem.html.

[48] Melvin L. SharpCollection, November 2021. original-date: 2020-06-05T12:50:00Z. URL: https://github.com/Flangvik/SharpCollection.

[49] sqlmap, November 2021. original-date: 2012-06-26T09:52:15Z. URL: https://github.com/sqlmapproject/sqlmap.

[50] Metasploit, November 2021. original-date: 2011-08-30T06:13:20Z. URL: https://github.com/rapid7/metasploit-framework.

[51] Empire, November 2021. original-date: 2019-08-01T04:22:31Z. URL: https://github.com/BC-SECURITY/Empire.

[52] Chirag Savla. Obfuscator, October 2021. original-date: 2020-08-09T10:14:32Z. URL: https://github.com/3xpl01tc0d3r/Obfuscator.

[53] midnite_runr. secretsquirrel/the-backdoor-factory, November 2021. original-date: 2013-05-30T01:04:24Z. URL: https://github.com/secretsquirrel/the-backdoor-factory.

[54] Andrea Fabrizi. Prism backdoor, October 2021. original-date: 2013-04-18T09:23:27Z. URL: https://github.com/andreafabrizi/prism.

[55] Jeroen van Kessel. icmpdoor - ICMP Reverse Shell, November 2021. original-date: 2020-11-11T22:32:06Z. URL: https://github.com/krabelize/icmpdoor.

[56] unhide(8): find hidden processes - Linux man page. URL: https://linux.die.net/man/8/unhide.

[57] Code Express. res·pound·er, November 2021. original-date: 2018-02-05T04:42:14Z. URL: https://github.com/codeexpress/respounder.

[58] Binary Defense Systems (BDS) is a sister company of TrustedSec, LLC, November 2021. original-date: 2015-04-10T17:18:44Z. URL: https://github.com/BinaryDefense/artillery.

[59] Jake Smith. BLUESPAWN, November 2021. original-date: 2019-05-28T22:41:36Z. URL: https://github.com/ION28/BLUESPAWN.

[60] 42.zip. URL: https://www.unforgettable.dk/.

[61] Zip bomb, October 2021. Page Version ID: 1050143384. URL: https://en.wikipedia.org/w/index.php?title=Zip_bomb&oldid=1050143384.

[62] Portspoof - A new approach to fight back port and service scanners. URL: https://drk1wi.github.io/portspoof/.

[63] Nikhil "SamratAshok" Mittal. Deploy-Deception, October 2021. original-date: 2018-10-16T17:33:15Z. URL: https://github.com/samratashok/Deploy-Deception.

[64] Thinkst Applied Research. Know. Before it matters. URL: `https://canarytokens.org`.

[65] TL;DR, November 2021. original-date: 2014-11-28T16:57:47Z. URL: `https://github.com/telekom-security/tpotce`.

[66] Hirato Kirata. Hirato/LaBrea, October 2021. original-date: 2012-02-28T13:38:57Z. URL: `https://github.com/Hirato/LaBrea`.

[67] IDA Pro – Hex Rays. URL: `https://hex-rays.com/ida-pro/`.

[68] GDB: The GNU Project Debugger. URL: `https://www.gnu.org/software/gdb/`.

[69] strace(1) - Linux manual page. URL: `https://man7.org/linux/man-pages/man1/strace.1.html`.

[70] strace, October 2021. Page Version ID: 1048904011. URL: `https://en.wikipedia.org/w/index.php?title=Strace&oldid=1048904011`.

[71] Binwalk, November 2021. original-date: 2013-11-15T20:45:40Z. URL: `https://github.com/ReFirmLabs/binwalk`.

[72] CyberChef, November 2021. original-date: 2016-11-28T10:34:07Z. URL: `https://github.com/gchq/CyberChef`.

[73] CyberChef. URL: `https://gchq.github.io/CyberChef/`.

[74] RocketChat/Rocket.Chat, November 2021. original-date: 2015-05-19T07:36:09Z. URL: `https://github.com/RocketChat/Rocket.Chat`.

[75] Discord | Your Place to Talk and Hang Out. URL: `https://discord.com/`.

[76] A real-time collaborative editor for the web, November 2021. original-date: 2011-03-26T13:09:02Z. URL: `https://github.com/ether/etherpad-lite`.

[77] Labs | Offensive Security, 2021. URL: `https://www.offensive-security.com/labs/`.

[78] Vulnerable By Design ~ VulnHub, 2021. URL: `https://www.vulnhub.com/`.

[79] Hacking Training For The Best, 2021. URL: `https://www.hackthebox.eu/`.

[80] TryHackMe | Cyber Security Training, 2021. URL: `https://tryhackme.com`.

[81] Hack The Box :: Battlegrounds, 2021. URL: `https://app.hackthebox.com/battlegrounds/lobby`.

[82] Cloud Native Computing Foundation. Production-Grade Container Orchestration, 2021. URL: `https://kubernetes.io/`.

[83] Wye Kede Jerel Yam. Strategies used in capture-the-flag events contributing to team performance. page 116.

[84] Wye Kede Jerel Yam. Strategies used in capture-the-flag events contributing to team performance. page 116.

[85] E. Wes Bethel, Scott Campbell, Eli Dart, Kurt Stockinger, and Kesheng Wu. Accelerating Network Traffic Analytics Using Query-Driven Visualization. In *2006 IEEE Symposium On Visual Analytics Science And Technology*, pages 115–122, Baltimore, MD, October 2006. IEEE. URL: `https://ieeexplore.ieee.org/document/4035755/`, `doi:10.1109/VAST.2006.261437`.

[86] Mark Thomas, Leigh Metcalf, Jonathan Spring, Paul Krystosek, and Katherine Prevost. SiLK: A Tool Suite for Unsampled Network Flow Analysis at Scale. In *2014 IEEE International Congress on Big Data*, pages 184–191, Anchorage, AK, USA, June 2014. IEEE. URL: `http://ieeexplore.ieee.org/document/6906777/`, `doi:10.1109/BigData.Congress.2014.34`.

[87] Jason Smith. FlowPlotter, July 2021. original-date: 2014-01-22T19:27:07Z. URL: `https://github.com/automayt/FlowPlotter`.

[88] Chris Sanders. FlowBAT, May 2021. original-date: 2014-07-23T19:43:46Z. URL: `https://github.com/chrissanders/FlowBAT`.

[89] Applied Detection and Analysis Using Flow Data - (BSides Nashville 2015) (Hacking Illustrated Series InfoSec Tutorial Videos). URL: `http://www.irongeek.com/i.php?page=videos/bsidesnashville2015/b00-applied-detection-and-analysis-using-flow-data-jason-a-smith`.

[90] Jason M. Pittman. Does Competitor Grade Level Influence Perception of Cybersecurity Competition Design Gender Inclusiveness? In *Proceedings of the 2015 ACM SIGMIS Conference on Computers and People Research*, pages 49–54, Newport Beach California USA, June 2015. ACM. URL: `https://dl.acm.org/doi/10.1145/2751957.2751974`, `doi:10.1145/2751957.2751974`.

[91] Massimo Ficco and Francesco Palmieri. Leaf: An open-source cybersecurity training platform for realistic edge-IoT scenarios. *Journal of Systems Architecture*, 97:107–129, August 2019. URL: `https://linkinghub.elsevier.com/retrieve/pii/S1383762118304442`, `doi:10.1016/j.sysarc.2019.04.004`.

[92] OOO — DEF CON CTF, 2021. URL: `https://oooverflow.io/dc-ctf-2019-finals/`.

[93] OOO — DEF CON CTF. URL: `https://oooverflow.io/dc-ctf-2020-finals/`.

[94] Yan Shoshitaishvili. The Order of the Overflow. page 17.

[95] The Dark Tangent. media.defcon.org. URL: `https://media.defcon.org/`.

[96] DEF CON CTF 2020 QUALS. URL: https://oooverflow.io/dc-ctf-2020-quals/.

[97] OOO — DEF CON CTF. URL: https://oooverflow.io/dc-ctf-2020-finals/.

[98] Tom Goodman and Andreea Ina Radu. Learn-Apply-Reinforce/Share Learning: Hackathons and CTFs as General Pedagogic Tools in Higher Education, and Their Applicability to Distance Learning (A Preprint). 2020. Publisher: Unpublished. URL: http://rgdoi.net/10.13140/RG.2.2.21810.43205, doi:10.13140/RG.2.2.21810.43205.

[99] Raghu Raman, Sherin Sunny, Vipin Pavithran, and Krishnasree Achuthan. Framework for evaluating Capture the Flag (CTF) security competitions. In *International Conference for Convergence for Technology-2014*, pages 1–5, Pune, India, April 2014. IEEE. URL: http://ieeexplore.ieee.org/document/7092098/, doi:10.1109/I2CT.2014.7092098.

[100] Muhammad Mudassar Yamin, Basel Katt, and Mariusz Nowostawski. Serious games as a tool to model attack and defense scenarios for cyber-security exercises. *Computers & Security*, 110:102450, November 2021. URL: https://linkinghub.elsevier.com/retrieve/pii/S0167404821002741, doi:10.1016/j.cose.2021.102450.

[101] Valdemar Švábenský, Pavel Čeleda, Jan Vykopal, and Silvia Brišáková. Cybersecurity knowledge and skills taught in capture the flag challenges. *Computers & Security*, 102:102154, March 2021. URL: https://linkinghub.elsevier.com/retrieve/pii/S0167404820304272, doi:10.1016/j.cose.2020.102154.

[102] Zackary Crosley. Inductive Programming Techniques for Network Traffic Comprehension and Reflection. page 58.

[103] Kees Leune and Salvatore J. Petrilli. Using Capture-the-Flag to Enhance the Effectiveness of Cybersecurity Education. In *Proceedings of the 18th Annual Conference on Information Technology Education*, pages 47–52, Rochester New York USA, September 2017. ACM. URL: https://dl.acm.org/doi/10.1145/3125659.3125686, doi:10.1145/3125659.3125686.

[104] David H. Tobey, Portia Pusey, and Josh Chin. Cybersecurity Competitions in Education: Engaging Learners through Improved Game Balance. In *Proceedings of the 2015 ACM SIGMIS Conference on Computers and People Research*, pages 99–100, Newport Beach California USA, June 2015. ACM. URL: https://dl.acm.org/doi/10.1145/2751957.2751969, doi:10.1145/2751957.2751969.

[105] European Union Agency for Cybersecurity. *CTF events: contemporary practices and state of the art in capture the flag competitions.* Publications Office, LU, 2021. URL: https://data.europa.eu/doi/10.2824/313553.

[106] Eric Nunes, Nimish Kulkarni, Paulo Shakarian, Andrew Ruef, and Jay Little. Cyber-Deception and Attribution in Capture-the-Flag Exercises. *arXiv:1507.01922 [cs]*, July 2015. arXiv: 1507.01922. URL: `http://arxiv.org/abs/1507.01922`.

[107] C. Cowan, S. Arnold, S. Beattie, C. Wright, and J. Viega. Defcon Capture the Flag: defending vulnerable code from intense attack. In *Proceedings DARPA Information Survivability Conference and Exposition*, pages 120–129, Washington, DC, USA, 2003. IEEE Comput. Soc. URL: `http://ieeexplore.ieee.org/document/1194878/`, `doi:10.1109/DISCEX.2003.1194878`.

[108] Erich Prisner. *Game Theory through Examples*. American Mathematical Society, Providence, Rhode Island, 2014. URL: `https://www.ams.org/clrm/046`, `doi:10.5948/9781614441151`.

[109] E. Wes Bethel, Scott Campbell, Eli Dart, Kurt Stockinger, and Kesheng Wu. Accelerating Network Traffic Analytics Using Query-Driven Visualization. In *2006 IEEE Symposium On Visual Analytics Science And Technology*, pages 115–122, Baltimore, MD, October 2006. IEEE. URL: `https://ieeexplore.ieee.org/document/4035755/`, `doi:10.1109/VAST.2006.261437`.

[110] Qijun Gu. An Analysis of Security Competitions for a Beginner's Guide. (1):22, 2017.

[111] Kevin Chung. Lowering the Barriers to Capture The Flag Administration and Participation. page 6.

[112] Menelaos Katsantonis, Panayotis Fouliras, and Ioannis Mavridis. Conceptual analysis of cyber security education based on live competitions. In *2017 IEEE Global Engineering Education Conference (EDUCON)*, pages 771–779, Athens, Greece, April 2017. IEEE. URL: `http://ieeexplore.ieee.org/document/7942934/`, `doi:10.1109/EDUCON.2017.7942934`.

[113] Valdemar Švábenský, Pavel Čeleda, Jan Vykopal, and Silvia Brišáková. Cybersecurity Knowledge and Skills Taught in Capture the Flag Challenges. *Computers & Security*, 102:102154, March 2021. arXiv: 2101.01421. URL: `http://arxiv.org/abs/2101.01421`, `doi:10.1016/j.cose.2020.102154`.

[114] Valdemar Švábenský, Pavel Čeleda, Jan Vykopal, and Silvia Brišáková. Cybersecurity knowledge and skills taught in capture the flag challenges. *Computers & Security*, 102:102154, March 2021. URL: `https://linkinghub.elsevier.com/retrieve/pii/S0167404820304272`, `doi:10.1016/j.cose.2020.102154`.

[115] European Union Agency for Cybersecurity. *CTF events: contemporary practices and state of the art in capture the flag competitions*. Publications Office, LU, 2021. URL: `https://data.europa.eu/doi/10.2824/313553`.

[116] Llanos Tobarra, Antonio Perez Trapero, Rafael Pastor, Antonio Robles-Gomez, Roberto Hernandez, Andres Duque, and Jesus Cano. Game-based

Learning Approach to Cybersecurity. In *2020 IEEE Global Engineering Education Conference (EDUCON)*, pages 1125–1132, Porto, Portugal, April 2020. IEEE. URL: `https://ieeexplore.ieee.org/document/9125202/`, `doi:10.1109/EDUCON45650.2020.9125202`.

[117] Massimo Ficco and Francesco Palmieri. Leaf: An open-source cybersecurity training platform for realistic edge-IoT scenarios. *Journal of Systems Architecture*, 97:107–129, August 2019. URL: `https://linkinghub.elsevier.com/retrieve/pii/S1383762118304442`, `doi:10.1016/j.sysarc.2019.04.004`.

[118] Ann-Marie Horcher. Shall We Play a Game? : Building Capture the Flag Games for Non-Traditional Players. In *2020 Research on Equity and Sustained Participation in Engineering, Computing, and Technology (RESPECT)*, pages 1–2, Portland, OR, USA, March 2020. IEEE. URL: `https://ieeexplore.ieee.org/document/9272410/`, `doi:10.1109/RESPECT49803.2020.9272410`.

[119] Erik Trickel, Francesco Disperati, Eric Gustafson, Faezeh Kalantari, Mike Mabey, Naveen Tiwari, Yeganeh Safaei, Adam Doupe, and Giovanni Vigna. Shell We Play A Game? CTF-as-a-service for Security Education. page 11.

[120] Mark Thomas, Leigh Metcalf, Jonathan Spring, Paul Krystosek, and Katherine Prevost. SiLK: A Tool Suite for Unsampled Network Flow Analysis at Scale. In *2014 IEEE International Congress on Big Data*, pages 184–191, Anchorage, AK, USA, June 2014. IEEE. URL: `http://ieeexplore.ieee.org/document/6906777/`, `doi:10.1109/BigData.Congress.2014.34`.

[121] Wye Kede Jerel Yam. Strategies used in capture-the-flag events contributing to team performance. page 116.

[122] Peng Liu, Wanyu Zang, and Meng Yu. Incentive-Based Modeling and Inference of Attacker Intent, Objectives, and Strategies. *ACM Transactions on Information and System Security*, 8(1):41.

[123] Jan Vykopal, Valdemar Švábenský, and Ee-Chien Chang. Benefits and Pitfalls of Using Capture the Flag Games in University Courses. *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, pages 752–758, February 2020. arXiv: 2004.11556. URL: `http://arxiv.org/abs/2004.11556`, `doi:10.1145/3328778.3366893`.

[124] Artem Garkusha, Evgeny Abramov, and Oleg Makarevich. An effectiveness of application of <> competitions concept within higher education (Student contribution). In *Proceedings of the 7th International Conference on Security of Information and Networks - SIN '14*, pages 43–46, Glasgow, Scotland, UK, 2014. ACM Press. URL: `http://dl.acm.org/citation.cfm?doid=2659651.2659732`, `doi:10.1145/2659651.2659732`.

[125] Mark Thomas, Leigh Metcalf, Jonathan Spring, Paul Krystosek, and Katherine Prevost. SiLK: A Tool Suite for Unsampled Network Flow Analysis at Scale. In *2014 IEEE International Congress on Big Data*,

pages 184–191, Anchorage, AK, USA, June 2014. IEEE. URL: http://ieeexplore.ieee.org/document/6906777/, doi:10.1109/BigData.Congress.2014.34.

[126] Valdemar Švábenský, Jan Vykopal, Pavel Seda, and Pavel Čeleda. Dataset of shell commands used by participants of hands-on cybersecurity training. *Data in Brief*, 38:107398, October 2021. URL: https://linkinghub.elsevier.com/retrieve/pii/S2352340921006806, doi:10.1016/j.dib.2021.107398.

[127] Michael Sentonas. Council Post: Ransomware: Double The Trouble In 2021. Section: Innovation. URL: https://www.forbes.com/sites/forbestechcouncil/2021/09/24/ransomware-double-the-trouble-in-2021/.

[128] IP Flow Information Export (ipfix) -. URL: https://datatracker.ietf.org/wg/ipfix/about/.

[129] rfc7011, 2013. URL: https://datatracker.ietf.org/doc/html/rfc7011.

[130] PentesterLab: Learn Web Penetration Testing: The Right Way, 2021. URL: https://www.pentesterlab.com/.

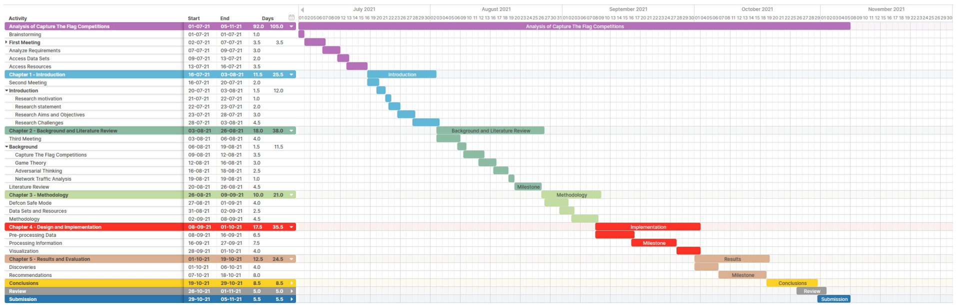
[131] Hack The Box :: Battlegrounds. URL: https://app.hackthebox.com/battlegrounds/lobby.

[132] midnite_runr. secretsquirrel/the-backdoor-factory, November 2021. original-date: 2013-05-30T01:04:24Z. URL: https://github.com/secretsquirrel/the-backdoor-factory.

[133] Hirato Kirata. Hirato/LaBrea, October 2021. original-date: 2012-02-28T13:38:57Z. URL: https://github.com/Hirato/LaBrea.

Appendix A

Project Management Gantt Chart



Activity	Start	End	Days	
Analysis of Capture The Flag Competitions	01-07-21	05-11-21	92.0	105.0
Brainstorming	01-07-21	01-07-21	1.0	
▸ First Meeting	02-07-21	07-07-21	3.5	3.5
Analyze Requirements	07-07-21	09-07-21	3.0	
Access Data Sets	09-07-21	13-07-21	2.0	
Access Resources	13-07-21	16-07-21	3.5	
Chapter 1 - Introduction	16-07-21	03-08-21	11.5	25.5
Second Meeting	16-07-21	20-07-21	2.0	
▾ Introduction	20-07-21	03-08-21	1.5	12.0
Research motivation	21-07-21	22-07-21	1.0	
Research statement	22-07-21	23-07-21	2.0	
Research Aims and Objectives	23-07-21	28-07-21	3.0	
Research Challenges	28-07-21	03-08-21	4.5	
Chapter 2 - Background and Literature Review	03-08-21	26-08-21	18.0	38.0
Third Meeting	03-08-21	06-08-21	4.0	
▾ Background	06-08-21	19-08-21	1.5	11.5
Capture The Flag Competitions	09-08-21	12-08-21	3.5	
Game Theory	12-08-21	16-08-21	3.0	
Adversarial Thinking	16-08-21	18-08-21	2.5	
Network Traffic Analysis	19-08-21	19-08-21	1.0	
Literature Review	20-08-21	26-08-21	4.5	
Chapter 3 - Methodology	26-08-21	09-09-21	10.0	21.0
Defcon Safe Mode	27-08-21	01-09-21	4.0	
Data Sets and Resources	31-08-21	02-09-21	2.5	
Methodology	02-09-21	08-09-21	4.5	
Chapter 4 - Design and Implementation	08-09-21	01-10-21	17.5	35.5
Pre-processing Data	08-09-21	16-09-21	6.5	
Processing Information	16-09-21	27-09-21	7.5	
Visualization	28-09-21	01-10-21	4.0	
Chapter 5 - Results and Evaluation	01-10-21	19-10-21	12.5	24.5
Discoveries	01-10-21	06-10-21	4.0	
Recommendations	07-10-21	18-10-21	8.0	
Conclusions	19-10-21	29-10-21	8.5	8.5
Review	26-10-21	01-11-21	5.0	5.0
Submission	29-10-21	05-11-21	5.5	5.5

Figure A.1: Gantt Chart