jamk

Examination of correlation between education, role and type of solutions used to solve cyber-security challenges

Harri Hietaranta

Master’s thesis
November 2022
Master's Degree Programme in Information Technology, Cyber Security

jamk | Jyväskylän ammattikorkeakoulu
University of Applied Sciences

jamk

Description

Harri Hietaranta

Examination of correlation between education, role and type of solutions used to solve cyber-security challenges

Jyväskylä: JAMK University of Applied Sciences, November 2022, 115 pages

Technology, Master's Degree Programme in Information Technology, Cyber Security

Permission for web publication: Yes

Language of publication: English

Abstract

A cyber security challenge was constructed and hosted for Gofore Plc, which is a company enabling hosting of communality events for employees. The purpose of the challenge was to get data on cyber-security skills of employees while providing them with a fun event and an educational challenge while most of the employees were working remotely.

A Capture the Flag challenge was constructed in an AWS environment for the participants to solve. The challenge consisted of several different types of AWS components to provide intentionally vulnerable or badly configured websites, web-applications, and servers. Each step of the challenge contained a flag in format of a hash, which could be sent to a Slack Application bot built for this challenge, to mark the progress of the participant.

Participants were given hints inside the applications and servers, pointing them to the next location to try to exploit. The individual puzzles inside the challenge environment varied in difficulty and technicality to provide the participants with different paths to solve and exploit.
When announcing the challenge, a training event was held as a remote meeting introducing the basic tools and techniques needed to solve the individual challenges. It was not told which tools are needed for which puzzle.

After the playtime for the challenge was over, a playthrough event was held as a remote meeting, where the solutions were demonstrated. All participants were requested to fill out a questionnaire about the challenge, where they provided information about their educational background, current role and which individual challenges they solved.

Correlation between education and solutions used and between current role and solutions used was observed with the assumption of employees with technical role or education would prefer more technical solutions. Because of the small number of participants and underrepresentation of lower educational levels and non-technical occupational roles, no real conclusions could be drawn. Inside the small group, non-technical participants, and participants with lower level of education fared well and defied the hypothesis.

Keywords/tags (subjects)

Cyber security training, Capture the Flag, Gamification

Miscellaneous (Confidential information)

jamk

Description

Hietaranta, Harri

Työnkuvan, koulutuksen ja kyberturvallisuusharjoituksessa käytettyjen ratkaisun välisen korrelaation tutkiminen

Jyväskylä: Jyväskylän ammattikorkeakoulu. Marraskuu 2022, 115 sivua

Technology, Master's Degree Programme in Information Technology, Cyber Security

Julkaisun kieli: Englanti

Julkaisulupa avoimessa verkossa: kyllä

Tiivistelmä

Tietoturvahaaste toteutettiin yhteistyössä Gofore Oyj:n kanssa, joka mahdollistaa yhteisöllisten tapahtumien järjestämisen työntekijöilleen. Haasteen tarkoituksena oli kerätä tietoa osallistujien kyberturvallisuustaidoista samalla järjestäen työntekijöille hauska tapahtuma ja opettavainen haaste etätyösuosituksen aikana.

Haaste rakennettiin AWS ympäristöön osallistujien ratkaistavaksi. Haaste rakentui useista eri AWS-komponenteista, joiden avulla saatiin tarjottua tarkoituksella haavoittuvaisia tai huonosti konfiguroituja verkkosivuja, verkkosovelluksia ja palvelimia. Haasteen jokainen vaihe sisälsi tiivistemuotoisen lipun, jonka saattoi palauttaa Slack-sovelluksena toimivalle botille osallistujan edistymisen merkkaamiseksi. Osallistujille tarjottiin vihjeitä seuraavasta hyökkäyskohteesta sovellusten ja palvelinten sisällä. Haasteen eri pulmat vaihtelivat haastavuudeltaan ja teknisyydeltään tarjoten osallistujille vaihtoehtoisia ratkaisuja ja polkuja.

Haasteen julkistamisen yhteydessä pidettiin koulutustapahtuma etätapaamisena, jossa esiteltiin perus työkalut ja tekniikat, joilla haasteen pulmat saadaan ratkaistua. Osallistujille ei paljastettu mitä työkalua tarvitaan mihinkäkin pulmaan.

Haasteen jälkeen, osallistujille järjestettiin läpipeelutapahtuma etäkokouksena, jossa pulmien ratkaisut esiteltiin. Kaikkia osallistujia pyydettiin täyttämään kyselylomake haasteesta, jossa heitä pyydettiin vastaamaan kysymyksiin koulutuksestaan, nykyisestä työtehtävästään ja siitä mitkä pulmat saivat ratkaistua. Sekä koulutuksen, että nykyisen työtehtävän vaikutusta ratkaistuihin pulmiin sekä haasteen läpipääsyyn tutkittiin. Oletuksena oli, että korkeamman koulutustason tai teknisemmän työnkuvan omaavat pärjäisivät haasteessa paremmin ja valitsisivat teknisempiä ratkaisuja.

Vastausten pienestä lukumäärästä, ei-teknisen työtehtävän tai matalan koulutustason omaavien aliedustuksesta johtuen, kerätyistä tiedoista ei voitu tehdä johtopäätöksiä. Pienen ryhmän sisällä sekä ei-teknisen työnkuvan omaavat, että matalamman koulutustason omaavat pärjäsivät hyvin ja uhmasivat hypoteesia.

Avainsanat (asiasanat)

Kyberturvallisuuskoulutus, CTF, Lipunryöstö, Pelillistäminen

Muut tiedot (salassa pidettävät liitteet)

Contents

Abbreviations **8**
1 Introduction **10**
2 Theory **11**
2.1 Gamified learning 11
2.2 Penetration testing and red teaming 13
2.3 Capture The Flag challenges 14
2.3.1 Capture the Flag service providers 17
2.3.2 Capture the Flag events 17
2.4 Infrastructure technologies and techniques 18
2.5 Challenge solving techniques 20
2.5.1 Web and communications related techniques 20
2.5.2 Local techniques 27
3 Methodology **30**
3.1 Initial hypothesis and research questions 31
3.2 Research methods 32
4 Building the challenge **34**
4.1 CTF challenge design and build process 35
4.2 Infrastructure for the challenge 39
4.3 The challenge puzzles 42
4.3.1 Path 1 44
4.3.2 Path 2 51
4.3.3 Common path 52
4.4 Training session 53
5 Results **56**
5.1 Respondents' statistics 58
5.2 Respondents' subjective answers 64
5.3 Puzzle solving results 66
5.3.1 Path 1 puzzle questionnaire answers 68
5.3.2 Path 2 puzzle questionnaire answers 77

5.3.3 Common path puzzle questionnaire answers 84
6 Conclusions 91
7 Discussions 94
References 96
Appendices 104
Appendix 1. Questionnaire 104
Appendix 2. Questionnaire answers 107
Appendix 3. Questionaire answer summary 108
Appendix 4. Source code of Slackbot lambda 112
Appendix 5. Training presentation slide 2 115

Figures

Figure 1. URL structure example 22
Figure 2. Example usage of fuzzing tool ffuf 23
Figure 3. Example line of /etc/passwd file 26
Figure 4. Example of /etc/shadow 27
Figure 5. CTF design and build process flow 35
Figure 6. CTF infrastucture topology diagram 39
Figure 7. Example of a message sent to Slackbot via Slack 42
Figure 8. Possible solution paths of the challenge 43
Figure 9. Obfuscated JavaScript 44
Figure 10. De-obfuscated JavaScript 45
Figure 11. Screen caption from inside the exploitable service 45
Figure 12. The Github repository 46
Figure 13. One example of using steganography to hide data 47
Figure 14. Github repostitory commit history view 48
Figure 15. SSH RSA-key in commit history 49
Figure 16. Decompiled Java sourcecode 50
Figure 17. Inspection of network requests on website loading event 51
Figure 18. Ransom note 54

Figure 19. Challenge completion numbers 57
Figure 20. Level of education of questionnaire respondents 58
Figure 21. Challenge completion status by educational background 59
Figure 22. Respondent numbers by occupational role 60
Figure 23. Occupational titles by educational level 61
Figure 24. Respondents' Previous experience in CTFs 62
Figure 25. Challenge completion and training session attendance by previous CTF experience 63
Figure 26. Challenge difficulty assesment by occupational role 64
Figure 27. Challenge difficulty assesment by challenge completion status 65
Figure 28. Number of solvers per challenge 66
Figure 29. Solvers in Sepposorsa puzzle questions by occupational role 67
Figure 30. Deobfuscated the javascript solvers by occupational role 68
Figure 31. Deobfuscated the javascript solvers by educational background 69
Figure 32. SSH Key in Github repository solvers by occupational role 70
Figure 33. SSH Key in Github repository solvers by educational background 71
Figure 34. Cracked the SSH key password with John or similar software solvers by occupational role 72
Figure 35. Cracked the SSH key password with John or similar software solvers by educational background 73
Figure 36. Found the SSH key password hidden in an image solvers by occupational role 74
Figure 37. Found the SSH key password hidden in an image solvers by educational background 75
Figure 38. Reverse engineered the Java application solvers by occupational role 76
Figure 39. Reverse engineered the Java application solvers by educational background 77
Figure 40. The image url leading to another server solvers by occupational role 78
Figure 41. The image url leading to another server solvers by educational background 79
Figure 42. Anonymous login FTP service solvers by occupational role 80
Figure 43. Anonymous login FTP service solvers by educational background 81
Figure 44. Reverse engineered the ELF binary solvers by occupational role 82
Figure 45. Reverse engineered the ELF binary solvers by educational background 83
Figure 46. Found password hash file by fuzzing solvers by occupational role 84
Figure 47. Found password hash file by fuzzing solvers by educational background 85

Figure 48. Cracked Overlord password hash solvers by occupational role 86
Figure 49. Cracked Overlord password hash solvers by educational background 87
Figure 50. Escaped rbash solvers by occupational role 88
Figure 51. Escaped rbash solvers by educational background 89
Figure 52. Found the leaked data solvers combined occupational roles and education 90

Tables

Table 1. Numerical statistics of participants 56

Abbreviations

APA	American Psychological Association
API	Application Programming Interface
APP	Application
ASCII	American Standard Code for Information Interchange
AWS	Amazon Web Services
CDN	Content Delivery Network
CSS	Cascading Stylesheet
CTF	Capture the Flag
CVE	Common Vulnerabilities and Exposures
DB	Database
DNS	Domain Name Service
EC	Elastic Computer Cloud
ELF	Executable and Linkable Format
FTP	File Transfer Protocol
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
ID	Identifier
IDE	Integrated Development Environment
IP	Internet Protocol
ISCED	International standard classification of education
IT	Information Technology
Jamk	Jyväskylä University of Applied Sciences
LSB	Least Significant Bit
MDN	Multiple Domain Network
MITM	Man in The Middle
OS	Operating System
OSINT	Opensource Intelligence

OWASP	The Open Web Application Security Project
PDF	Portable Document Format
POC	Proof of Concept
QA	Quality Assurance
RSA	Rivest–Shamir–Adleman
SFTP	Secure File Transfer Protocol
SOC	Security Operation Centre
SQL	Standardized Query Language
SSH	Secure Shell
UI	User Interface
URI	Unified Resource Identifier
URL	Unified Resource Locator

1 Introduction

The inspiration for the thesis came from my fascination with Capture the Flag challenges (CTF). A CTF challenge is a puzzle where solving the puzzle leads to a token representing a symbolic flag. Like the sport or military challenge carrying the same name, the holder of the flag has the proof of infiltrating the target. Depending on the execution of the CTF challenge, the flag can be an actual flag, any other item, or a virtual item, such as a line of text. In cyber security challenges, the flag is commonly a line of text like an identifiable code or a funny sentence. The winner is the one who finds the flag first.

The challenges in the game were designed to allow the participants to test a large variety of tools and techniques. The scenario was not designed to be a realistic production environment, but instead a puzzle-like experience with realistic characteristics. Individual challenges and techniques were realistic, but the environments contained obvious hints which would not be found in a real production system. Some puzzles had clearer hints and some puzzles required tools or knowhow to view them. The idea was to test participants' technical competences and introduce them to the technical solutions and tools used in offensive cyber security industry. The tools and techniques are commonly used by malicious actors in actual cyberattacks as well as hobbyists solving CTF challenges. The selected tools were tools that require no special training to use and have decent documentation.

The customer for the study, Gofore is a publicly listed consultancy company based in Tampere, Finland. Gofore's initial business focus was on software development, but has expanded into leadership consulting, quality assurance, change management, hardware development and cyber security. (Gofore Oyj, 2020)

Gofore is characterized as an attractive and responsible employer in the media on different occasions, for example being named Engineering Employer of the Year 2022 (Gofore Oyj, n.d.) and Employer Brand of the year in Finland 2021. (Gofore Oyj, 2021) Employee welfare is an important

part of the reason for the company's success in mentioned accreditations. Information technology (IT) companies in general provide better employee benefits than several other industries. Flexible working hours, choice of tools, lunch- and culture benefits as well as extended healthcare are becoming commonplace in the IT sector. In addition to the benefits, Gofore provides numerous types of community activities and professional development possibilities. Community activities are held both by the employer and employees but provided by the employer.

Gofore holds communality events for employees. The purpose of the communality events is to build a pleasant working community as well as teach or study stuff not necessarily directly related to work but might be fun to learn. Most of the events are organized by the employees and from employee initiative. The events can be competitions, teaching events, sporting events, tasting events or whatever comes to mind within ethical guidelines and reasonable cost. During the Covid-19 pandemic, the communality events have been mostly remote or with a very small number of participants.

A CTF event, by nature, is something that can be arranged remotely for the whole company. The event was held in September and November in 2021. A communality event was a perfect opportunity to gather enough people to participate in the challenge and a questionnaire at the end. Constant learning is necessary in the IT industry and cyber security concepts can be difficult to grasp if one has no experience in the field. To get people to participate in voluntary learning, motivation is needed. Positive competition can improve motivation (Shannon, 2020). Participating in a CTF challenge can be both fun and educational.

2 Theory

2.1 Gamified learning

Hamari (2015) describes the two definitions of gamification based on peer-viewed literature as "*1) The use of game elements in non-game contexts (Deterding et al., 2011).*

2) A process of providing affordances for gameful experiences which support the customers' overall value creation (Huotari & Hamari, 2012 - Study 2)."

A CTF competition does not strictly fit in either of these categories, because the context is intentionally a game while pretending not to be and defining customers in general or their value is difficult in this context. Even though the categorizations do not fit, the described Huotari & Hamari (2012) suggest that "*gamefulness emerges from the psychological consequences which derive from using the gamified system*" (Huotari & Hamari, 2012). From that sentence, it could be derived that the object of gamification is to trigger psychological responses such as emotions to create an impactful learning experience. The idea is to make the exercise more engaging giving the sensations of reward and achievement. Competitiveness can be a driving factor as well, but all gamified learning experiences don't need to be competitions. (Huotari & Hamari, 2012) Gamification introduced into a social group can produce social benefits as well. Koek, Chen and Yu (2022) mention that game content can be a beneficial factor in producing positive attitudes and behaviours. Koek, Chen and Yu discuss the phenomenon purely with Video games and use role playing choices as examples of moral dilemmas. (Koek et al., 2022) The context is not clearly translatable to a gamified learning environment where a clear persona is not assumed similarly to as in role playing games.

Hamari (2015) lists different psychological factors linked to games: Autonomy, flow, suspense, relatedness, immersion, achievement, and playfulness. (Hamari, 2015) A CTF can have these factors included in different forms. Autonomy can be attempted with implementing different solution options and allowing freedom of tools. A flow or concentrated focus helps to achieve better results, as there are several elements to consider at the same time. Implementing puzzles requiring information from different locations or progress steps challenges the concentration of the participant. Solutions where success is not immediately observable can be used to add suspense. Relatedness can be attempted with familiar environments, products, or brands. A sense of achievement is a desired effect from solving the puzzles. Playfulness can be attempted with competition and humoristic elements in the puzzles and story.

2.2 Penetration testing and red teaming

Penetration testing is the act of imitating a hacker for the purpose of compromising a system, software, server, or network. Penetration testing can be targeted towards software or hardware. Most commonly penetration testing is performed on web-applications and carried out via network connection. The penetration test scope can vary vastly between assignments. The number of targets, attack surface on targets, and agreed parameters affect the extent of the penetration test.

A penetration test cannot confirm a system is secure, it can only confirm if it is insecure. Penetration test can be used to provide enough confidence of a system's or product's probable security for a release. Penetration tests should be treated as part of development lifecycle and maintenance. Even without any changes to a system or product, surrounding dependencies can change or new vulnerabilities can be discovered in dependencies or technologies used. (Clark, 2013)

Development processes and penetration testing should ideally be considered as complementing each other. The development team should to some degree be aware of what issues the penetration testers will discover. Experienced penetration testers can still discover vulnerabilities that are not usually considered in development process or are too subtle for developers to consider. (The National Cyber Security Centre, 2022)

Red teaming is an offensive exercise or test method like penetration testing. In cyber security challenges, teams are usually assigned a colour. The colours indicate the role of the team in the exercise. The number of colours varies by need and organizer. The most used teams are red and blue, where red team is attacking, and blue team is defending. Blue team represents IT personnel in a company, administrators, Security Operation Centre (SOC) operators, developers, public relations, and management for example. Red team represent hackers, cyber criminals, burglars, activists, terrorists or whatever the scenario needs. Additional colours can be used and there is no official rulebook for them. In a red team exercise or test, the minimum number of teams is one, the red one. Blue team is usually automatically assumed inside the target even if one is not especially

formed for the occasion. Additional teams are usually used when the occasion is a cyber security exercise or similar. White team is usually the team running the exercise. The exercise might be scripted to have certain events, such as different attacks, data leaks to the press, equipment failures for example. Green team is usually a maintenance team. Their job is to construct and keep the infrastructure for the exercise maintained. (Shamane, 2021)

Red teaming means testing security of a target by imitating an attacker. The purpose is to provide different viewpoints that maintainers might have not considered when designing and implementing security controls and processes. Red teaming can be performed by an internal red team or an external contractor. Using a contractor might provide different results than using an internal red team. Internal red team might be aware of processes and customs in the company, which may distort the approach to the testing. Maintaining an internal red team also requires significant resources which may be too costly for most companies. (Clark, 2013)

Red teaming usually has a broader scope than just testing IT systems or software products of a company. The read team exercise can contain testing physical access to customer premises which may consist of lockpicking, social engineering or exploiting unsupervised parts of security. Companies may confuse red teaming with penetration testing or use them as synonyms. (CSRC Content, n.d.) (The Development, Concepts and Doctrine Centre, n.d.)

2.3 Capture The Flag challenges

CTF challenges are gamified cyber security challenges where the object is to solve one or several problems usually by using cybersecurity skillset. CTFs are a common entry point to the cyber security field. (CTF 101 n.d.)

CTF challenges can be based on red team or incident response skills. Commonly CTF competitions have been more red team or penetration test focused, where participants attack a target and try

to gain access and collect flags before other contestants. There are also CTF competitions and online challenges based on forensics, cryptography, and various other cyber security skills.

CTFs can consist of several different types of challenges consisting of various technologies and techniques. There are technologies that are constantly used in different CTFs, such as steganography, cryptography, hash cracking and exploiting common vulnerabilities. These different techniques and technologies can be combined as larger scenarios or as separate puzzles. (Dubey, 2020)

CTF challenges have been a part of the security community ever since DEF CON -event held in 1996 in Las Vegas. The first CTFs were not as organized as the ones today. Points were given by judges, or the targets were not clearly defined. The number of CTFs has been increasing over the years. DEF CON remains a popular event with a popular CTF which now requires teams to qualify before entering because of the sheer number of potential participants. (DDTek, 2020)

Currently CTFs are no longer only for events and conferences but for broader audience over the internet. There are several platforms on which users can legally hack into different types of computers, solve puzzles and challenges, and compete against each other. These sites have several types of challenges, puzzles, and machines with different difficulties. (*CTF Sites - Biggest Collection Of CTF Sites*, n.d.)

CTF challenges have different categorizations depending on service provider, event, or any other organizer. Gonzalez et al. (2019) divides CTF challenges in to eight different categories in their own university CTF:

- Firmware
- Forensics
- Man In the Middle (MITM)
- Network forensics
- Network programming

- Programming
- Reversing
- Web

These categories have some overlap, such as MITM can belong into nearly every category for its nature of stealing data in transit somewhere between sender and recipient and Network forensics could be included inside Forensics or just Networking combined with Network programming. As comparison, Hack The Box (*How to Play Challenges*, n.d.) prefers more technique focused categorisation:

- Reversing
- Stego
- Crypto
- Web
- Forensics
- Open Source Intelligence (OSINT)
- Pwn
- Mobile
- Misc

Reversing is reverse engineering, which focuses on finding flags for example from directly in the source code in the easier ones and from encrypted memory spaces or similar in the more difficult ones. Stego focuses on Steganography, which is covered in this chapter, and similar data hiding puzzles. Crypto focuses on cryptographic algorithms and other encryptions. Web is a broader category with several network and web-application related technologies involved. Forensics is detective work, finding data in memory dumps, disk dumps or hacked systems. OSINT is about finding data in publicly available sources, such as social media. Pwn, which is a wordplay of the word own, and is used mostly in gaming and other computer related activities (*What Does 'Pwn' Mean?*, n.d.), is quite similar to reverse engineering, except exploit development is a large part. Mobile focuses on mobile systems such as phones. Final category Misc is a category for everything not fitting the above categories.

2.3.1 Capture the Flag service providers

Hack the box has been a popular platform for CTF type challenges in the recent years. Hack the box was founded in 2017, and currently it has over one million users. Hack the box has challenges and cyber security labs for all skill levels from beginner to proficient professionals. It now offers products for universities, companies, and individuals. Hack The Box also has an academy type of learning environment. (*All About Hack The Box*, 2022)

Try Hack Me is also a service provider for CTF challenges online. Try Hack Me was founded in 2018. The environment consists of virtual rooms created by staff and users. The virtual rooms can consist of theoretical portions and practical portions. A room can have a virtual computer attached to it, which can be used as target of the training exercises, such as hacking techniques. (*TryHack-Me | Cyber Security Training*, n.d.)

VulnHub is a website which provides resources and links for learning about cyber security. The main functionality is to provide virtual computer images, which can be deployed on home computers or security labs. VulnHub is a free community-based service, which can affect the quality and security of the provided machine images since they are not reviewed by any staff. (*About ~ VulnHub*, n.d.)

Both Hack The Box and Try Hack Me provide both free and paid content. Try Hack Me reports 80% of its learning contents to be free. All three of the mentioned platforms depend on community provided material but Hack The Box and Try Hack Me have an approval process for submitted material.

2.3.2 Capture the Flag events

CTFs as remain popular as dedicated events as well as part of information security related events and conferences. The DEF CON event, which is considered as the start of computer related CTF

events, is still one of the most popular and prestigious CTFs. (Davis et al., 2014) Disobey, a Nordic cyber security event, holds both an entry CTF and an event CTF. With the entry CTF, participants can buy a distinct and a cheaper badge, which indicated that the participant solved the CTF. (*CTF*, n.d.) Blackhat is the original American event and Blackhat Europe, Blackhat Asia and Blackhat MEA are local events distributed around the world. Blackhat events are famous for their lectures, briefings, and trainings, but often hold a CTF competition too. (*Black Hat*, n.d.) HitCon is a Taiwanese cyber security event, which holds a CTF event usually accessible remotely by practitioners around the world. (*HITCON CTF 2022*, n.d.)

2.4 Infrastructure technologies and techniques

Amazon Web Services (AWS) is an ecosystem for building and hosting various types of web-applications and deploying numerous types of services. Amazon calls AWS "The leading cloud platform". According to the AWS info-page, they have significantly more services and functionalities than any other cloud platform. AWS provides several services that allow hosting of files, services, databases, and servers in addition to numerous other products not listed. AWS has serverless components, databases, virtual servers, and file storages. (*What Is AWS*, n.d.; *Cloud Products*, n.d.)

Serverless architecture refers to an architecture the underlying server architecture is not in the control of the developer of the functionality of the application. A service provider such as AWS provides the platform on which functionality can be implemented. A developer must only be concerned with the functionality of the code. Commonly in serverless architectures components are smaller and instead of a complete application only fulfil a single functionality each. An application consists of several components fulfilling their functionality individually. Such functionalities can be a login, a single calculation, or a retrieval of a database record. This of course is just a single representation of what serverless architecture can mean. (*Serverless Architectures*, n.d.)

Serverless architecture can be implemented in various ways, but the main principle is that the developer or administrator of the client organization does not need to concern with maintaining the

underlying servers or their software. Serverless components can be implemented with containers, virtual computers, or other compartmentalizing techniques. Containers are more light weight than virtual computers because they mostly compartmentalize by using namespaces, whereas virtual computers have a virtualized hardware layer emulating actual hardware components such as the central processing unit or hard drive. The choice of the service provider should not be visible to the customer using serverless components to implement their own services. (Nguyen, 2019)

AWS Elastic Computer Cloud (EC2) is a service where instances of different types of virtual machines can be deployed to in several different methods. EC2 has several operating systems and distributions to choose from. The virtual machines can be deployed from AWS-provided or custom images. Configured EC2 instances can be converted into images for easier replication. Different orchestration tools such as Kubernetes and Terraform can be used to create virtual machines. The EC2 instance is a complete virtual server with an operating system, just the virtualization and everything under that is under AWS control. (*Secure and Resizable Cloud Compute – Amazon EC2 – Amazon Web Services*, n.d.)

AWS Simple Storage Service (S3) is a scalable object storage service in AWS ecosystem, which is mostly used to store and serve files. Webpages which are made of static content are most efficiently distributed as files. Common uses for S3 include storing static webpages for web-services or storing logfiles for archiving. Data can be replicated to several regions around the world to provide resilience and accessibility. The access to the data can be controlled in various ways, including authentication or access control lists. AWS has server service which can be used to analyse the S3 data to gain statistics and insights. (*Cloud Object Storage – Amazon S3 – Amazon Web Services*, n.d.)

Slack Apps are applications that can be added to a Slack workspace, which can be used to automate tasks and run code. It can be used to monitor channels, send automated responses and

messages, or make channel messages interactive using buttons. Bots are one usage of Slack Apps. (Slack, n.d.)

The Slack App can use the Slack event Application Programming Interface (API) to monitor events within the scope it has privileges to. Usual triggers are mentions or direct messages. Mentioning the bot's username, with the @-annotation, is a popular way of triggering the bot to perform something on a channel. Sending direct messages to the bot is a trigger more used with actions and data not meant for a wider audience but for a single user.

More functionality to Slack Apps can be added using the Slack API. Using the API, external resources can be integrated to the bot. External applications need to authenticate to the slack API using credentials created for the application. Identifying which messages, the external applications are referencing to are identified by a unique identifier. Identifiers are also issued to users and channels and other resources referable in the slack API. (Slack, n.d.)

2.5 Challenge solving techniques

All techniques hackers or penetration testers use are not hacking techniques. The same techniques used by several other professions in the IT field as well as non-IT people, are the basis for most vulnerabilities, exploits and defences. Their understanding helps information security professionals in their work. There is no universal categorisation for those techniques which would fit all purposes. For this purpose, the techniques are categorized as *web and communications related* and *local***.**

2.5.1 Web and communications related techniques

Modern websites often consist of **resources** from several different sources. JavaScript libraries, Cascading Stylesheets (CSS) and images are the most common resources included from external sources such as content distribution networks (CDN). CDNs can be provided by different organiza-

tions such as the owner of the resource, a third party CDN provider, or the organization hosting the site. Including resources from remote locations is always a security risk and the source should be always trusted, and the integrity of the file inspected. (*Identifying Resources on the Web - HTTP | MDN*, n.d.; *Subresource Integrity - Web Security | MDN*, n.d.)

Inspecting these sources can provide additional information on adjacent systems and third-party suppliers. Developer tools on browsers allow inspection of these sources. (*Network Monitor — Firefox Source Docs Documentation*, n.d.)

Web resources are often referred to by their **Uniform Resource Locator (URL).** URL has an acronym commonly not opened because of its commonplace use. URL is one implementation of Uniform Resource Identifier (URI). URL contains the information of how and from where a resource can be retrieved. The structure in common website URLs is described in Figure 1. The protocol part contains the information on how to retrieve the resource; in the example, using the encrypted HTTPS-protocol. The domain or authority part contains the authority governing the domain addresses. Global network of Domain Name Servers (DNS) has records of these public authorities and their nameservers. The resource name identifies the resource. The name can be the actual name of the file or some other reference depending on implementation. The URL parameters and values section contain additional variables attached to the URL. These parameters have numerous different uses depending on the server implementation. In the example, *thumbnail=0* could instruct the server to return the full image and not a thumbnail. (*Resources and URIs - HTTP | MDN*, n.d.)

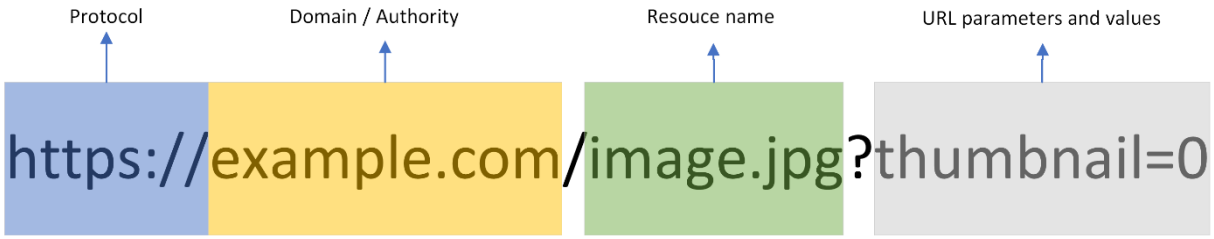


Figure 1. URL structure example

URLs can be fuzzed. **Fuzzing** is a methodology used in several types of inputs, including website URLs sent to a webserver. Fuzzing means trying different inputs and observing the impact. The input can be random characters, words, numbers, binary or many other types, depending on the system and the input interface being tested. Fuzzing can be used to find errors, faults, and misconfigurations. (*Fuzzing | OWASP Foundation*, n.d.)

```
┌──(blaudoom㉿HTBKALI2021)-[~]
└─$ ffuf -u http://10.129.194.186/FUZZ -w /usr/share/wordlists/dirb/big.txt

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v1.1.0
________________________________________________

 :: Method           : GET
 :: URL              : http://10.129.194.186/FUZZ
 :: Wordlist         : FUZZ: /usr/share/wordlists/dirb/big.txt
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200,204,301,302,307,401,403
________________________________________________

.htpasswd               [Status: 403, Size: 279, Words: 20, Lines: 10]
.htaccess               [Status: 403, Size: 279, Words: 20, Lines: 10]
server-status           [Status: 403, Size: 279, Words: 20, Lines: 10]
:: Progress: [20469/20469] :: Job [1/1] :: 852 req/sec :: Duration: [0:00:24] :: Errors: 0 ::

┌──(blaudoom㉿HTBKALI2021)-[~]
└─$ ffuf -u http://10.129.194.186/FUZZ -w /usr/share/wordlists/dirb/big.txt
```

Figure 2. Example usage of fuzzing tool ffuf

When speaking of http-URL-fuzzing, different words are given in the URL, for example *http://example.com/admin* or *http://example.com/password*. There are tools especially designed for this task, *ffuf*, which is shown in Figure 2, and *dirbuster* for example. This sort of fuzzing is commonly included in crawling or spidering within tools commonly used. Usually, terms crawl or spider imply recursive approach where discovered directories are fuzzed separately with same or more targeted wordlists, as well as collecting other information from the sites visited. Fuzzing websites and web-applications for files and directories can be considered part of crawling or spidering but they should not be used as synonyms. (Panta, 2015)

One resource type mentioned earlier in the chapter was JavaScript. JavaScript source files or similar TypeScript source files are common components in websites and web-applications. (Brown,

2016) Included scripts are often minimized or obfuscated when viewed. **Obfuscating** means making code less readable for humans and analysers. It is used in malicious code, but also in normal applications. Minimization could be considered the most basic form of obfuscation. In minimization, variable and function names are shortened and whitespaces are removed when not functional. Minimization is used to decrease the file size of the code. However, since code must be readable to interpreters, interpreters can be used to make the code human readable again. Using multi-layer obfuscation can make this more difficult and special tools or toolchains are needed to make the code readable again. Several browsers have capabilities to display JavaScript code in de-obfuscated form, but multi-layered obfuscation can still make the code hard to read. **De-obfuscating** code depends on most used methods of obfuscating code. (Ndichu et al., 2020)

Online de-obfuscators such as https://deobfuscate.io (*JavaScript Deobfuscator*, n.d.) have algorithms to assess the methods used, but it also provides options to modify the de-obfuscating process to achieve the desired result. These options include but are not limited to common obfuscating methods of packing variables into arrays and using proxy-functions.

Websites have broader functionality than just rendering static HTML-pages. Good coding practices encourage the use of a version control system. **Git** is a version is a decentralized version control system. Version control systems are most commonly used to record changes made to software source code. Other types of information can be version controlled as well. Using a version control system helps developers keep track of other developer changes and helps keep the previous versions of the code accessible. The code is stored in what several version control systems refer to as a repository. Git is a version control system which performs nearly all operations locally and can be used as a local version control. These changes can be pushed to a remote repository to be accessible to other locations or developers. Each up-to-date instance of the repository contains the information of the same changes as other local repositories, including history. (Chacon & Straub, 2014)

GitHub is a commercial service provider of hosted Git-repositories and related products. GitHub is one of the most used Git service providers on the internet. (*GitHub Commands 27.44% Market Share in Software Configuration Management*, n.d.) GitHub offers private and public repositories and is used by various open-source projects. The files and their history are visible to everyone on a public repository. (*GitHub's Products*, n.d.)

Secure coding practices dictate that all secrets such as passwords and keys should be kept encrypted and in a secure location. Source code and version control are not considered secure locations and storing secrets in either location is discouraged. (Turpin, 2010) There are public examples of companies posting Secure Shell (SSH) credentials or keys to public repositories. (Bradbury, 2019)

SSH is commonly known by the acronym, which is the command used to run the client implementation of the protocol on most systems, is a secured network protocol used for remote login and other network services. SSH provides a possibility to authenticate users with public key authentication in addition or instead of password authentication. (Lonvick & Ylonen, 2006)

In **public key authentication**, the public key is published and can be distributed to servers or recipients of files. The private key is kept secret by the owner of the key, in this case the user logging into a server. Data which can be decrypted with the public key of the pair, can only be data which was encrypted with the private key of the pair. When a server receives a message, which can be decrypted with the public key of a user, the message must have been sent by the holder of the private key and this can be used as authentication. (*What Is SSH Public Key Authentication?*, n.d.)

SSH is used by variety of other protocols as well. **File Transfer Protocol (FTP)** is a protocol for transporting files over the internet. By default, FTP is insecure without any transport encryption. (*File Transfer Protocol*, 1985) FTP is still widely used, but using Secure File Transfer Protocol (SFTP)

adds an SSH layer to encrypt the communication. (*SFTP Protocol, Clients, Servers Etc. Page by the Original Author of SFTP.*, n.d.)

FTP-servers can allow anonymous access, which means that no login credentials are required to log in, or the credentials are username Anonymous and password Anonymous. The implementation of anonymous access varies between server applications used. FTP servers allowing anonymous access can have a purpose when sharing publicly accessible files. As with all publicly accessible domains, discretion on what to share is advised.

SSH is widely used to access Linux systems remotely. Remote and local sessions on Linux servers are assigned a handler to handle the session. Usually, this is a binary which is a shell interpreter such as *sh, the bourne shell,* or *bash, the bourn again shell,* which are fully interactive shell interpreters and can run normal commands. In some cases, the user can be assigned a limited functionality session handler such as *rbash*. The limited functionality session handler can be used to enable the user to use the access only for predefined tasks, such as restarting services or editing defined files. There are several ways of assigning a session handler. Most commonly session handler is assigned in the **/etc/passwd** file.

```
gjordis:x:1000:1000:,,,:/home/gjordis:/bin/bash
```

Figure 3. Example line of /etc/passwd file

In Figure 3, an example line from the */etc/passwd* file is represented. Different values are separated by a colon (:). The parts meaningful here are the first one, the username *gjordis*, identifying the user. The third value *1000* is the user id which can also be used to identify the user. The last value */bin/bash* is the application assigned to handle sessions for user *gjordis* when the user logs in. Historically passwords were stored in the file, but they have moved to /etc/*shadow*. (Gite, 2006)

```
pollinate:*:18677:0:99999:7:::
gjordis:$6$Ob02I4lMsZr.KOck$k.Scyoc●●●                                    ●●● 55frZ36kZB0ORO/:18750:0:99999:7:::
postgres:*:18772:0:99999:7:::
```

Figure 4. Example of /etc/shadow

Similar file is the */etc/shadow* shown in Figure 4*,* which is the successor of */etc/passwd* in holding passwords. In the line for user *gjordis* the string starting with "6" is the hash of the user's password. The "6" indicated the type of hash algorithm used. In this case it is SHA512 Crypt. (*Example_hashes [Hashcat Wiki]*, n.d.)

2.5.2 Local techniques

Steganography means hiding data in another data. The word steganography is derived from Greek words steganos meaning covered and the graphei meaning writing. (*Definition of STEGANOGRAPHY*, n.d.) Usually in CTF challenges this is done by inserting non-visible data inside images, which will not be rendered when the image is viewed. There are several methods of steganography. To view the data, the correct method is needed to extract the data. (Khamis, 2021) Steganography has been seen used as a communication tool between infected computers and attackers command and control server. (Manky, 2019)

- Least Significant Bit (LSB) steganography hides the data in bits of bytes which affect the value the least, resulting in least distortion in the image. (Panwar et al., 2020)
- Masking as a method is used mostly on grayscale images. The data is hidden in the visual area, but in a way that is unnoticeable to humans. Masking is more robust than LSB steganography because it is not affected by compression in similar fashion. (Nosrati et al., 2011)
- Transformations are a sophisticated and complex way of hiding data. Transformations modify the jpg discrete cosine functions to hide the data. (Nosrati et al., 2011)

Hiding data is not encrypting it. Hidden data can be encrypted with different cryptographic functions. Encrypted keys and passwords are other common uses of cryptographic functions in compu-

ting. Different cryptographic functions can be used to calculate a hash or a concentrate of data. In asymmetric functions, this action cannot be reversed with calculations. One use of asymmetric cryptographic function generated hashes is comparing secret data, such as passwords. When a user saves a password into a system, the system calculates the hash with given conditions, algorithm and iterations for example and saves that hash to the database instead of the clear-text password. This way, if the database is stolen, the passwords are not clearly readable. (Halunen, 2012)

Even if password hashes cannot be reversed, they can be **cracked**. Cracking means trying to find what string the hash corresponds to. More sophisticated algorithms and number of iterations and using a salt when generating the password hash make the progress more difficult. The common ways of cracking hashes are: Bruteforcing, which means trying symbols to calculate hashes and compare them with the original; Dictionary attack, where hashes are calculated from dictionary words or leaked lists of known or commonly used passwords; Hybrid method, which is a combination of both. Using known words with added numbers or symbols such as a year are used to calculate the hash to compare; Rainbow tables, which are pre-calculated hashes. (Baloch, 2015)

Tools such as John the Ripper and hashcat can be used for cracking. They have automation for calculating the hash and comparing it to the original. John the Ripper can try to guess the target hash by its attributes such as length.

Archives are a common target of encryption. Since archives are meant for storing or transporting data, encryption is advised. Zip is a format of compressed archive. While compressing data into a Zip-archive, it can be encrypted. Different encryption algorithms can be used. (*How to Password Protect a Zip Folder*, n.d.) Because the archive can be extracted with the correct password, the password or its hash must be included in the archive. The password hash can be extracted from the archive with tools such as zip2john and cracked with the tools mentioned in this chapter.

Bruteforcing is a very similar technique to fuzzing, where different inputs are tried, and results are observed. In bruteforcing, trying different inputs is focused on trying different username and password combinations until a valid combination is found. Bruteforcing can be executed character by character, known as a simple brute force attack, or with a wordlist which tries only words found on the list, a dictionary attack. Dictionary attacks are not efficient in finding random alphanumeric passwords unless they are commonly used such as “password123”. These commonly used passwords can be found in wordlists distributed by information security professionals, companies, and hackers. If the password is a simple dictionary word, the probability of discovering it in reasonable time is significantly higher than discovering a random alphanumeric password, logical alphanumerical passwords, such as “abc123” are discovered in shorter times. Hybrid attacks combine words and random characters or for example incrementing numbers. (*What Is a Brute Force Attack?*, n.d.)

Not all data is purposefully hidden or encrypted. Data included in compiled applications can be valuable or sensitive or both. Compiled applications usually have some clear text portions readable directly from the binary, but for most data and source code, the application will need to be reverse engineered. **Reverse engineering** is a wide term usually referring to decompiling a binary back to its source code. Unlike decompiling interpreter languages, binary applications require additional information to be decompiled. Some applications are designed to be difficult to decompile and can have obfuscation or encryption embedded. (Breuer et al., 2017)

Decompiling means reverting a compiled application or other file back into its source code form. Because compilers can strip away non-functional parts of the source code when compiling, reverting to the identical source code is usually impossible. Instead, the decompiler can try to revert the compiled execution level commands back to the way they are represented in the source code language. (Eilam, 2005)

Decompiling can be used to analyse applications such as malware to gain insights on their functionality, to recover lost source code to some extent, and for hiding flags in CTF challenges. Depending on the language, decompiling can require additional information, such as the target architecture the application was compiled to, or the compiler used to for compilation. Some applications store this information in plain text inside the compiled file in headers. For example, Executable and Linkable format (ELF) files, known as ELF-binaries contain ELF-headers containing additional information on the application.

Interpreted languages are commonly easier to decompile, since their compiled forms must be interpretable by the interpreter. Java can be considered both compiled and interpreted. Java is commonly compiled from java-files to class-files which are packed inside a zip-archive, which can have filename extensions such as jar meaning java archive or war meaning web application archive. Java can be decompiled with most java compilers and runtime executables. Several Integrated Development Environments (IDE) for java development have built-in decompilers which can be used to revert java archives and libraries into readable source code. There are several online Java decompilers which can be used to view the source code as well. (*8 Best Java Decompilers in 2022 | Java Hungry*, n.d.)

There are several tools for decompiling binaries, one popular tool is Ghidra, which generates a readable C source code of the application. (Eilam, 2005)

3 Methodology

The object was to construct a controlled environment platform where participants could solve puzzles at their own rate and report their progress via automated process. Background information, results of which puzzles they solved and subjective feedback questionnaire would be used for comparable information and supporting feedback for possible similar future events.

3.1 Initial hypothesis and research questions

The initial hypothesis was that employees with either a technical position or a higher educational level would choose technical solutions to the puzzles and advance further in the challenge. Non-technical or less educated employees were expected to choose more of the non-technical solutions to advance in the challenge when both technical and non-technical techniques were provided in the training session held at the start of the challenge. The intended subject of the research was to compare the technical-orientation of participants education, but no comparable framework was discovered to categorize degrees or education properly by its technical-orientation. Instead, the International standard classification of education (ISCED) standard was used for evaluating the educational level of the participants and the level was taken as the other comparable attribute in addition to the occupational role.

Some of the puzzles for the challenge were non-technical and did not require technical expertise. These puzzles could be solved by investigating given information and clues. Employees with a non-technical role were expected to favour these puzzles to advance in the challenge.

Companies keep educating their employees on cyber threats and proper operating procedures. Different learning methods work for different types of people. The purpose of the research was to observe the types of solutions people with different backgrounds and education choose. This data could be used to identify approachable topics within different groups. With the different topics for different groups, different teaching material could be provided. For groups that fare well in the challenge and show interest in the topic, similar challenging teaching events or platform could be provided. For groups that choose solutions that are deductible and do not require technical expertise, a more top-level teaching approach could be provided. The training session contained nearly all the answers for the challenge, they just needed to be applied correctly. If non-technical participants could use the techniques provided in the training session, similar approach of gamifying the teaching could be used for other training as well.

Red-team techniques were selected for the exercise because CTF challenges are very popular and more attractive to the larger audience than blue-team techniques. Being a hacker is more romanticized by pop-culture than being a system administrator.

The research questions are:

Can a preference of non-technical solutions to puzzles be observed in a multi-choice CTF with employees having lower educational level or non-technical occupational role?

How do employees with lower educational level or non-technical occupational role progress compared to employees with higher education or more technical role in a gamified learning environment?

Do participants with little to no prior experience in CTFs prefer recently taught techniques more than participants with experience?

3.2 Research methods

The research was conducted by providing the participants with a cyber security CTF challenge, with a teaching session before the challenge and a questionnaire form after the challenge. The progress of the participants could also have been monitored from a database, but this tool was not used. The questionnaire was presented to all participants, regardless of if they completed the whole challenge.

The research methods are mixed. Where the questionnaire participants were asked to answer after the challenge, is quantitative except for the feedback at the end of the questionnaire, which was excluded from the analysis, interpreting the results is mostly qualitative.

Of the 800 employees in the company at the time of the event, 130 participated in the training session, 60 participated in the challenge, 11 completed the challenge, 14 answered the question-

naire. The questionnaire form including answers and the summary of the answers is included as appendix 1 and the summary of the answers as appendix 2.

The questions in the questionnaire can be categorized in two categories:

- Exercise participation related information (Questions 1,2,5 & 6)
- Background information (Questions 3,4 & 7)

Questions 1 and 5 were designed to collect the statistics on how many respondents finished the challenge and individual puzzles. Question 1 was “Did you finish the hunt” which meant if the respondent caught the last flag. Question 5 mapped which puzzles each respondent solved. Using this data, the preference towards technical or non-technical as well as recently taught or not taught solutions requiring puzzles. The less- or non-technical solutions were guessing the password, which was simple combination of using both admin as username and password, following URL from an image, logging into an anonymous FTP-server, using steghide to find the ssh-key password in an image, using an online Java-decompiler to view the Java code. All the simple tools, techniques and services for these solutions were explained in the training session and required little to no technical expertise other than using computer applications with instructions. The instructions were available to everyone throughout the whole challenge. The final challenges required some technical skills and were intentionally harder.

Questions 2 and 6 were to offer insights if the challenge was solved by too many or too few participants. If a considerable portion of respondents had answered the challenge being too hard and only a few respondents had answered solving the challenge, the difficulty of the challenge would have been too hard for the participants. As well as if the answers would suggest that the challenge was too easy, and there would be a significant portion of respondents answering having solved the whole challenge, the difficulty would be too easy. If most of the solved puzzles would be with the most obvious techniques which were taught at the teaching session and most respondents an-

swered having attended, the techniques explained in the teaching session were successfully taught to some degree.

The background questions provide the information of participants education, occupational role, and previous experience in CTFs. Exercise questions are used to map the progress and solutions chosen by participants. Questions 3 and 4 map the participants education and occupation where question 7 maps the amount of experience in CTF challenges. These Background questions are used as to categorise the participants when comparing their answers for exercise related questions.

In question 3 the answer options are based on the education levels by Eurydice National Education Systems database. (*The Structure of the European Education Systems 2018 / 19 : Schematic Diagrams*, n.d.). The question is used to measure the respondent's educational level. In question 4 "Is your position .. (Pick closest one that applies) or use other" the roles had a categorisation to technical and non-technical. Of these roles, "Managerial / Accounting / Sales or similar non-technical", "UI Designer, Graphical Designer or similar", "Service Architect", "Change leader / Consultant" were categorized as non-technical. This categorisation is based on common project role responsibilities in Gofore projects and cannot be considered a scientific classification but dividing roles in a tech-company into technical and non-technical is ambiguous. Question 7 was used for the possibility that participants might performed especially well inside their own educational or occupational group if they had notable previous CTF experience.

4 Building the challenge

Building a CTF challenge is a process with several steps. It is easy for anyone with knowledge in server maintenance to boot up a server with a vulnerable service running, but it is more difficult to create entertainment value in the CTF. Some CTFs are by purpose resembling real life scenarios and do not contain planted hints or story, but most do. The difficulty can be increased with the subtlety of the hints and tools needed to discover them. When a CTF is targeted to a broader au-

dience with varying skill level, clear hints and usually a comical story is constructed to guide the process.

4.1 CTF challenge design and build process

Different CTF makers have different methodologies for creating challenges. The process I used in building this challenge can be broken into a few clear steps described in Figure 5.

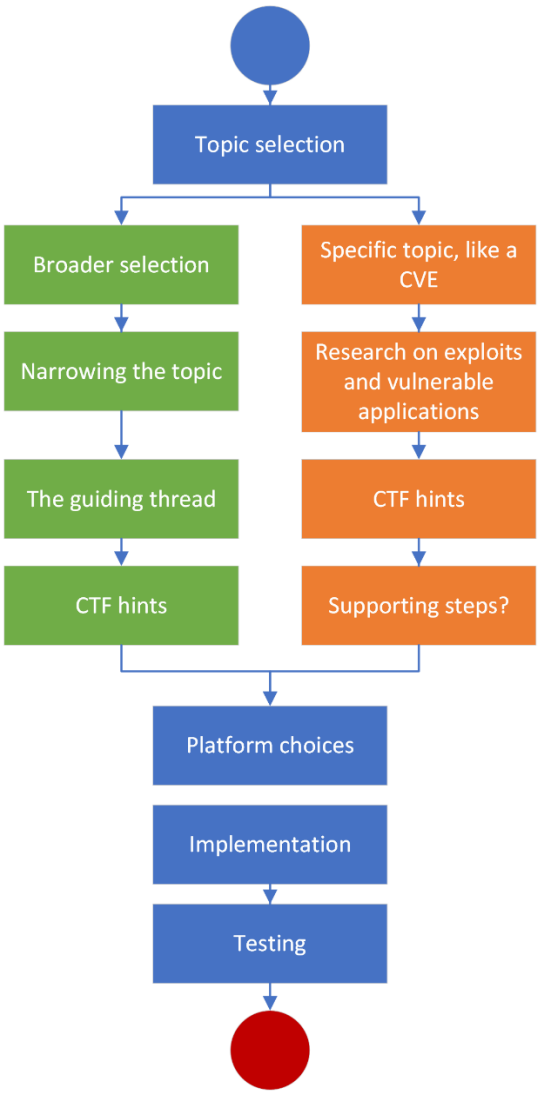


Figure 5. CTF design and build process flow

The selection of topic should be the start of a CTF design. The topic defines the whole challenge. Splitting the topic into two categories, a broad selection and a specific selection can help.

If the purpose of the CTF is to showcase a new vulnerability published in the Common Vulnerabilities and Exposures (CVE) database, common misconfiguration or a trending technology, the focus of building the challenge should be on how to portray the desired aspects of that topic in chosen fashion, either lifelike or game like or something else. These types of challenges are popular in Hack The Box and Try Hack Me, where challenges about new vulnerabilities are released regularly.

When the CTF is a larger event or challenge, several topics and technologies are more likely to be used to keep the challenge from becoming repetitive. These types of challenges are popular in hacker events as competitions and for getting special entry tickets highlighting that you solved the puzzle. Finnish hacker event Disobey arranges a CTF challenge where limited number of hackers can purchase a special entry badge if they solve the puzzle. (Disobey, 2020)

For a specific topic research is needed. If the topic is a vulnerability, public exploits, and proofs of concept (POC) offer a good starting point. POCs are often crude implementations to present the exploitation of a vulnerability in a specific way. POCs can target a specific version or a configuration on a service or application and modifying the target slightly can be used to require small changes to released POCs to solve the challenge. Requiring participants to modify the source code in the POC forces the participants to understand the actions the POC performs, resulting in a better understanding of the actual vulnerability.

Similar approach is to use a research paper or an existing CTF from an event as a basis for the challenge. As participants search for solutions, finding the paper or presentation on the same subject is likely. These sort of papers and presentations often have detailed descriptions on how to exploit a vulnerability or an application. Modifying the application slightly requires the participants to understand the exploitation process instead of just following the instructions given.

If the challenge is broader, choices for technologies need to be narrowed. The criteria for the topics can be chosen case-by-case. Some of the criteria I have used previously have been different parts of a software stack, products by same company, products using same communication protocol and software used by financial companies.

Popular way of choosing topics is to follow a storyline or a guiding thread. The story or thread can be based on an attack chain or following stolen data or employee traffic or conversations. Making the steps in a challenge follow a storyline can add entertainment value to the challenge. An example of a storyline implementing several types of puzzles is:

1. A tweet on twitter or a post on linkedIn on an account of an *SoftwareCompany Inc* gives a press release on a big software delivery to a company named *RandomCorp inc*.
2. Searching the SoftwareCompany website reveals that all their products are hosted on individual subdomains on example.com
3. Enumerating subdomains on example.com with variations of RandomCorp Inc reveal a subdomain randomcorp.example.com
4. Testing the application reveals that the login is vulnerable to SQL-injection which allows login as admin
5. The application is a service portal with uploaded memory dumps of employee computers
6. Analyzing the memory dump provides an IP address for a domain controller
7. This leads to attacking the domain controller and followed by several other steps.

Weather using broader or specified topic hints can be used. The type and number of hints can define or be defined by if the challenge is trying to imitate a real-life scenario or system, or if the challenge is clearly a CTF game. Not all CTFs use hints at all. If a vulnerability is public and especially if it is well known, participants can be expected to identify it without hints. The number and type of hints can also affect the difficulty of the challenge. Providing clear hints on blog posts or chat containing messages such as “Did you remember to update the log4j2?” suggests that the next target should be exploited with log4shell exploit (*Log4Shell*, 2021). Hints can also be subtle, such as identifying a header response.

Hints can also be purposefully misleading often called as “rabbit holes”. (*Definition of RABBIT HOLE*, n.d.) Rabbit holes are used to steer the participants into a wrong direction or make them spend time on a dead-end.

If the challenge was constructed to educate on a specific topic, but the topic is either simple, or exploitable with a single tool, or a platform where the challenge will be released requires more content, supporting steps and features can be added to the challenge. If the specific topic requires supporting steps, following a context or a story can provide structure where additional puzzles can be implemented. If the targeted vulnerability is located in a PDF rendering library and inserting malicious code into a document rendered into a PDF provides code execution capabilities on the server, providing access, a following step could be related to a clean-up script removing old pdf-files with root privileges.

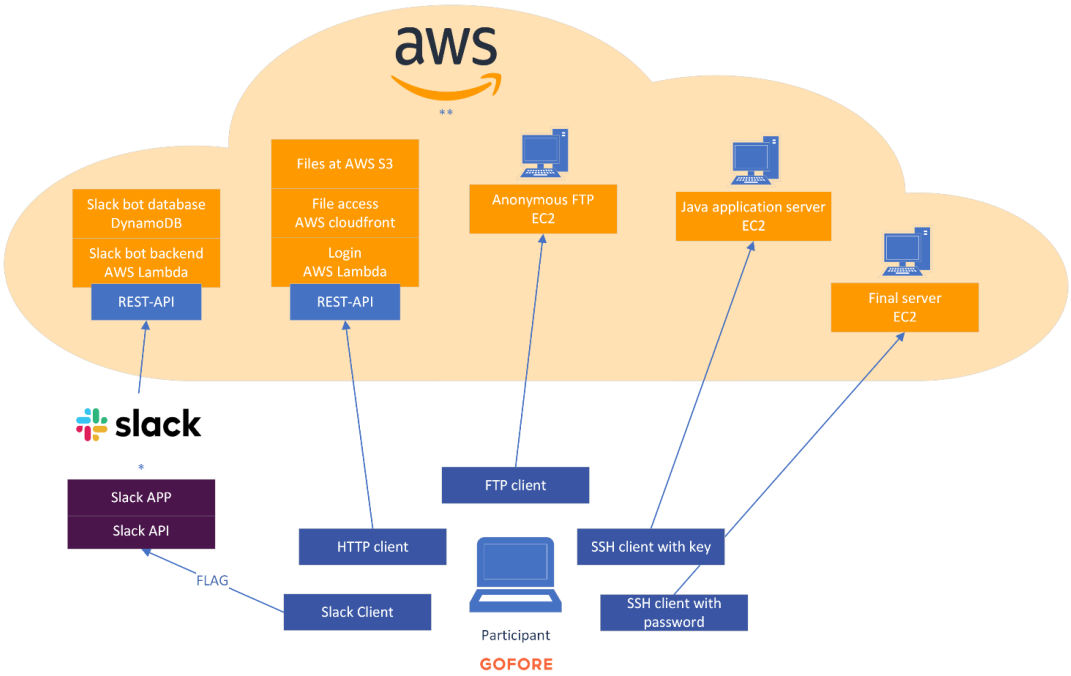
Designing the challenge is an iterative process, where steps need to be taken back in some cases. Figure 5 describes the process as a one-way waterfall but defining parts of the challenge can cause refactoring in previously designed steps. Platform technology choices, which is illustrated at near the end of the flow chart, should be a parallel process to the whole design process. The viability of the implementation of each puzzle and service should be considered when designing them. POCs or vulnerabilities and distributions of software can affect the platform choices. Some software is delivered in containers and containers are often used in presentations and POC explanations of vulnerabilities.

The intended and unintended security flaws should be considered when selecting a platform. Creating a vulnerable installation of a service while limiting the impact of exploitation should be a top priority. Limiting the privileges of an application mean to be exploited is commonly achieved with containers.

After implementing all the steps of the challenge, the challenge should be tested. Ideally, the testing should be conducted by someone unfamiliar with the design of the challenge, but skilled with the topics in question. While testing the challenge, prior knowledge of the implementation should be kept to a minimum. Important aspects to test would be if the challenge can be completed with the information given or if too much information is given, and if the implementation is secure and only the intended paths are exploitable.

4.2 Infrastructure for the challenge

The difficulty in constructing a cyber security hacking challenge is to make a system which is secure in every other aspect than the exploitable route created for the puzzle. The point of the exercise or the research was not to maintain or configure servers or other infrastructure. Infrastructure work was kept at a minimum so the work effort could be directed at the puzzles.



* Slack. (n.d.). *Media Kit*. Slack. Retrieved 11 November 2022, from https://slack.com/media-kit
** *Logos*. (n.d.). Press Center. Retrieved 11 November 2022, from https://press.aboutamazon.com/logos

Figure 6. CTF infrastucture topology diagram

Amazon AWS was chosen to host the challenge puzzles, to keep the overhead of maintaining an underlying system at a minimum. AWS has serverless components, databases, virtual servers, and file storages, which were needed for the exercise. AWS also allows penetration testing tools, excluding denial of service, to be used against services you are hosting on the platform (*Penetration Testing, Test the AWS Environment against Defined Security Standards*, 2022). To limit the exposure of underlying systems and discouraging participants in trying to exploit them, serverless architecture, mainly AWS Lambda and DynamoDB, was used when possible. Where serverless solutions were not viable, different options were used. For systems where shell access was required, AWS EC2 instances, virtual machines, were used. Static files were served from AWS S3 buckets, which are object storages commonly used to serve files. The topology is illustrated in Figure 6, which can be divided into three segments: AWS, Slack, and participant's computer.

AWS Lambdas were used to build simple components fast for login and for Slack APP bot for flag collecting, serverless solutions require minimal configuration. NodeJs implementation of both login and Slack App bot backend functionality were created with Lambdas. The Lambda-functions communicated with an AWS DynamoDB**,** which is an noSQL database service.

EC2 Servers have some advantages over serverless solutions. As some we wanted some puzzles to have server properties, as different open ports, shell, and a filesystem, not all components could be implemented as serverless solutions. For full kernel and filesystem virtual computers, AWS EC2 product was selected. In the puzzles where the solution was to exploit a service, the services were installed on an EC2-instance. The EC2 instance is a complete virtual server with an operating system, just the virtualization and everything under that is under AWS control. Even tough AWS offers several orchestration tools for EC2 instances, in this case, the virtual machines were deployed manually. The operating system on the virtual machines was the Linux based Amazon AMI 2. In the EC2, instances services such as a http and ftp -servers were installed.

AWS S3 was used or service static files like HTML-pages, images, and stylesheets. In the S3 configuration, files can be set as public or private. Both public and private files can also be served via AWS **Cloudfront**, which is a content distribution system, but files marked as private can only be served via Cloudfront. This is a simplification of this case. Private files in S3 can be accessed via other AWS resources as well if permissions are configured properly.

In this case, public files and private files were served via two different Cloudfront distributions, one which requires authentication to access the private files and one that is open.

A return interface for the flags in the challenge was needed. A Slackbot was constructed to receive the flags from the participants. The bot would verify the flag and invite the participant to a new slack channel if the sent flag was correct. The slack channels could be used for peer support in a way that the previous steps would not be spoiled for the participants who have not reached the same progress.

The Slack App bot was constructed in the AWS lambda environment, which is part of the AWS serverless architecture possibilities. Combining AWS lambda from serverless components and Slacks own Slack App functionality, an efficient flag delivery system was constructed with minimal overhead. Slack allows external applications to communicate with a slack server via an API, when authorized by the company's administrator. The application uses different endpoints to send messages to the Slack API. The communications are authenticated with an API-key.

The AWS lambda receives a http-request via the Slack App and Slack event API, when a message is sent to the Slackbot via Slack. When the flag is verified and saved to the database, the bot sends two messages back to the Slack API, one to respond to the sender with a funny phrase to indicate that the flag was correct, and the other to invite the user to the new Slack channel corresponding with the flag.

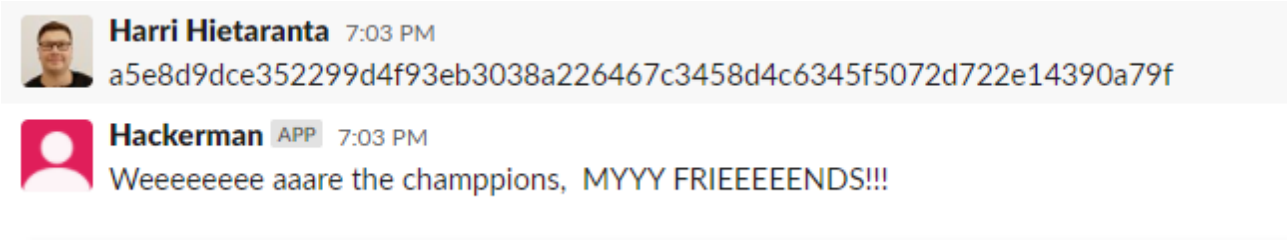


Figure 7. Example of a message sent to Slackbot via Slack

The code of the bot is simple in logic. It receives a http-request, parses the required data from the message, verifies the flag and if the flag is correct, sends a message back, invites the user to a new channel and saves the event to a database. The two-message conversation is illustrated in Figure 7. The whole code for the lambda implementation is readable in appendix 4.

4.3 The challenge puzzles

For a longer challenge such as this, we wanted to add various categories of puzzles to keep the challenge interesting. Mostly working on the web restricts the use of some categories such as hardware hacking or Mobile. In addition, creating decent network related puzzles can be challenging. Web puzzles were an obvious choice based on the environment the challenge was held at. In addition, local file-based puzzles were easy to arrange by distributing the files. OSINT elements were planned but were left out after an employee noticed a twitter account immediately after it was created, mostly likely with keyword based alert or similar.

The puzzles follow a storyline created for the challenge. A message tells the participants, that data has been stolen from our server. The message implies that there is something wrong with our web-application and an URL is given to a website illustrated on top left corner of Figure 8. The par-

ticipants are to investigate and replicate how the attacker compromised the system. Hints are given on the way. The hints start off as very clear and as the challenge progresses, become more hidden and imitating real life data.

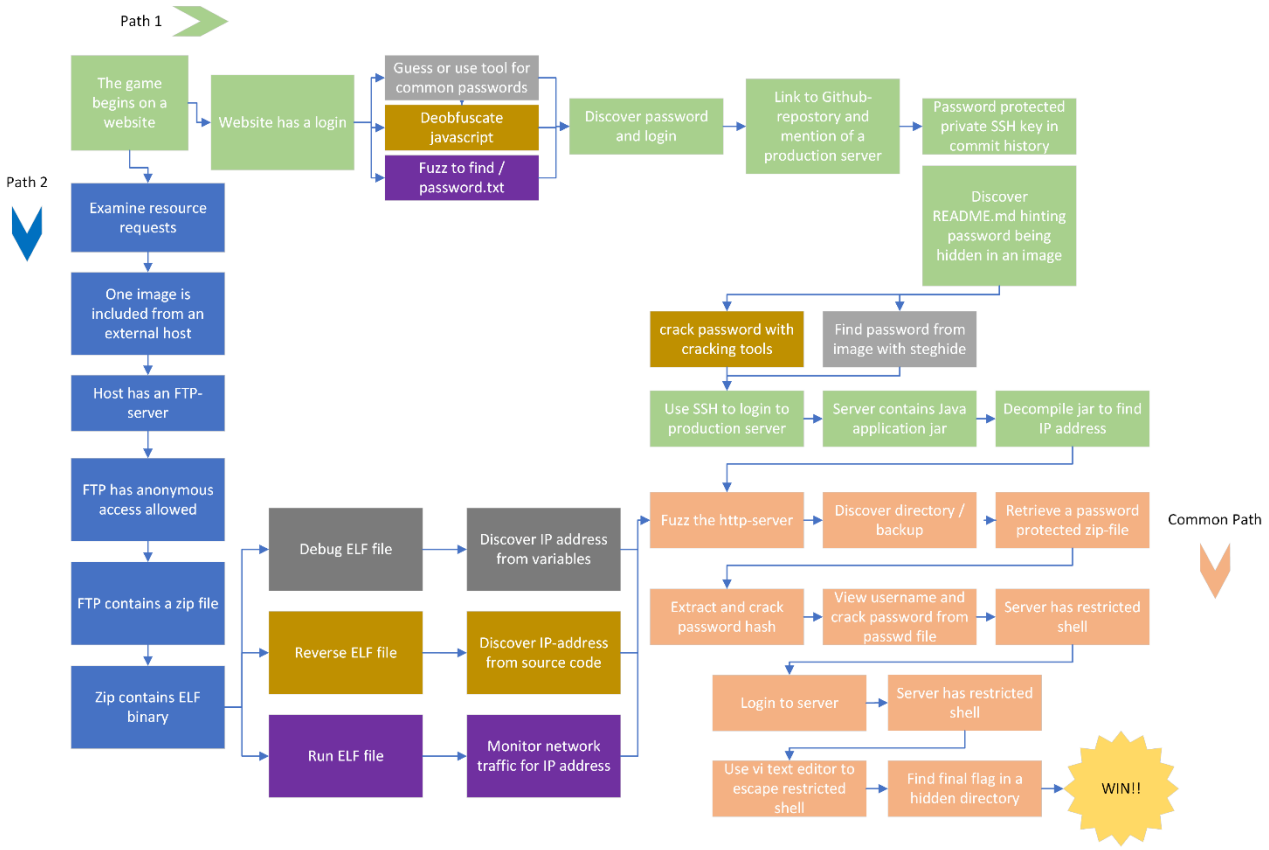


Figure 8. Possible solution paths of the challenge

The first step is to either log in to the service on the given URL or examine the network requests of web resources on the first page. This first choice divides the progress into two separate paths, which later merge when progressing far enough. Figure 8 illustrates both paths, where the path with first choice being the website login follows the flow right and then down on the progress diagram and path 2 where the first choice was to examine the network traffic of the website follows the flow first down and then right, merging with path 1 on login of the final server. Path 1 is col-

oured with light green, path 2 with blue and the common path with light orange. The alternative puzzles which lead to the same result, are marked with grey, brown and purple.

4.3.1 Path 1

Path 1 has three intended possible solutions for accessing the page behind the login. There is only one username and password combination, and each solution provides reveals it. The username and password combination is very simple, **Admin** as both username and password. This combination is something anyone can guess. Most web vulnerability scanners such as Nikto and Burp Suite Pro will also try these default password combinations when testing pages with login forms. These sort of username and password combinations which are considered *weak,* can also be discovered quickly with a brute force attack. *Guessing* or getting the answer from an automated tool, or bruteforcing the login are considered the same solution for the login part.

Another approach in testing websites is fuzzing or dirbusting them, which is discovery of files and directories by name. This process is described in chapter 2.5 Challenge solving techniques. In the challenge, a file called password was served from the S3 bucket and it was easily discoverable with either guessing the filename or using a fuzzer to discover the file. The file was located at the root of the base URL in the following fashion: *https://example.com/**password***. The file contained a password needed to login to the service. Finding the file containing the username and password combination is one of the three solutions.

The site also included a JavaScript file, which can be seen in Figure 17 with the name **script.min.js.** This naming convention suggests that the JavaScript file is obfuscated and consists of arrays with hex-strings as illustrated in Figure 9.

```
var _0x2d35=["\x76\x61\x6C\x75\x65","\x75\x73\x65\x72\x6E\x61\x6D\x65",…
```

Figure 9. Obfuscated JavaScript

The code can be converted back to readable form with several online tools such as https://deobfuscate.io/.

```
function login() {
    var zanea = document.getElementById("username").value;
    var jakeria = document.getElementById("password").value;
    if (zanea !== "admin" || jakeria !== "admin") {
        alert("Wrong username and/or password");
        return;
    }
    ;
    window.location.replace("http://" + zanea + ":" + jakeria +
"@d13yk48c09yexb.cloudfront.net/index.html");
}
```

Figure 10. De-obfuscated JavaScript

When de-obfuscated to readable format as shown in Figure 10, it can be read from the code, that two variables are compared to the string '*admin*' and those strings are retrieved from http elements with ids ***username*** and ***password***. Discovering the username and password combination from the obfuscated JavaScript file is considered the third and the most challenging solution to the login part.

This is just a static test for the Teppo Sorsa service!
This site will be operational as soon as we get it that way. Lets not give out users to anyone else yet. Lets just use the admin/admin account.

Source code at: https://github.com/harrihietaranta/sorsaService

The live service will be installed on : ec2-3-250-98-161.eu-west-1.compute.amazonaws.com

Figure 11. Screen caption from inside the exploitable service

When successfully signed to the exploitable website, a link to a public GitHub repository is found on the page. There is also a mention of an address where the service will be installed. In Figure 11, a URL to the public GitHub repository is shown. There is also a hint for future exploits, the ad-

dress for another server. Recording all addresses is a good practice in both CTF and penetration testing.

main | 1 branch | 0 tags | Go to file | Add file | Code

harrihietaranta Update README.md c4ef1ee on 3 Sep 2021 10 commits

app	initial commit	13 months ago
README.md	Update README.md	6 months ago
flag.txt	IMPORTANT! info on README	13 months ago
index.html	initial commit	13 months ago
index.png	initial commit	13 months ago
password	initial commit	13 months ago
scripts.js	initial commit	13 months ago
scripts.min.js	initial commit	13 months ago
styles.css	initial commit	13 months ago
teppo.jpg	also jpg of teppo	13 months ago
teppo.png	tihihihi	13 months ago

README.md

sorsaService

The Seppo Sorsa service

This service is going to be AWESOME! The BEST service ever! No other service is going to be like this!

- The password is in an IMAGInary location. Now that the keys are here not anymore, its safe to add

! The server is running as user sepposorsa now.

Figure 12. The Github repository

The GitHub repository contains the source files for the service which was penetrated in the first step of the puzzle. The files and the repository contain hints and next steps. The README.md, which is written in markdown is displayed in parsed form in the repository's front page. The file contains a clear hint that the password might be in an image. Steganography was one of the topics of the teaching session at the start of the challenge. Another hint or clear information in the file is the username for the server. This is a CTF type of a hint, and this information would not be written in a readme-file in regular software development project, unless security is neglected.

```
┌──(blaudoom㉿HTBKALI2021)-[~]
└─$ mkdir ctftest

┌──(blaudoom㉿HTBKALI2021)-[~]
└─$ cd ctftest

┌──(blaudoom㉿HTBKALI2021)-[~/ctftest]
└─$ echo "hello" > file.txt

┌──(blaudoom㉿HTBKALI2021)-[~/ctftest]
└─$ cat file.txt
hello

┌──(blaudoom㉿HTBKALI2021)-[~/ctftest]
└─$ steghide embed -ef file.txt -cf stego.jpeg
Enter passphrase:
Re-Enter passphrase:
embedding "file.txt" in "stego.jpeg"... done

┌──(blaudoom㉿HTBKALI2021)-[~/ctftest]
└─$ rm file.txt

┌──(blaudoom㉿HTBKALI2021)-[~/ctftest]
└─$ steghide extract -sf stego.jpeg
Enter passphrase:
wrote extracted data to "file.txt".

┌──(blaudoom㉿HTBKALI2021)-[~/ctftest]
└─$ cat file.txt
hello

┌──(blaudoom㉿HTBKALI2021)-[~/ctftest]
└─$
```

Figure 13. One example of using steganography to hide data

From several methods of hiding data, one was chosen to be used in the challenge. Using Steghide was chosen, because the tool is found on the Kali Linux and Parrot OS security testing Linux distributions. Figure 13 displays the technique of hiding plain text inside a jpeg-image. This example

was used in the training session. In the puzzle, a password for an SSH-key was embedded inside an image with the same tool.

After discovering a password, something to use the password on is required. In CTFs parts needed to solve a challenge can usually be discovered in different puzzles or different locations. It is good practise to write down all passwords, names, usernames, email addresses and URLs a player comes across.

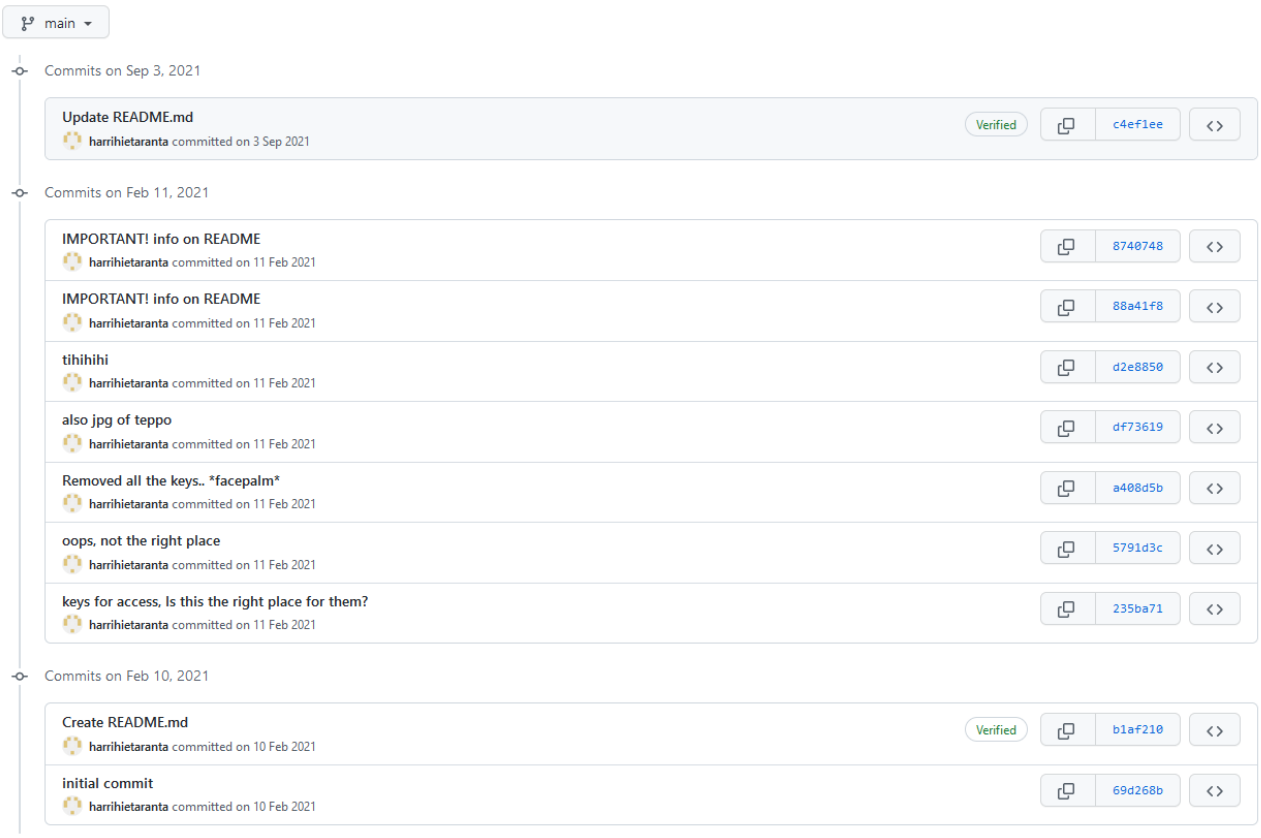


Figure 14. Github repostitory commit history view

In code-auditing and white box penetration testing as well as in CTFs, inspecting the version control history is common practice. In Figure 14, a commit message of “removed all the keys” is a hint to inspect the commit. Removing files from a Git repository does not remove them from previous

commits. Git retains all the changes made to the repository because of its structure of saving references to the previous version. Version control tools using git can be used to view the history, but most Git service providers offer a history view as well.

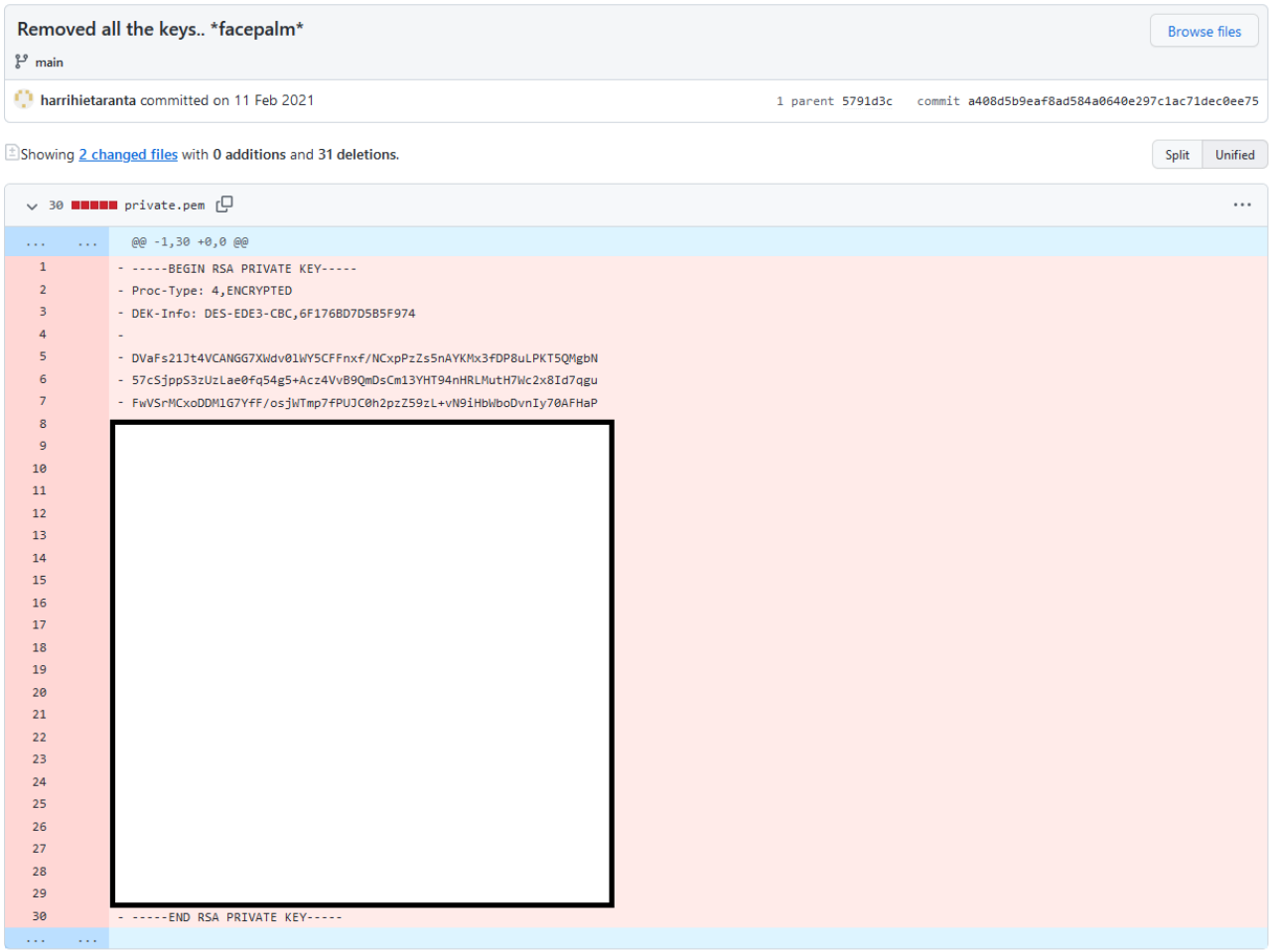


Figure 15. SSH RSA-key in commit history

If secret data is committed to the version control, there are different ways of viewing the data. One method is to view the commit where the data was either added or removed. The whole data is either added or removed lines. In Figure 15, the RSA private SSH-key is shown as removed lines

in the commit which removed the key from the repository. The key is ASCII-text and can be copied directly from the browser and pasted into a text-file on the filesystem.

As visible in the RSA header in Figure 15, the key is encrypted. The hash of the password can be extracted with tools such as John the Ripper. John the Ripper has an auxiliary script ssh2john, which extracts the password hash from the RSA key file. This tool extracts the hash in a format that John the Ripper can understand it and crack it with appropriate wordlist. Cracking passwords is explained in chapter 2.5 “Challenge solving techniques”.

Collected information from previous steps now include: a server address from the webservice which login was hacked, username from the README.md file on GitHub repository, RSA key from the commit history of said repository and the password for the RSA key from cracking or bruteforcing it. SSH public key authentication requires that the private key is only readable by the user using it and the directory containing the key should not be writable by others. After changing the file privileges to correct values, with this information, a successful login to the server should be

From the server to which the SSH-key grants access to, a compiled Java-application can be discovered. The application contains functions for encrypting and decrypting data and creating users. These functions are designed to misdirect the participants.

```
public class ServerConnector {
   static boolean sendNewUserToServer(String user, String password) throws IOException {
      URL url = new URL("http://13.53.141.112");
      URLConnection con = url.openConnection();
      HttpURLConnection http = (HttpURLConnection)con;
      http.setRequestMethod("POST");
      http.setDoOutput(true);
      Map<String, String> arguments = new HashMap();
```

Figure 16. Decompiled Java sourcecode

Address for the next server to target is included in the source code which is visible after decompiling. As shown on Figure 16, an IP-address can be discovered inside the class *ServerConnector*. IP-addresses are not usually considered confidential information such as passwords but can be hidden from source code using vaults or build variables. Using build variables still expose these values to de-compilation, since the build-variable placeholders are replaced by actual values when compiling. The IP address leads to the final server of the challenge which is common for both paths and the exploitation process is described in chapter 4.3.3.

4.3.2 Path 2

Path 2 starts at the same website as path 1, but instead of trying to log into the website, inspecting the network traffic of loaded web-resources reveals another host. As shown in Figure 17, content is loaded from two different sources while loading the website for the service. For the hint to the puzzle, the second source is there to provide the participants another target to start scanning. The second host is highlighted with a failing GET-request to draw attention to it, since this is the early stages of the challenge.

Status	Method	Domain	File
🚫	GET	13.51.72.234	teppo.jpg
304	GET	d1olkaswt1kd48.cloudfront.net	/
	GET	13.51.72.234	teppo.jpg
304	GET	d1olkaswt1kd48.cloudfront.net	scripts.min.js
403	GET	d1olkaswt1kd48.cloudfront.net	favicon.ico

Figure 17. Inspection of network requests on website loading event

Enumerating the IP-address revealed in examining the web-resource hosts, a running FTP-service can be discovered on the server on port 21. The server can be logged in anonymously with the username Anonymous and no password. After logging into the anonymous FTP, an ELF application can be discovered inside. There are three possible intended solutions for extracting the IP address of the final server from the ELF binary.

The first solution is to execute the binary application. The application mentions doing a ping and exits. Ping is network traffic, which is designed to question network devices for existence. Ping packets are usually targeted at a specific host instead of broadcasting. Inspecting network traffic with a monitoring tool such as Wireshark reveals the target of ping, which is the final server.

The second solution is to execute the application with a debugger attached. With a debugger attached, the variables can be read at runtime. Following debugger printouts, the value for the host variable can be observed when the ping function starts.

The third solution is to open the ELF file in a decompiler such as *Ghidra* or Radare2. Examining the source code will reveal a function *ping()*. The function *ping* receives 2 variables, host and *packetcount*. Following the source code to the point in the *main()*-function where the ping()-function is called, the two variables are shown. Three lines up, the values of the host IP-address and the *packetcount* are assigned to the variables.

4.3.3 Common path

In the training, it was emphasized that when you discover a new target, you start the enumeration process again from the beginning. On the final server, there is a http server discoverable. Fuzzing was covered in chapter 2.5. With fuzzing with almost every web-content or directory wordlist, a directory named *backup* can be discovered. In the directory there is a file called a passwd.zip.

The archive is encrypted and there is no password to use. The password hash can be extracted from the archive with tools such as zip2john, which extracts the hash in a format compatible with John the Ripper. After the hash has been extracted, it can be cracked with John the Ripper or a similar tool. Hash cracking has been covered in chapter 2.5. The password for the file is a common word found in several wordlists pre-installed on penetration testing Linux distributions. With the discovered password, the archive can be extracted. The archive contains a passwd file with username and password in a specific hash format. Cracking the hash will yield the login password.

The final server is accessible with the discovered username and password. When the user logs in to the server, the assigned shell is *rbash*, or The Restricted Shell. *Rbash* limits the commands the user can run. I this case, the user was allowed to run the text editor Vi. Vi has a functionality run system commands from a built-in terminal. If a shell binary such as bash in ran through Vi's terminal, the session gets a normal unrestricted shell session. This does not increase the user privileges, just removes the limitations *rbash* had on which commands were allowed to be ran.

After the session is running with a normal shell interpreter such as bash, the server can be enumerated. In the **/opt/data** directory, a file called **stolen_data.txt** can be discovered. This file contains thousands of lines of fabricated personal information and the final flag hidden amongst them.

4.4 Training session

The training session was held as a remote meeting and presentation. The session consisted of presenting the required techniques, with examples, to pass the challenge. The examples were not identical to the solutions to the puzzles, but rather demonstrated the tools or techniques. Using the same tools or techniques to solve the actual puzzles, some minor adjustment to the examples were required. The training session was recorded, so people who could not attend, could still watch it.

The training session structure was to cover each technique needed to solve the challenge. The techniques were accompanied by few suggested tools and an example on how to use one of them. The training session started with a message or call to action text with attempt at humour. "Gofore systems were breached and personal information of at least Seppo Sorsa and Teppo Sorsa were stolen. The Information was shared in Ylilauta. The leak has been removed from the site, but we need to find out what happened and what data was leaked. The hacker had slipped a note under the Gofore Tampere office door …" the slide is shown in appendix 5. The text in Figure 18 men-

tions that data was leaked, and it was posted on Ylilauta, a Finnish discussion forum. The call to action is to find out what happened

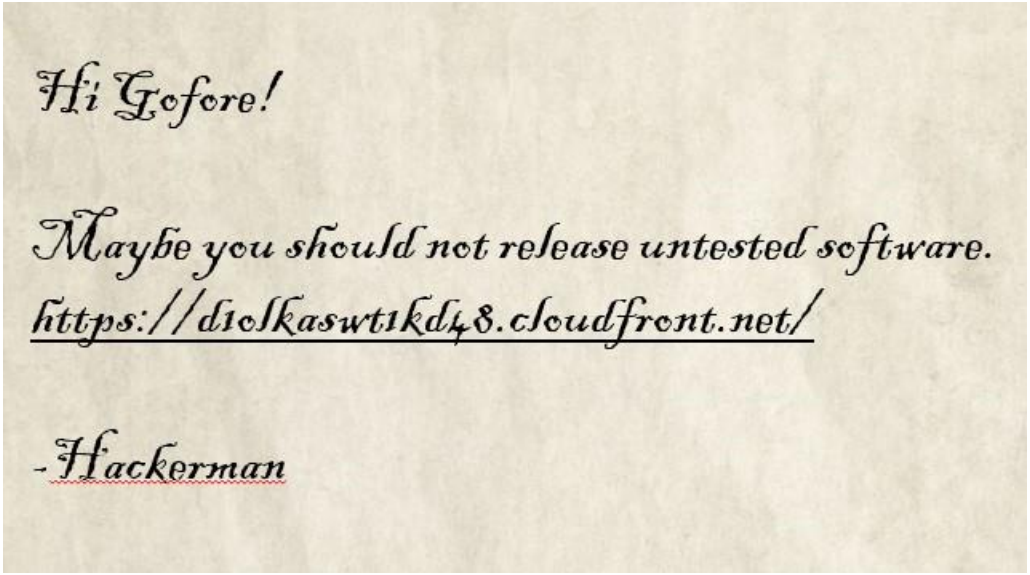


Figure 18. Ransom note

The note (Figure 18) which is mentioned being slipped under Gofore Tampere office door makes a remark on the quality of released software and provides with an URL to the vulnerable web application, which is the first step in the challenge. For employees not participating in the training, the presentation was available on the Slack channel of the challenge. From this introduction the training progressed to teach the techniques and tools.

The topics covered in the training session were what is a CTF challenge, where to send flags when you find them, and all the topics covered in chapter 2.5. Out of CTF concepts, the basics were explained; That the idea of a CTF is to capture an imaginary flag, which is usually a hash text, which is

formatted in such way to prevent guessing. That the idea is to compete in and learn hacking techniques while having fun, and where to find more CTFs if they get excited.

5 Results

When looking at the data in Table 1 , 130 employees participated in the training session. The progress was monitored via Slack by which different channels participants joined. On the first channel, where the exercise started, the highest number of members was 63. 12 employees managed to secure the final flag, meaning they completed some path of puzzles leading to the end of the challenge. A total of 14 employees answered the questionnaire and of those fourteen, 11 (78,6%) had completed the challenge and 3(21,4%) had not, as shown on Figure 19. The results are visualized by Microsoft Forms and summarized in appendix 3.

Table 1. Numerical statistics of participants

	Known to solve the challenge	**Known to start the challenge**	**Total number of participants**
All participants	12	63	130
Questionnaire respondents	11	14	14

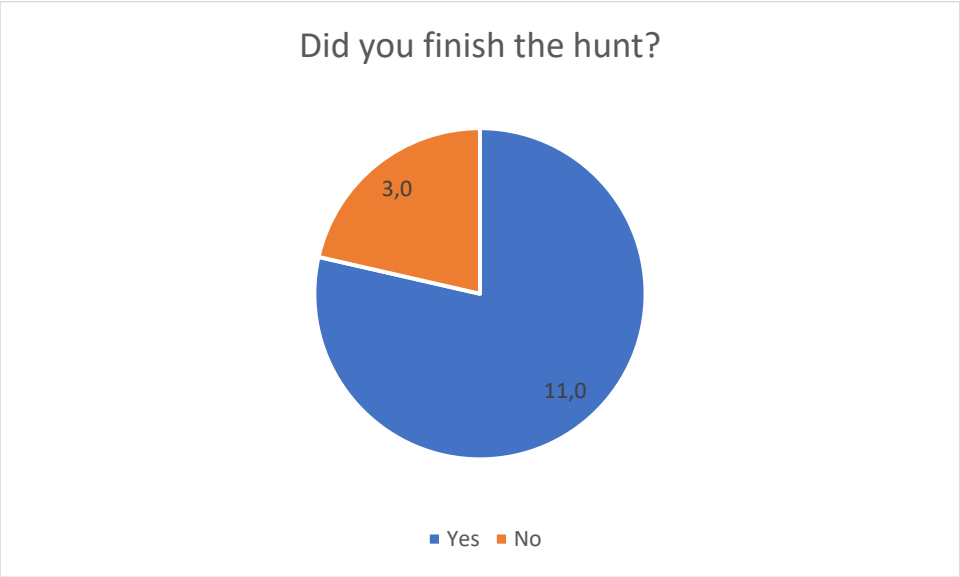


Figure 19. Challenge completion numbers

The number of answers indicate that most of the participants who completed the challenge, answered the questionnaire and most of the participants who did not complete it, did not answer. Out of 12 participants who completed the challenge, 11 answered the questionnaire. From the 63 participants in total, 14 answered the questionnaire, and 11 had completed the challenge. Out of the 51 participants who did not complete the challenge, 3 answered the questionnaire, leaving 48 participants who did not answer. The number of answers is too small for any proper conclusions, the original hypothesis cannot be confirmed or disputed.

5.1 Respondents' statistics

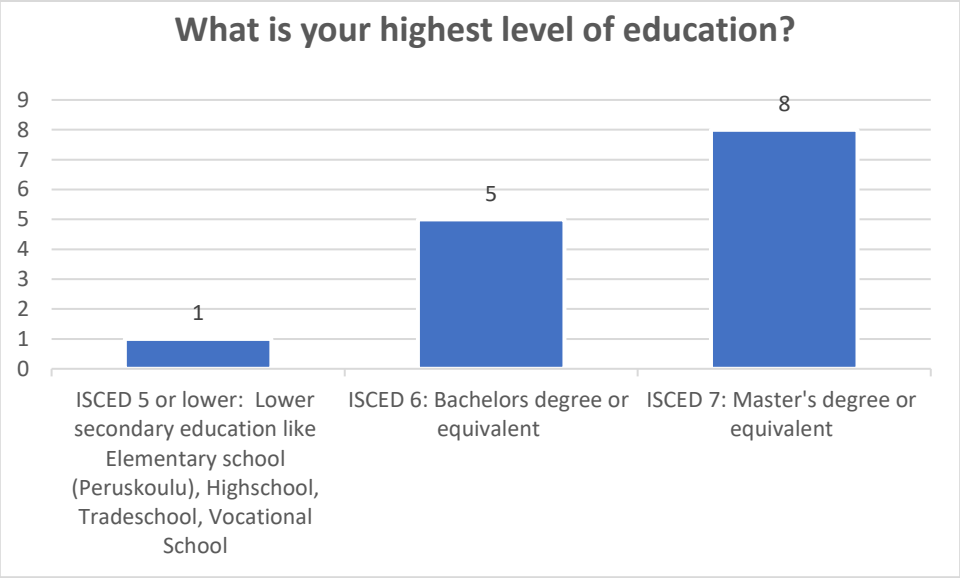


Figure 20. Level of education of questionnaire respondents

Over half of the participants who answered the questionnaire had ISCED level 7 education as is visible from Figure 20. ISCED level 7 is a master's degree of equivalent. Over third of the respondents had ISCED level 6 as their highest level of education. ISCED level 6 is a bachelor's degree or equivalent. Only one respondent answered having ISCED level 5 or lower. ISCED level 5 is a lower secondary education, highschool, tradeschool or a vocational school. (*The Structure of the European Education Systems 2018 / 19 : Schematic Diagrams*, n.d.)

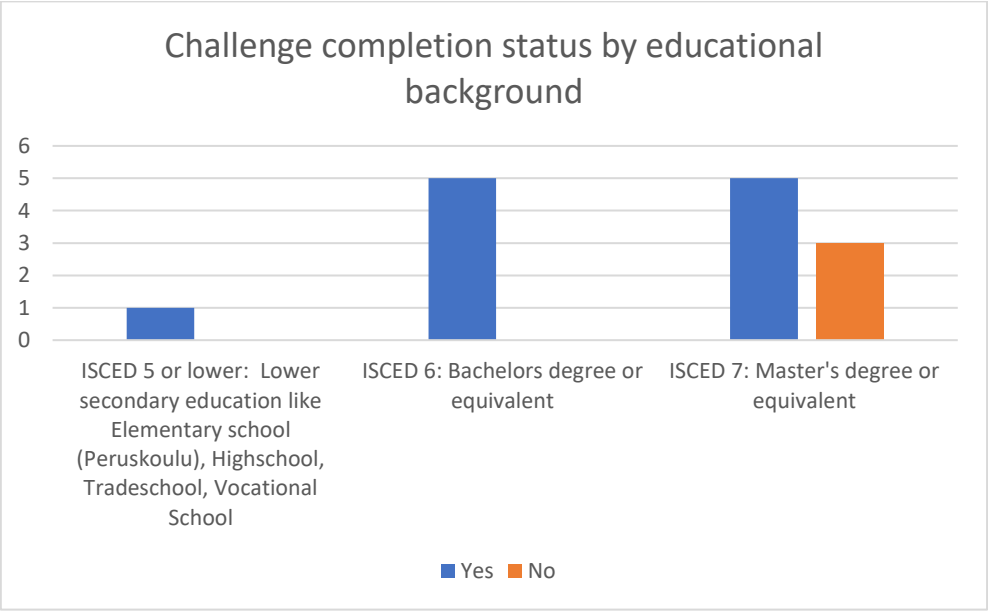


Figure 21. Challenge completion status by educational background

The impact of educational background on the results could not be confirmed or disputed, since in addition to the low number of responses, the 3 respondents who reported not having finished the challenge, were all with an ISCED level 7 degree, which was the most represented education level as shown in Figure 21. The group who answered the questionnaire was quite homogenous in regards of level of education, which is not unexpected from a technology consulting company. The number of respondents who completed the challenge was equal in both groups with ISCED level 6 or ISCED level 7 educational background. Since ISCED level 5 educational level had only one respondent and the ISCED level 6 and 7 had the same number of solvers for the challenge, the data offers no answers to the research questions.

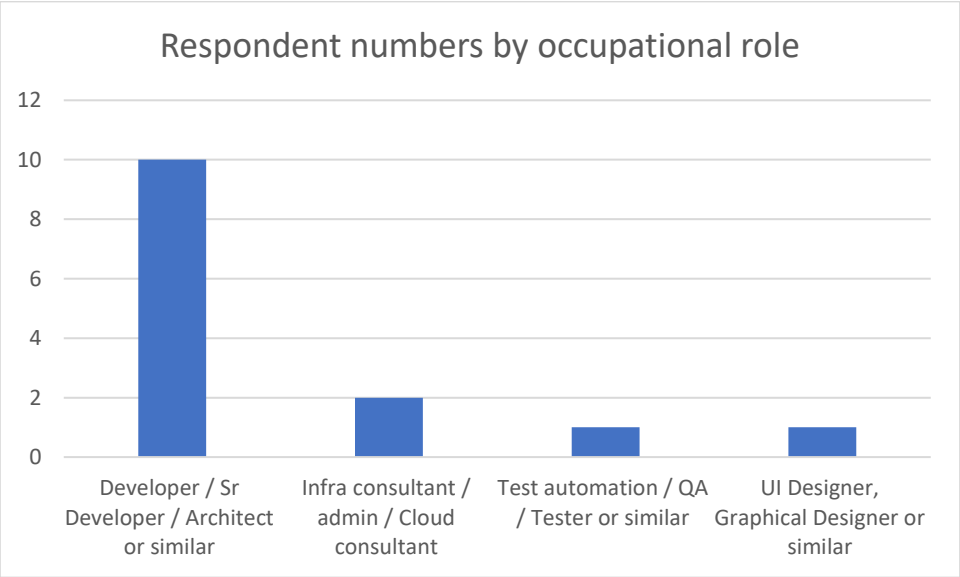


Figure 22. Respondent numbers by occupational role

Most of the respondents were developers, senior developers, architects or similar by occupation. The small representation of other roles is visible in Figure 22. The overrepresentation of a single occupational group distorts the already small result set. The overrepresented role group is considered technical. No conclusions can be drawn from the limited amount of data with one group overrepresented.

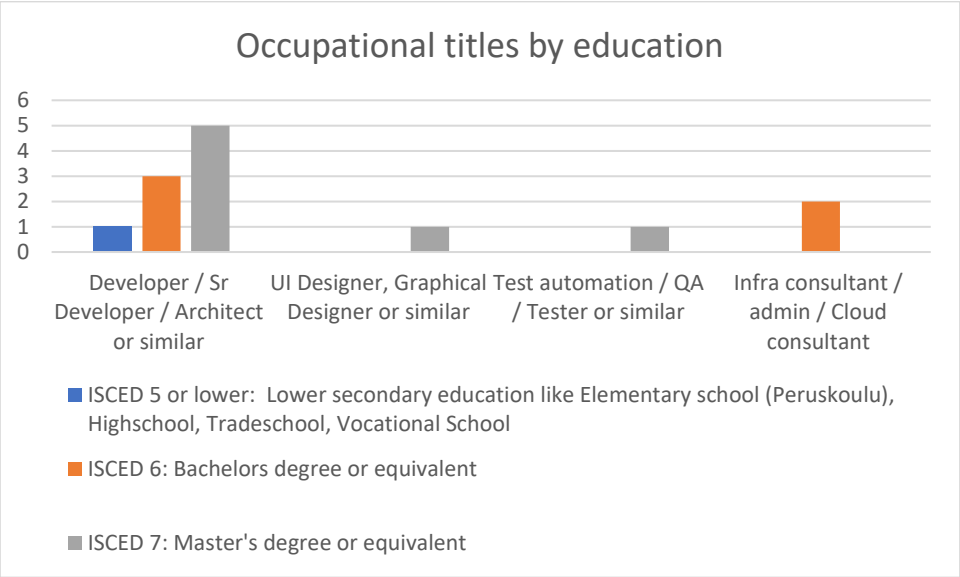


Figure 23. Occupational titles by educational level

In Figure 23 the overrepresentation of Developers, Senior developers and Architects is visible. Overrepresentation is not surprising in a company a large portion of employees belong to that role group and the nature of the challenge is closest to the expertise of those occupations right after cyber security consultants, of which none of participated in the challenge. The highest represented education level ISCED level 7 is also most represented in the developers and architects group. As the distribution of educational backgrounds in the whole group of respondents is nearly the same as the distribution inside the developers and architects occupational group, the same group represents majority of responses in both educational and occupational comparison.

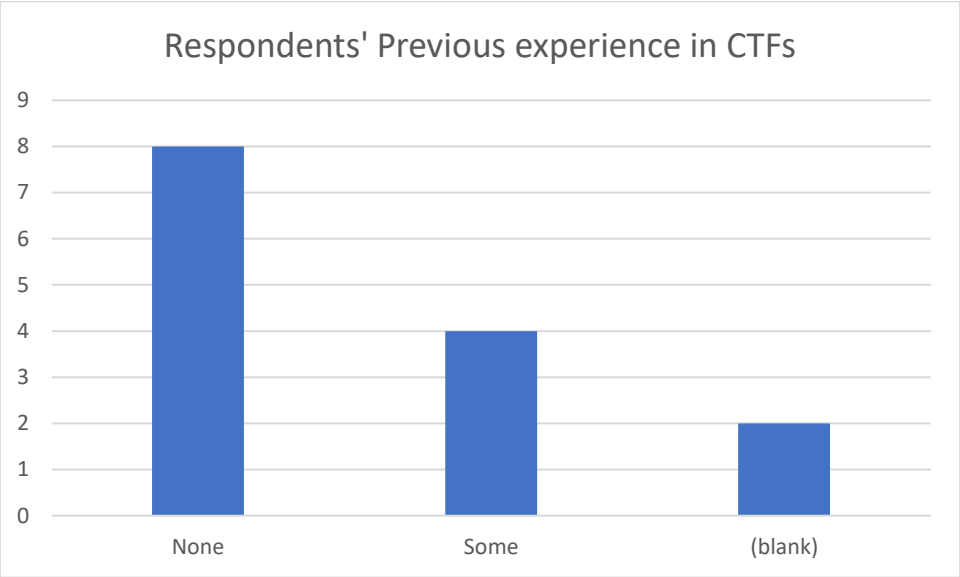


Figure 24. Respondents' Previous experience in CTFs

Respondents did not report having significant experience in CTFs. Out of the 14 respondents, 8 reported having no experience, 4 having some experience and 2 did not answer the question as shown in Figure 24. No respondent answered having “Alot” of experience in CTFs, which was the third answer option as seen on appendix 1.

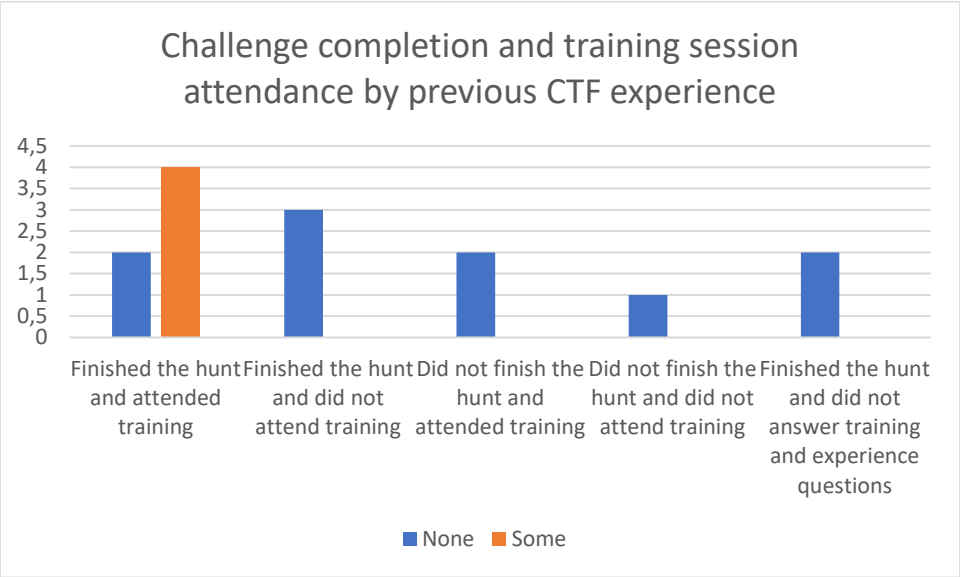


Figure 25. Challenge completion and training session attendance by previous CTF experience

Most of the respondents who answered having finished the hunt and attending the training session had some previous CTF experience as illustrated in Figure 25. 2 respondents who finished the hunt and attended the training session did not have any previous CTF experience. Respondents who had previous CTF experience assumably anticipated that participating in the training is beneficial in discovering the right tools and techniques. Respondents who finished the hunt without participating in training or did not finish the hunt regardless of participating in the training all had no previous CTF experience.

5.2 Respondents' subjective answers

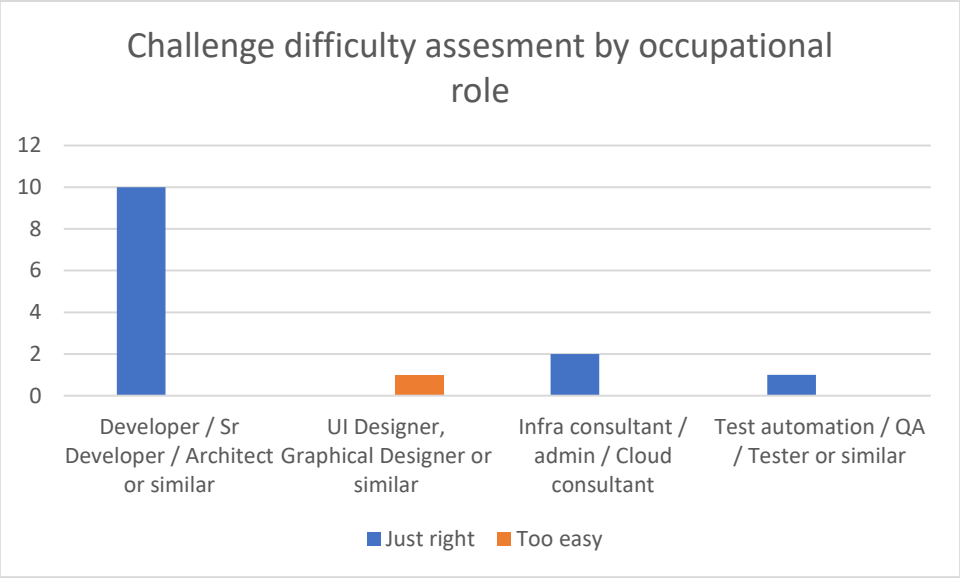


Figure 26. Challenge difficulty assesment by occupational role

The difficulty of the challenge was inquired as part of the questionnaire. This question was to be used to assess if the challenge was too difficult for mainly non-technical participants. Only one respondent reported the challenge to be "too easy" and the occupational role was non-technical. As is visible in Figure 26. As only one respondent considered the challenge too easy and most of the respondents considered it being of an appropriate difficulty, the challenge can be considered to be at the correct difficulty level. Lack of respondents who did not complete the challenge may distort the result.

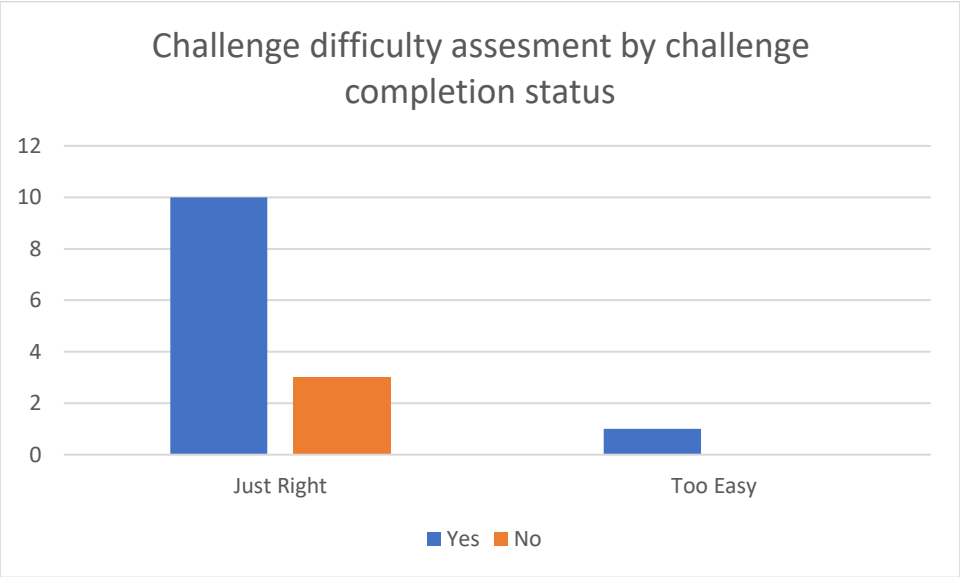


Figure 27. Challenge difficulty assesment by challenge completion status

The respondents who did not complete the challenge, report the challenge difficulty being "just right", as illustrated in Figure 27, which can be interpreted that they did not finish for some other reason than the challenge being too difficult. The challenge remained open for two months, to be accessible for participants to solve in their own pace. The timing of the event was during the autumn, which can be busy in the IT industry, which could have affected the amount of time participants were able to use on the challenge.

This question is not very accurately phrased, and the answers are based on the subjective experiences and opinions of the respondents. The motivation for the question was to have some reference to the other question of how difficult the CTF was considered. If experienced CTF hobbyists would have considered the challenge too difficult, or if participants with no experience had considered it too easy, the teaching aspect as well as the game aspect, would have suffered.

In the open feedback portion of the questionnaire, more non-technical routes were requested as well as more guidance. Most of the open feedback was generic appreciation for hosting such an

event. Some hacker-minded people had responded that they had searched for the code of the Slack App bot in Gofore shared Git-repository to get the flags.

5.3 Puzzle solving results

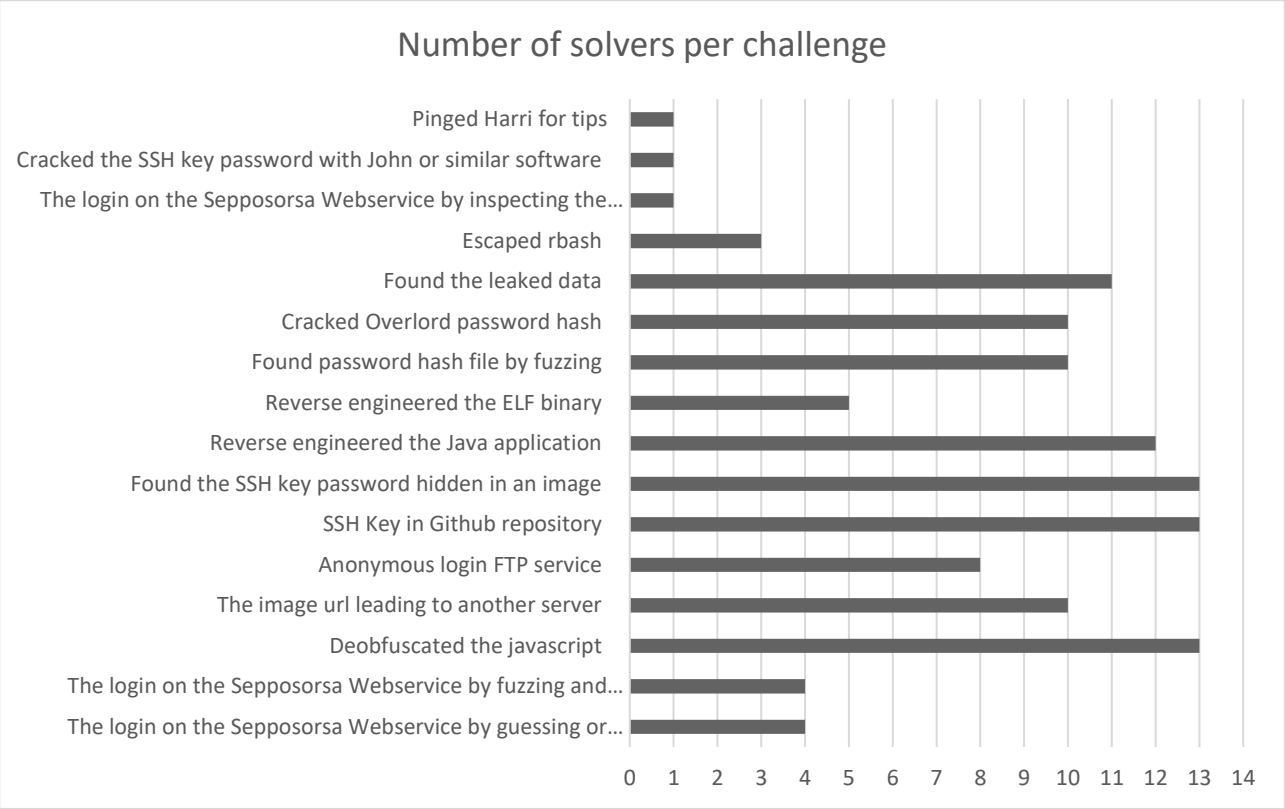


Figure 28. Number of solvers per challenge

From the puzzles solved, both purely technical as well as the less-technical solutions were reported to have been used. Figure 28 shows that the most solved puzzles were the steganography challenge of finding the SSH key password hidden in the image, finding the corresponding SSH key in the GitHub commit history and de-obfuscation of the JavaScript on the first vulnerable webpage. Solutions on path 1 such as GitHub related puzzles and Java application puzzle were solved by most whereas solutions in path 2 such as ELF binary reversal was solved was solved by less than half.

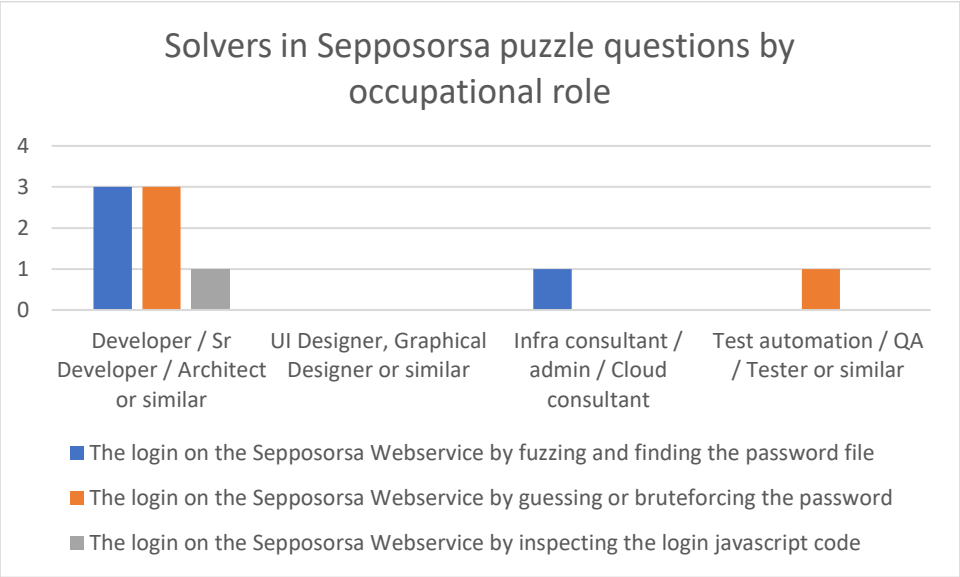


Figure 29. Solvers in Sepposorsa puzzle questions by occupational role

The answer choices “The login on the Sepposorsa Webservice by fuzzing and finding the password file”, “The login on the Sepposorsa Webservice by guessing or bruteforcing the password” and “The login on the Sepposorsa Webservice by inspecting the login javascript code” had overlap with the answer choice “Deobfuscated the javascript” and possibly were mixed up with “Found the password file by fuzzing” and which answer option related to which puzzle was possibly misunderstood by the respondents, since they got very little answers as shown in Figure 29. The questions were badly formed.

5.3.1 Path 1 puzzle questionnaire answers

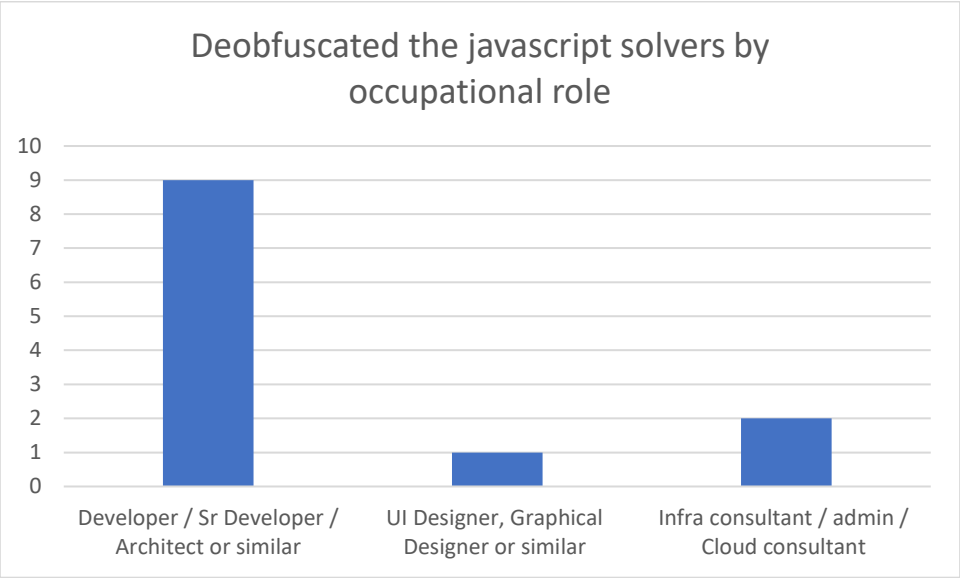


Figure 30. Deobfuscated the javascript solvers by occupational role

13 respondents solved the **De-obfuscated the JavaScript** puzzle, which was one of the options for the first puzzle for path 1, as shown in Figure 30**.** Only one respondent did not list the puzzle as solved. Since most of the respondents had answered having solved the puzzle, the distribution of solvers by occupational roles is nearly the same as the distribution of respondents in total.

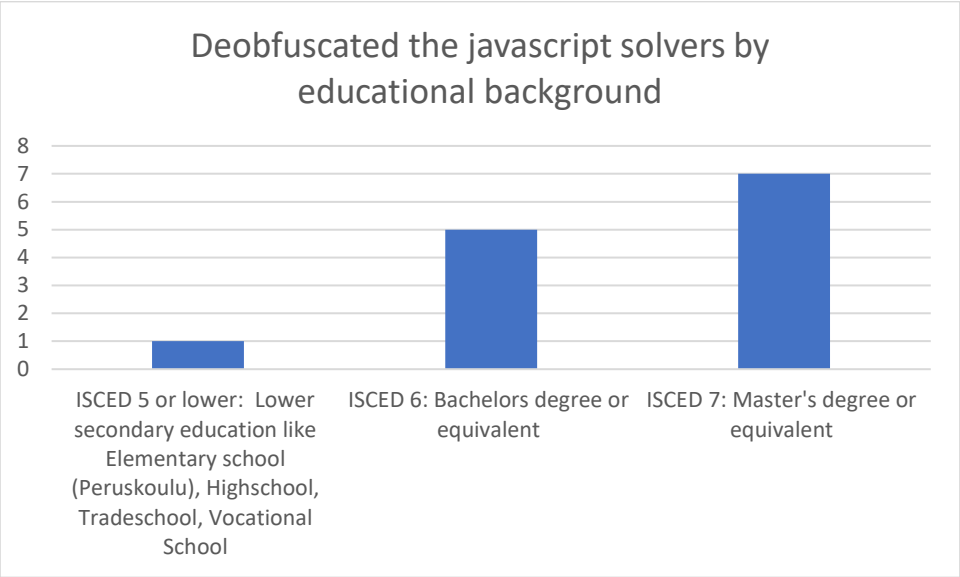


Figure 31. Deobfuscated the javascript solvers by educational background

The only respondent not having reported solving the JavaScript de-obfuscation puzzle was of the ISCED level 7 group as can be calculated from data from Figure 31 and Figure 20 where seven respondents of ISCED level 7 group have answered solving the puzzle and total number of respondents with ISCED level 7 education was eight, leaving the difference of one. Since the only respondent not having reported solving the puzzle, the distribution of solvers by educational background is nearly the same as distribution of correspondents in total.

The total number of respondents answering having solved the JavaScript de-obfuscating puzzle supports the assumption that puzzles requiring techniques taught it the teaching session were solvable for most participants.

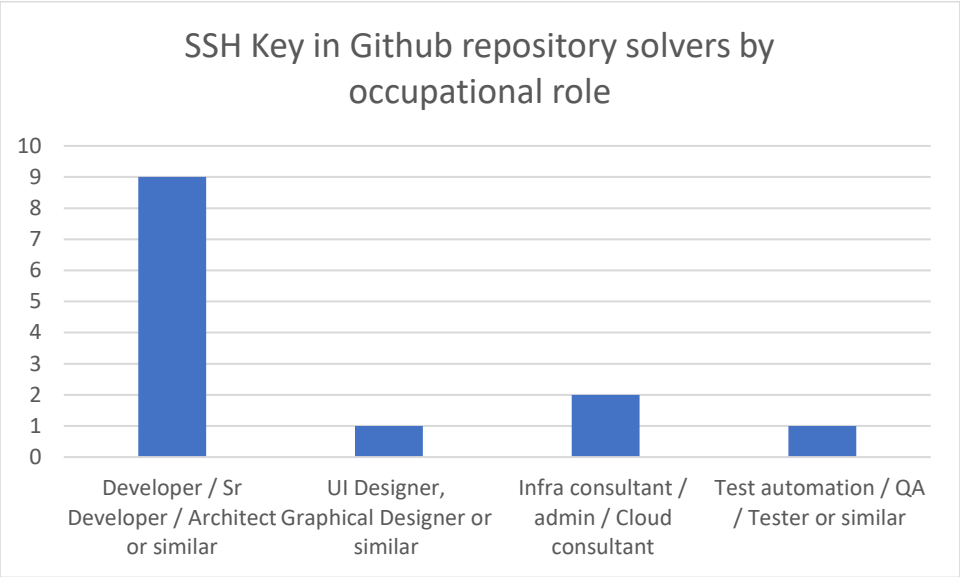


Figure 32. SSH Key in Github repository solvers by occupational role

13 respondents answered having solved the **SSH key in GitHub repository** puzzle. Only one respondent did not answer having solved the puzzle. When comparing data from Figure 21 and Figure 32 it indicates that the only respondent answered not having solved the puzzle is from the tester occupational group, which is considered technical. Since nearly all respondents had answered having completed the puzzle, the impact of occupational role cannot be seen on this puzzle.

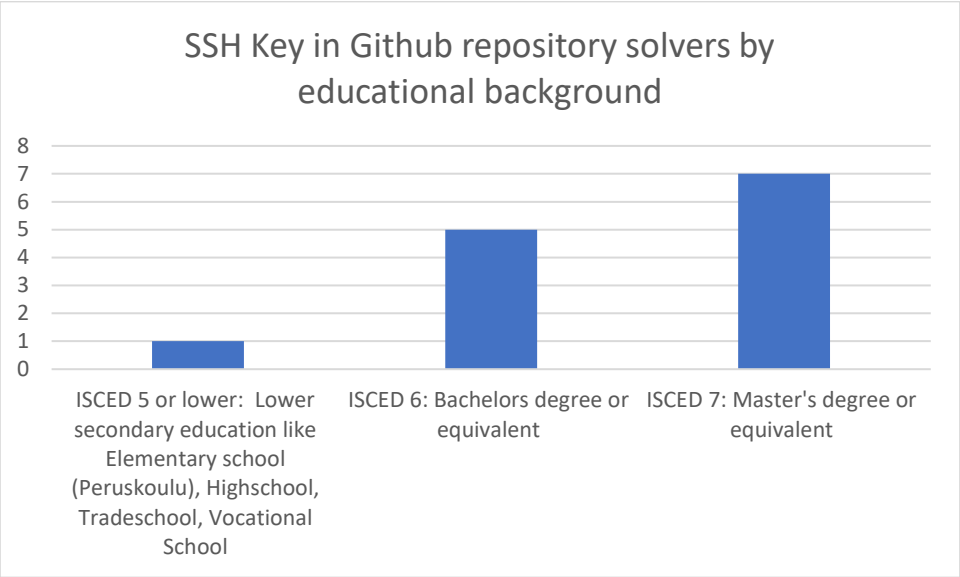


Figure 33. SSH Key in Github repository solvers by educational background

One respondent answered not having solved the SSH GitHub puzzle is from the ISCED level 7 group as can be seen comparing data from Figures Figure 20 and Figure 33. Since the only respondent who answered not having solved the puzzle is from the most represented educational level, no conclusions can be drawn from the data in relation of educational level affecting the solving of the SSH GitHub puzzle.

The total number of respondents having answered solving the password file puzzle supports the assumption that puzzles requiring techniques taught it the teaching session were solvable for most participants.

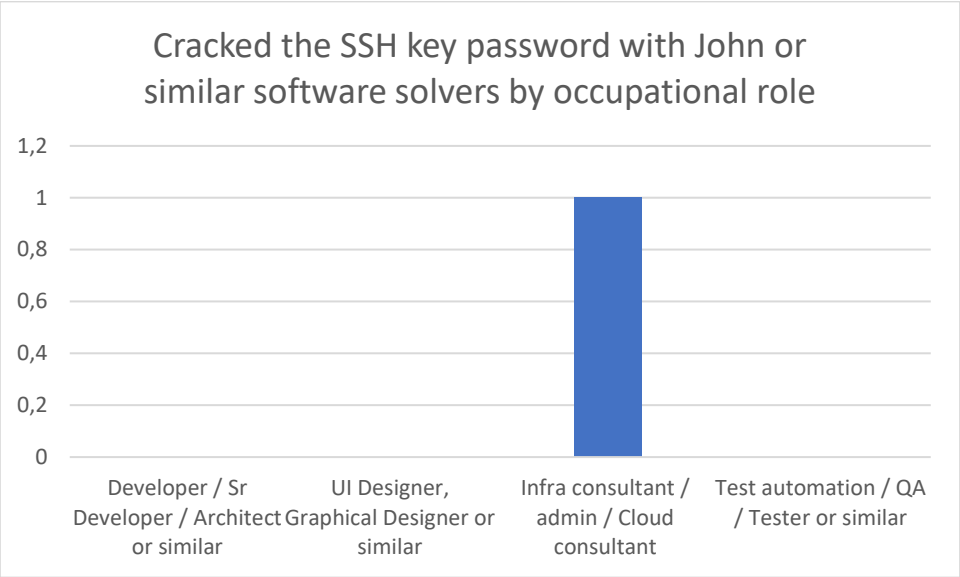


Figure 34. Cracked the SSH key password with John or similar software solvers by occupational role

Only one respondent answered solving the **Cracked the SSH key password with John or similar software.** As shown on Figure 34 and Figure 35, the only respondent was from the infrastructure, admin or cloud consultant occupational group. This group is considered technical. What can be concluded from the data is that cracking the password was not a popular choice. Cracking was taught in the teaching session at the start of the event.

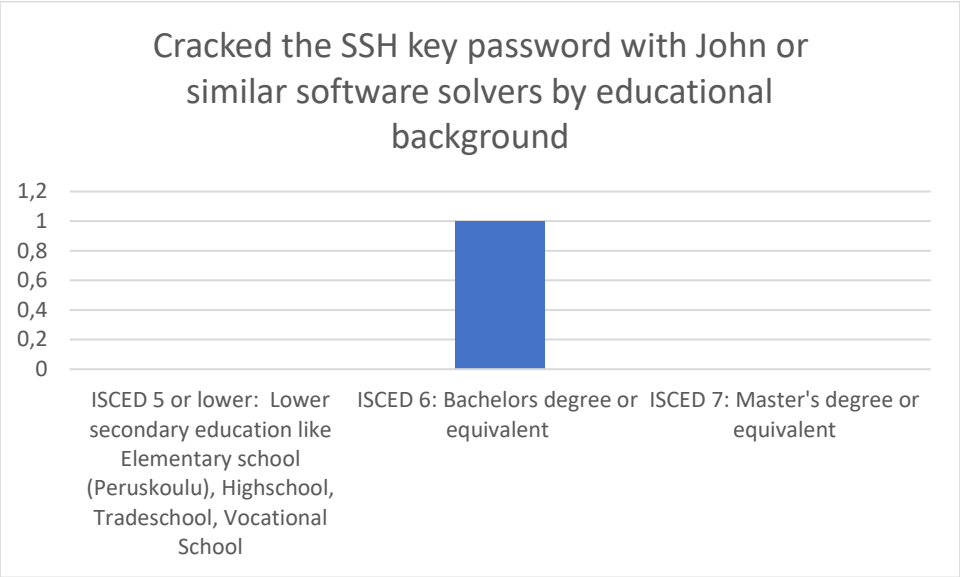


Figure 35. Cracked the SSH key password with John or similar software solvers by educational background

The only person that answered having used John to crack the SSH key password, had ISCED level 6 degree. The other way of solving the password, was more used as can be seen in Figure 36. This puzzle offers no other interesting data, except that the puzzle was not very popular. The hint for the next puzzle could have been too obvious.

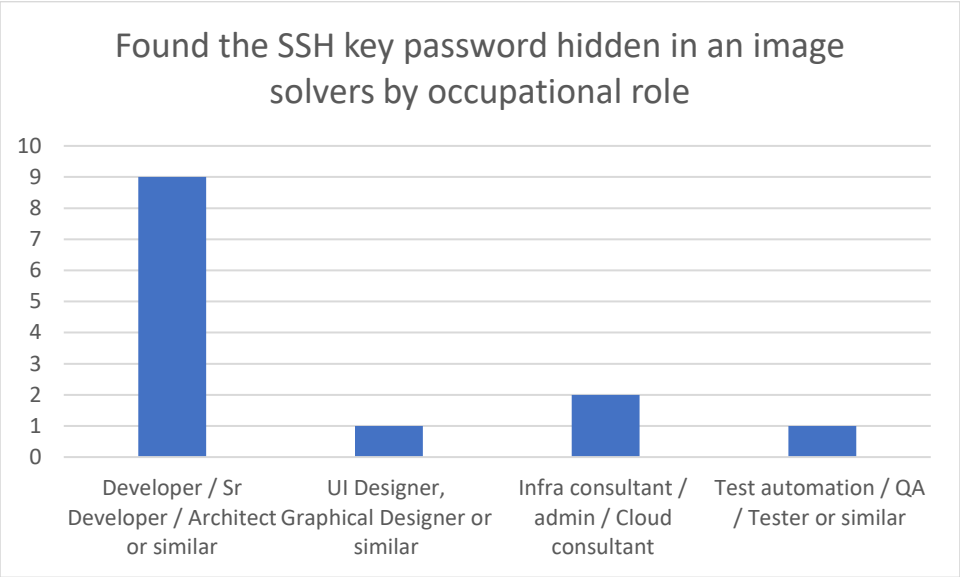


Figure 36. Found the SSH key password hidden in an image solvers by occupational role

Thirteen respondents answered having solved the steganography challenge **"Found the SSH key password hidden in an image"**. This is the same number of respondents that solved the previous steps on path 1 which can be seen when comparing Figure 30 and Figure 36. This puzzle was an alternative to cracking the key password with john the ripper or similar tool. Having nearly the whole group solve the puzzle gives no insight on effect of occupational role on solving the puzzle.

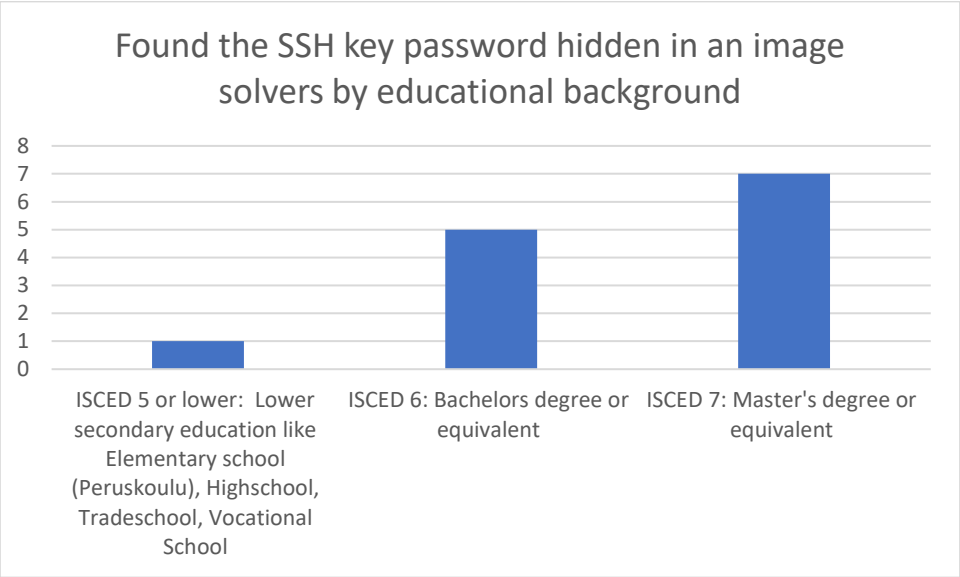


Figure 37. Found the SSH key password hidden in an image solvers by educational background

The distribution of educational backgrounds amongst the solvers of the puzzle differentiates only slightly from the distribution amongst the whole group of respondents since only one respondent did not solve the puzzle and had the most represented educational level degree.

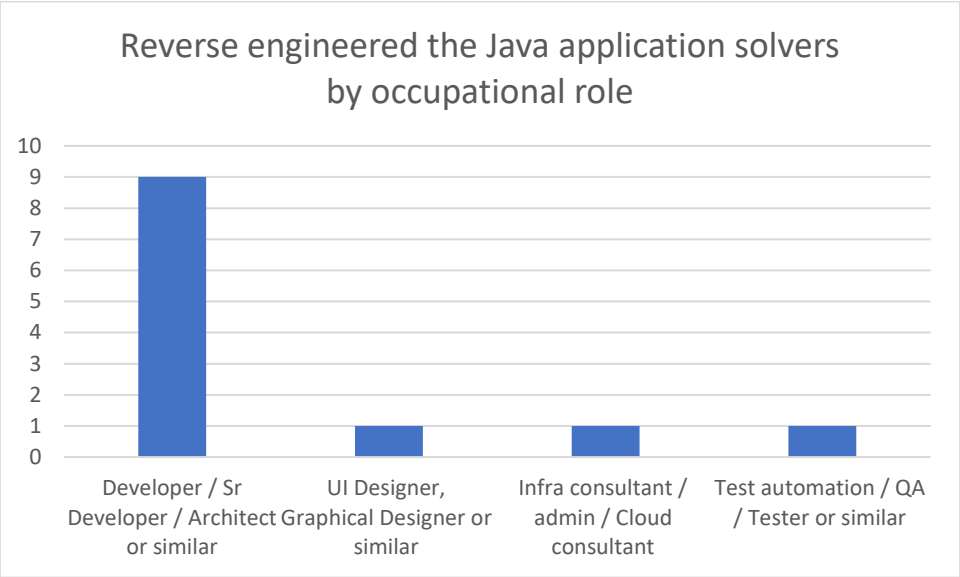


Figure 38. Reverse engineered the Java application solvers by occupational role

Twelve respondents answered having solved the **"Reverse engineered the Java application"** puzzle as is visible on Figure 38 and Figure 39. This is one less than on the previous steps of path 1. The distribution between occupational roles remains nearly the same, but one respondent from the automation occupational group, who had solved the previous puzzles in path 1, had not solved this puzzle. No solid conclusions can be drawn from the data.

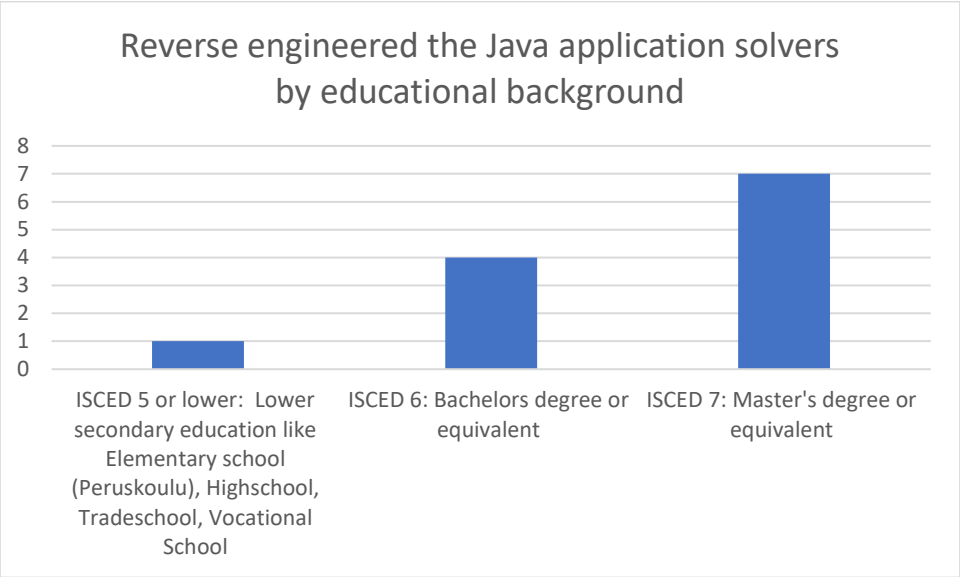


Figure 39. Reverse engineered the Java application solvers by educational background

The twelve respondents who answered having solved the Java reverse engineering puzzle resemble the total group of respondents by educational background, since the two respondents who did not solve the puzzle, were from the two best represented educational levels: ISCED levels 6 & 7. The only respondent who solved previous levels, but not this one, was from the most represented educational level of ISCED level 7.

5.3.2 Path 2 puzzle questionnaire answers

Path 2 was less used than path 1. Path 1 used more obvious hints where path 2 solutions were more easily discovered with tools of the trade. The first puzzles on path 2 did not require extensive technical skills. Later puzzles on each path were designed to be technically equally challenging up to the common path.

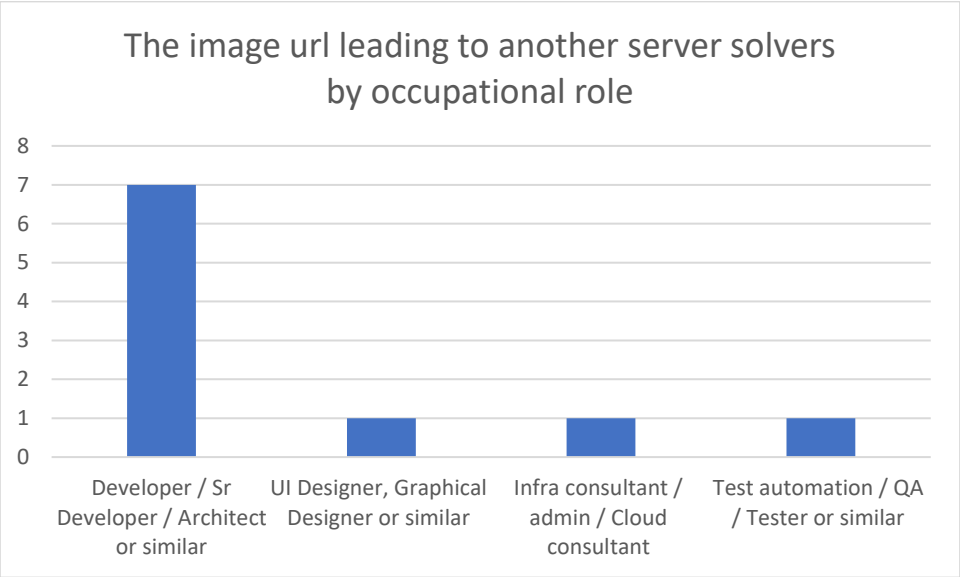


Figure 40. The image url leading to another server solvers by occupational role

The image URL leading to another server was the first puzzle on path 2. Ten respondents answered having solved the puzzle. This is less than on the first level puzzles on path 1 where thirteen respondents had solved some of the puzzles, this can be seen comparing data on Figure 30, Figure 32 and Figure 40. Developer occupation group is best represented, but all other groups have one solver each as well. Most respondents who answered not having solved the puzzle, are from the developer occupational group. Comparing percentage of solvers per group is not useful for groups as small as these.

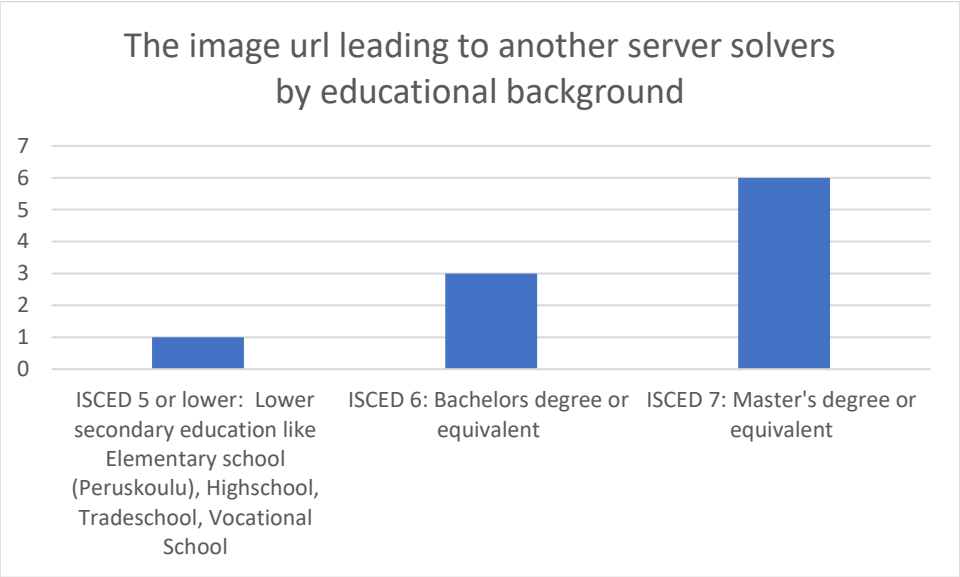


Figure 41. The image url leading to another server solvers by educational background

No significant difference between the distribution of educational background between solvers of the puzzle and the whole respondent group when comparing Figure 22 and Figure 41. Most of the solvers were from the most represented educational level, ISCED level 7. No conclusions can be drawn from the data.

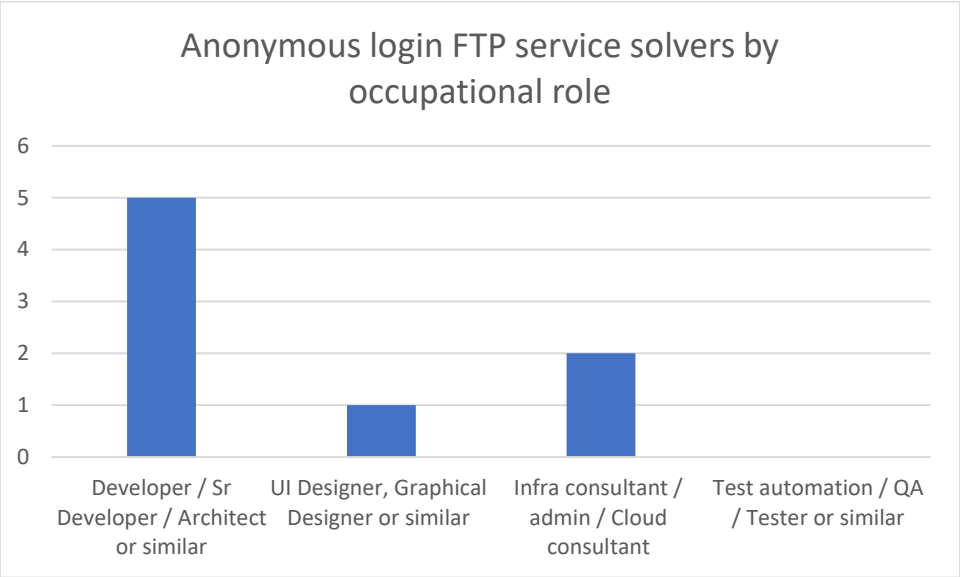


Figure 42. Anonymous login FTP service solvers by occupational role

The **anonymous login FTP server**, which was the second puzzle of path 2, was solved by 8 respondents as illustrated on Figure 28. The distribution of solvers by occupation appears similar to the distribution on the whole challenge with individual variation considered when comparing data in Figure 42. Some respondents solved puzzles from both paths.

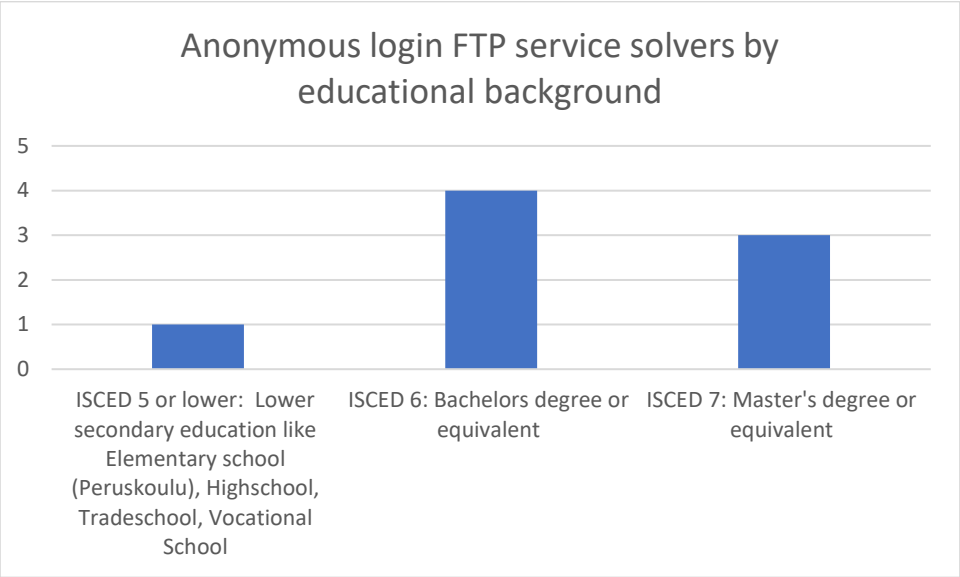


Figure 43. Anonymous login FTP service solvers by educational background

The respondents who answered having solved the anonymous FTP puzzle are mostly from the educational group who have ISCED level 6 education. Only one respondent with that educational level did not solve the puzzle, where five respondents from ISCED level 7, which is most represented in the whole group of respondents, did not solve the puzzle. ISCED level 7 is underrepresented amongst the solvers of the puzzle when comparing Figure 43 with the total number of ISCED level 7 respondents in Figure 22.

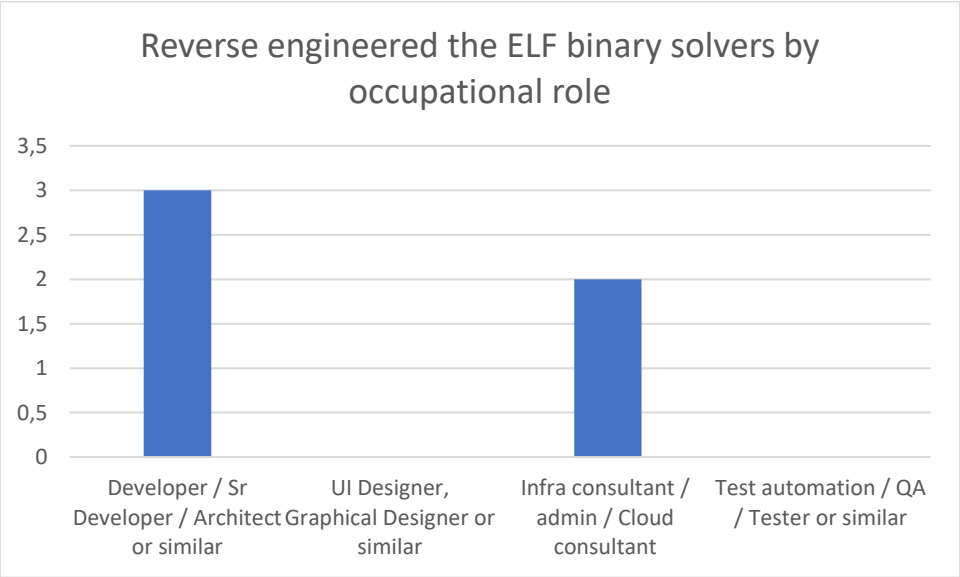


Figure 44. Reverse engineered the ELF binary solvers by occupational role

Only five respondents reported having solved the **Reverse engineered the ELF binary** puzzle as shown on Figure 44. This is significantly less than on the previous step on path 2. The only respondent with a non-technical occupational role did not continue the path 2 further. All five respondents who continued path 2 were of technical occupational groups. The small number of solvers and the only non-technical respondent not having solved the puzzle gives indication that reverse engineering a binary was considered more difficult than previous steps on path 2 or puzzles on path 1.

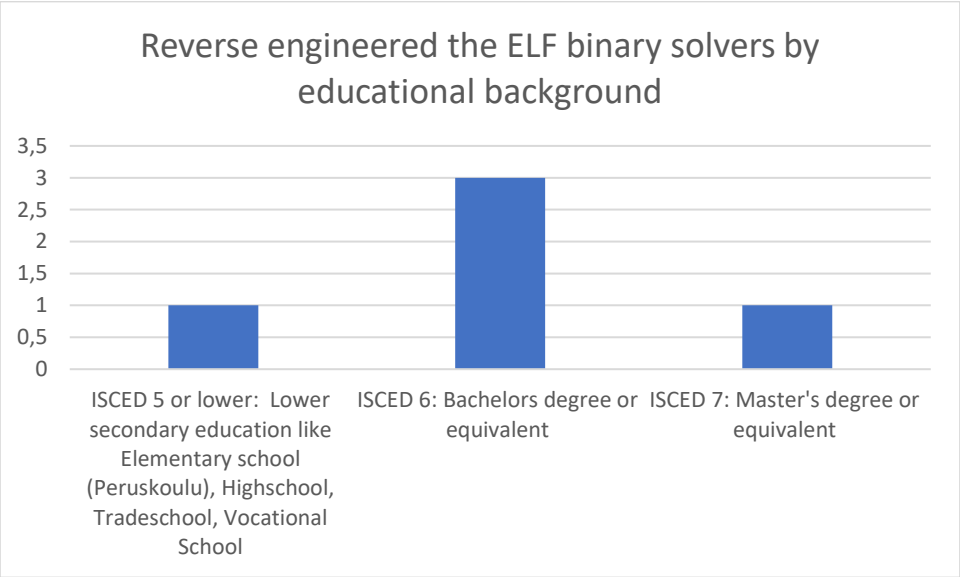


Figure 45. Reverse engineered the ELF binary solvers by educational background

Only one respondent with ISCED level 7 education reported having solved the puzzle. The distribution of educational background amongst the solvers of the ELF reverse engineering puzzle gives no support to the hypothesis that better educated participants would prefer the puzzles requiring more technical solutions. The only respondent with ISCED level 5 degree reported having solved the puzzle, but only one respondent from the most represented educational group, which is also the highest level of education amongst the respondents, ISCED level 7 reported having solved the puzzle. ISCED level 7 is significantly underrepresented amongst the solvers of the puzzle when comparing the total number of respondents in that group from Figure 20 and solvers from Figure 45.

Since the questionnaire had missing answer options for solving the ELF-binary puzzle either by running it and monitoring network traffic or debugging the application, participants could have solved the puzzle with those methods and would have not chosen the “Reverse engineered the ELF binary” option from the possible answers.

5.3.3 Common path puzzle questionnaire answers

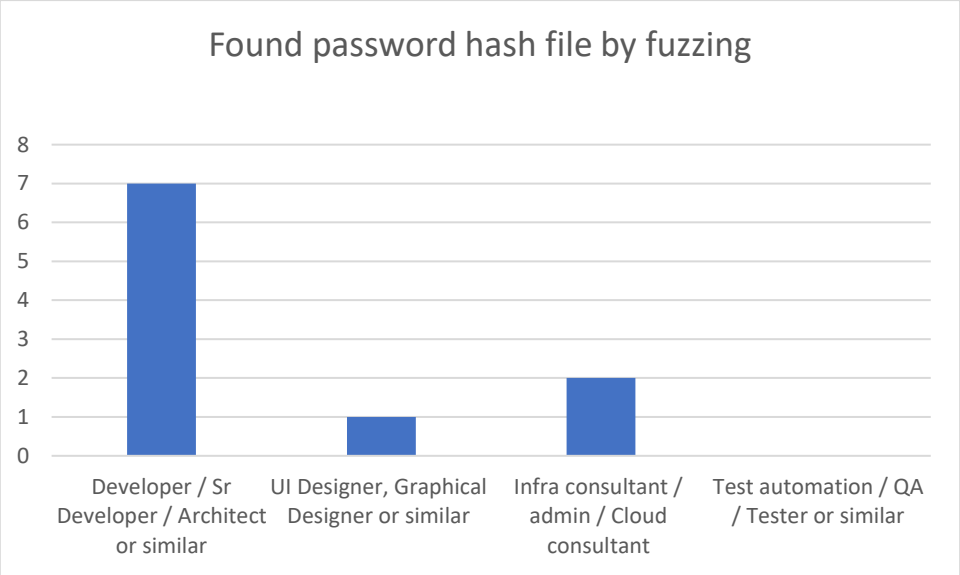


Figure 46. Found password hash file by fuzzing solvers by occupational role

10 respondents answered having solved the **Found the password file by fuzzing** puzzle, which was the first puzzle in the common path. When comparing data in Figure 46 and Figure 22, the distribution of occupational roles of the solvers of the password file fuzzing puzzle is nearly the same as the distribution of occupational roles in total. Since the distribution is similar and the dataset is small, no conclusions can be drawn and answers to the research question regarding occupational role can be formed. The respondents in the QA occupational group did not solve this puzzle or latter ones.

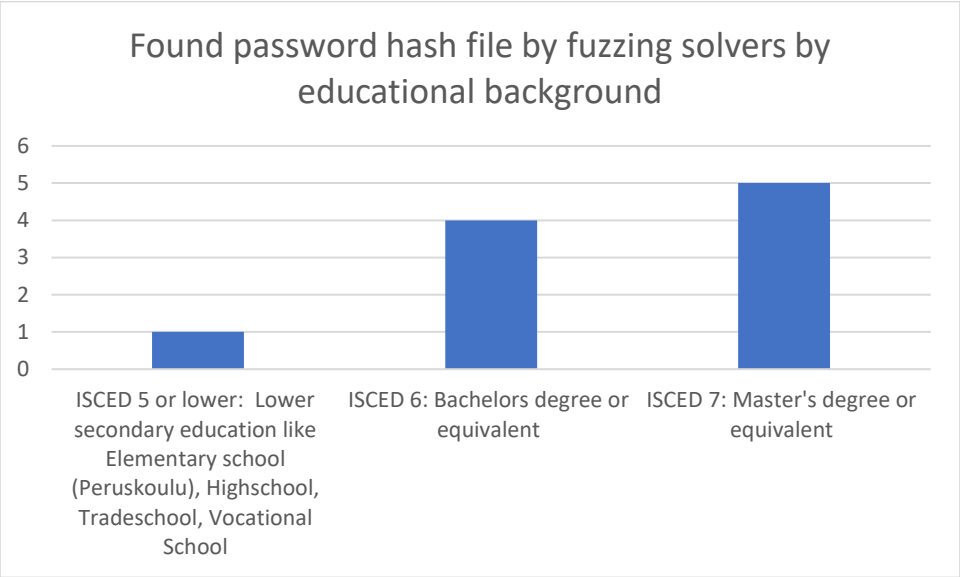


Figure 47. Found password hash file by fuzzing solvers by educational background

By comparing data from Figure 47 and Figure 20 it is show that all respondents answered not having solved the fuzzing puzzle are from ISCED levels 6 & 7 where three were from level 7 and one from level 6. The distribution of educational backgrounds of respondents having solved the puzzle remains nearly the same as the distribution of educational backgrounds in total. As the distribution of the educational background remained similar, the result does not offer any insight on the on the effect of educational background for this puzzle.

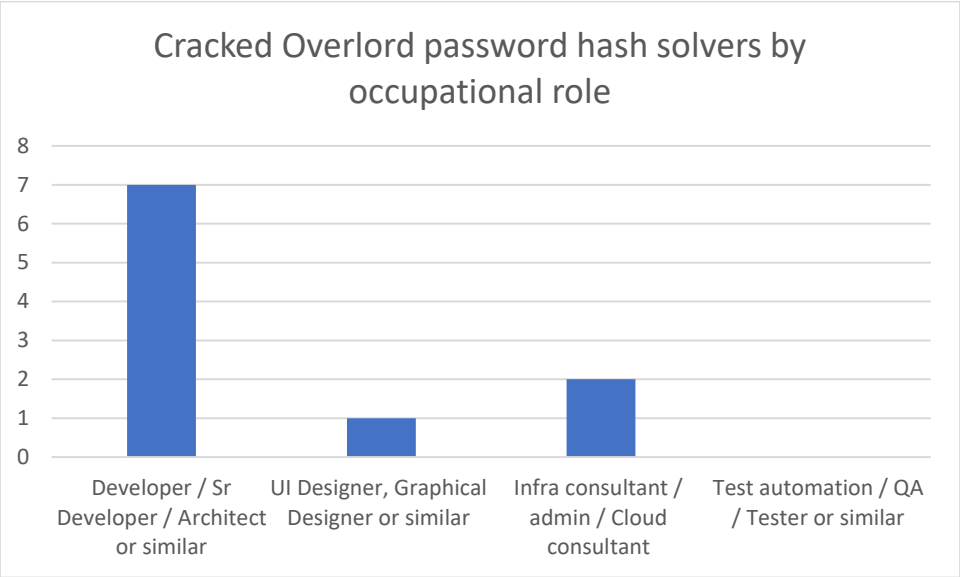


Figure 48. Cracked Overlord password hash solvers by occupational role

The same number of respondents, ten, answered having solved the **"Cracked Overlord password hash"** puzzle, that had solved the previous puzzle of finding the backup file containing the hash. The composition of the group has remained the same since the QA occupational group respondents did not proceed further after the paths merged. No new conclusions can be drawn from the data since the group remains the same and is comprised of both technical and non-technical participants.

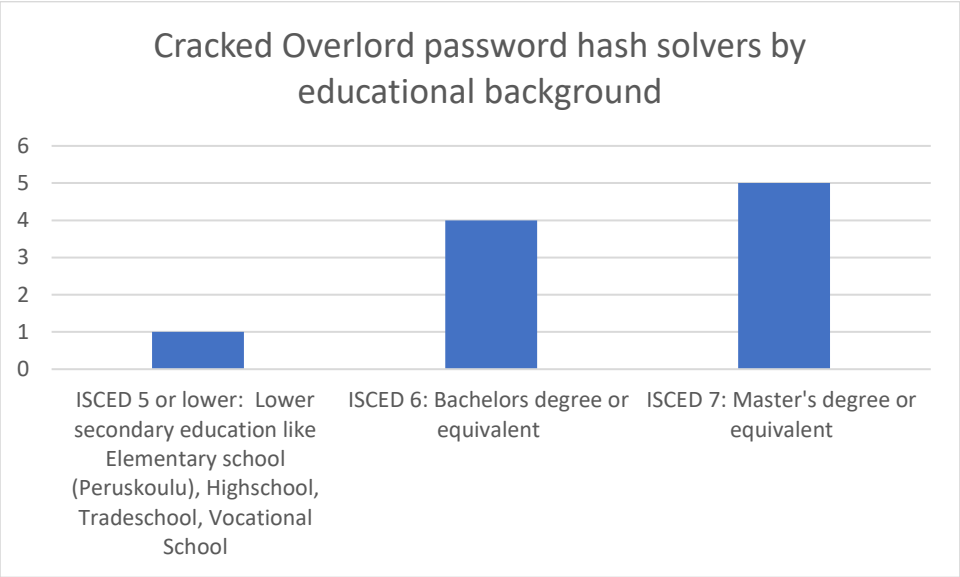


Figure 49. Cracked Overlord password hash solvers by educational background

As the same respondents who answered having solved the previous puzzle answered having solved this one as well, the composition of the group remains the same and the distribution of educational level remains the same as well. No new conclusions can be drawn from the data since all educational levels are still represented and the one which has diminished the most is the level with most respondents and is also the highest educational level of all respondents, ISCED level 7.

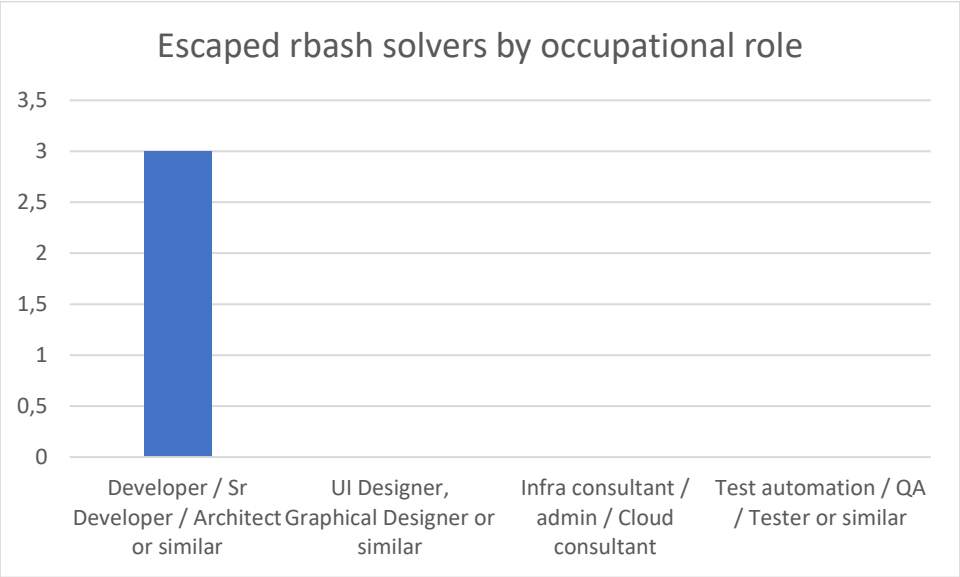


Figure 50. Escaped rbash solvers by occupational role

For the **"Escaped rbash"** puzzle, the number of respondents who reported having solved the puzzle dropped significantly. Only three respondents reported having solved the puzzle, which is the same number of respondents who reported having finished the whole challenge as visible on Figure 19 and Figure 50. The remaining respondents were all from the developer occupational group, which is a considered a technical occupational group. The only non-technical respondent did not solve this puzzle. Having only technical participants left at this stage according to the questionnaire answers is in line with the hypothesis of technical employees preferring technical solutions but does not confirm it since the group is so small and there were no non-technical solutions available at this stage of the challenge for the non-technical participants to choose from.

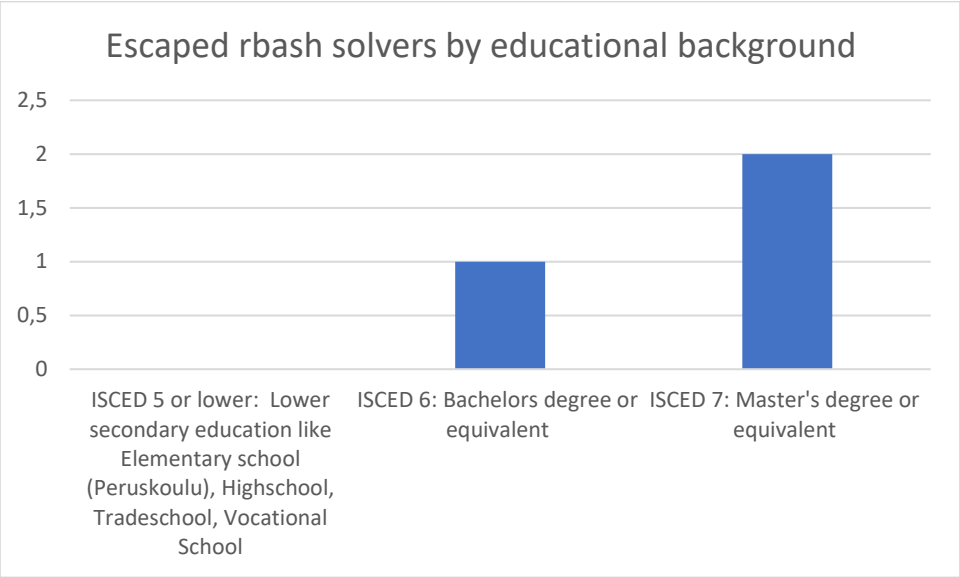


Figure 51. Escaped rbash solvers by educational background

The only respondent with ISCED level 5 education did not report having solved the *rbash* puzzle. The remaining respondents who reported having solved it, were from ISCED levels 6 & 7 as visible on Figure 51. With the number of respondents dropped to three, individual variation in skills affects the outcome. The result is in line with the hypothesis that employees with higher level of education advance further in the challenge but does not confirm it since the remaining group is too small and there was only one respondent with ISCED level 5 education and there were respondents with both ISCED level 6 & 7 education in the remaining group.

The total number of solvers for this puzzle would indicate that the puzzle was more difficult or that participants had used the techniques taught in the teaching session previously in the challenge. Escaping restricted command interpreters was not covered in the training session. The process is usually quite simple and there are several online resources that give detailed instructions for several applications including *Vi* used in this puzzle. (*GTFOBins*, n.d.)

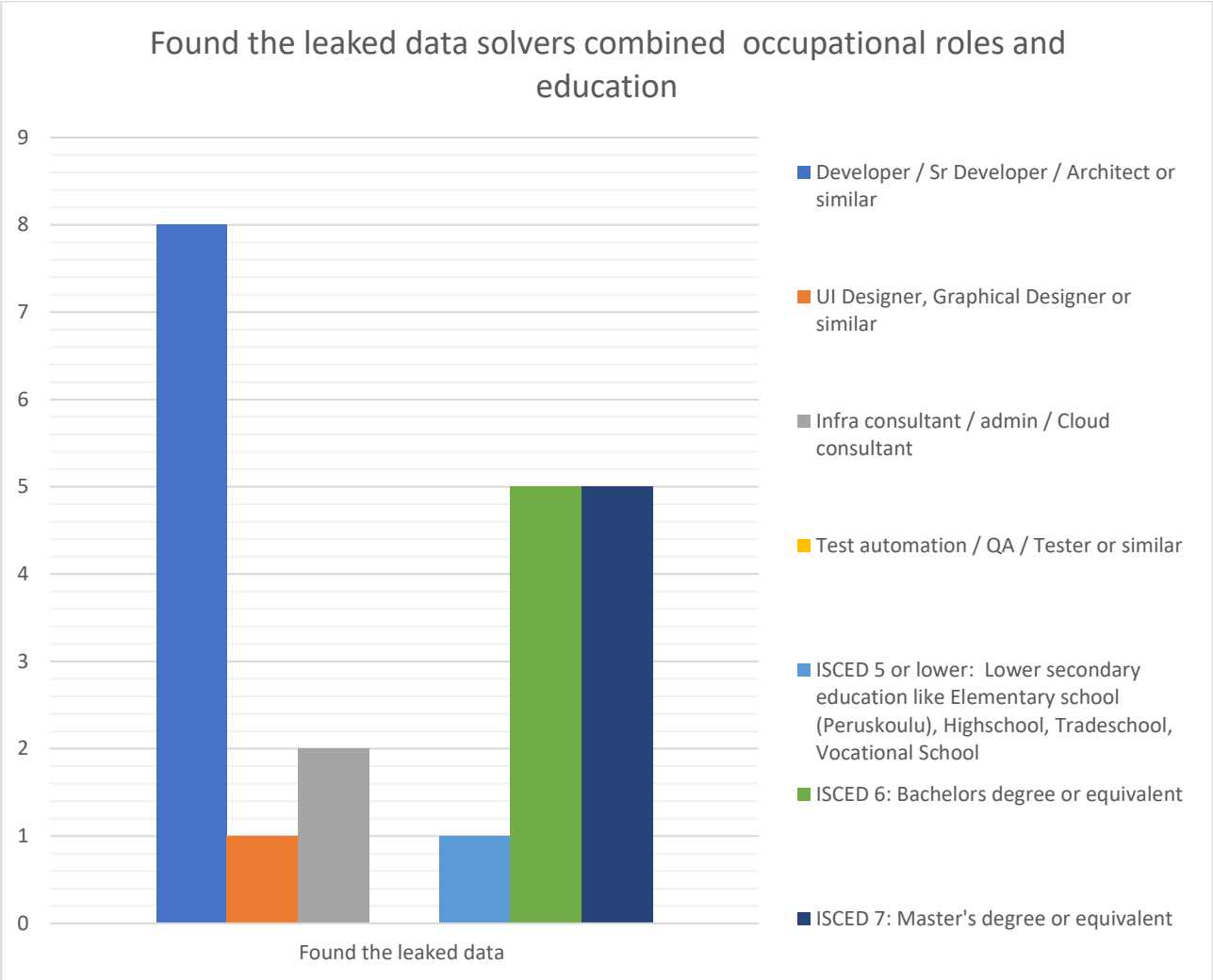


Figure 52. Found the leaked data solvers combined occupational roles and education

The **"Found the leaked data"** was the final puzzle in the challenge, but according to Figure 52 11 respondents have answered having solved that puzzle where only three had puzzle needed to solve this one as shown on Figure 45 or Figure 51. This answer option on the questionnaire has been most likely been mixed-up with some other puzzle, but it cannot be deduced which one. This data is there for invalid and offers no insight on the solvers of the last puzzle. This is the same number of respondents who reported having "finished the hunt". This would indicate that there

was some flaw in the challenge, and the final data flag could be retrieved without escaping the *rbash*, or respondents misunderstood the questions.

6 Conclusions

The number of responses was very low and the group of participants who answered was very homogenous. Only one participant reported their occupational role as non-technical. Similarly, only one participant had a degree below ISCED level 6. To summarize, no justified conclusions can be drawn from the answers. Employees being all from the same tech company can affect the variance negatively in the group. Can the results answer the research questions?

Can a preference of non-technical solutions to puzzles be observed in a multi-choice CTF with employees having lower educational level or non-technical occupational role?

The question cannot be answered with the data collected from the research. The similarity of education and occupational role of all respondents eliminates the existence of a control group. The number of respondents was too small and individual variation affected the results. Participants with both technical occupational role and high level of education preferred the path 1 with more non-technical solutions. On path 2 where the puzzles were purely technical, representatives from both ISCED level 5 educational level, which was the lowest amongst the respondents, and an UI-designer or designer, which was the only respondent with a non-technical occupational role, reported having solved puzzles. The respondent with the only non-technical occupational role did not finish the path 2, which would support the hypothesis, but with only one respondent, no conclusions can be drawn.

How do employees with lower educational level or non-technical occupational role progress compared to employees with higher education or more technical role in a gamified learning environment?

The question cannot be answered with the limited number of responses and homogenous nature of the respondent group made the comparison difficult. Participants with both technical occupational role and high level of education forfeited with the challenge in several levels, where participants with non-technical occupational role or lower level of education were in the group of participants filling out the questionnaire and processing far in the challenge. The one respondent with ISCED level 5 education progressed through the path 2 which was the more challenging and technical one. The respondent did not report having solved the last puzzle but reported having finished the challenge. The answers to the question for solving the last puzzle did not correlate with the answers to the question of having solved the whole challenge. Either one of those questions could have been misunderstood or the last puzzle had a solving option that was not intended and not provided as an answer option on the questionnaire.

Do participants with little to no prior experience in CTFs prefer recently taught techniques more than participants with experience?

This question cannot be answered with the limited data collected from the questionnaire. Only four respondents had previous CTF experience, all of them attended the training. In the group of respondents who had no previous CTF experience and finished the hunt, only two attended the training. Regardless of prior CTF experience, the techniques taught at the training were utilized by respondents. Path 1 had more techniques taught in the training than path 2 and was more popular. Most respondents answered having solved the first puzzle in path 2 which was taught at the training session, but not fewer respondents answered having solved the latter ones. The taught techniques appeared most popular.

Other aspect researched was that can gamification help teach technical concepts to people with no previous knowledge of the subject. Also, for this question, the answer remains unanswerable with the data from the research. Participants favouring the path1 with more solutions taught in detail in the training session, and not having prior CTF experience, could give slight indication that

recently taught topics were understood and used in practise, but the lack of non-technical participants does not indicate anything on the differences between technical and non-technical participants on the matter.

The questionnaire was clearly not properly formed, since there are some possibility of misunderstandings. Respondents have stated that they have finished the challenge and recovered the stolen data but have not completed the password cracking needed to login to the server. Most of the participants who started the challenge, but did not finish it, did not answer the questionnaire.

To get better results from similar research, the group of participants should be more carefully selected to include both technical and non-technical participants as well as participants of different educational levels. Avoid using a single group such as a company combined with no control over who participates to ensure variance in the focus group.

The number of participants should be ensured to provide enough data to minimize the effect of individual exceptions. Participants could be acquired via universities, social media or companies specialized in gathering participants for research. Motivating participants gathered from various sources could prove difficult without any incentive. Because the study is not limited to a questionnaire but has a laborious portion of solving the CTF challenge, participants with interest to the topic would most likely be overrepresented without a considerable incentive to attract other participants. Incentives could be monetary or student credits. Providing incentives for participating in studies requiring considerable amount of time from participants can attract participants who would not actually partake in the laborious portion of the study. Monitoring participation and progress would be crucial in keeping the results reliable.

The questionnaire should be constructed with better distinction on the focus of underlying education. In the current questionnaire, only the level of education is included and not if the education was technical, humane, artistic, applied sciences or formal sciences for example. A categorisation

based on faculties or similar existing categories should be implemented in the questionnaire. The questionnaire lacked options for some of the puzzle options. The encrypted zip-archive puzzle was not a selectable option in solved puzzles. The first level questions on path 1 had some overlap and one solving option was left out completely. As for the ELF-binary puzzle, only one of the solving options were added to the questionnaire by accident. The options were to run and monitor network traffic, debug, or de-compile the application. Only the de-compilation option was added to the questionnaire. The challenge got modifications later in the design process, which accidentally were not all reflected on the questionnaire.

The paths in the challenge should be more clearly divided into technical and non-technical approaches. The paths should still be swappable in several points in the challenge, so choosing one non-technical solution would not bind you to use non-technical solutions for the rest of the challenge. More non-technical solutions should be added. An additional Boolean yes or no checkbox for each solved puzzle that “Did you use a tool taught in the initial teaching session or its materials?”.

7 Discussions

The vulnerabilities and misconfigurations the puzzles were based on, as well as the techniques and tools used to exploit them, are commonly used by cyber security professionals and cyber criminals. Some tools, for example Steghide, are not as commonly used by professionals outside training, but can be used by cybercriminals for hiding messages and payloads in public web.

The puzzles and challenges were purposefully made obvious with hints and clues, which would not be so apparent in real environments. The purpose was to engage the participants into finding the next step of the game and to focus on solving the challenges instead of having to repeat the discovery and enumeration phase after each success, imitating how a real-life scenario would play out.

The vulnerabilities in these types of challenges are usually easily exploitable with the most common tool or public exploit. In hosted environments such as Hack The Box, in more demanding challenges, exploitation may require sophisticated exploit chaining or bypassing web-application-firewalls. In actual audits, the vulnerabilities are often there, but not always exploitable. There might be network settings, firewalls, or other mitigations for these vulnerabilities. It is still important to find them when conducting a penetration test or a security audit, because the environment might change later, providing a combinable vulnerability, or removing some existing mitigation.

One of the biggest hurdles with professionals breaking into cyber security with a background in participating in CTF competitions or platforms, is that knowing a target is vulnerable gives extra motivation to keep on trying different approaches and methods. In an actual penetration test assignment, there is usually no prior knowledge of the target's vulnerability. There can be a previous penetration test report in which previous vulnerabilities are disclosed, and testing their fixes is a good approach, but there is no certainty that the tester will be able to compromise the system. This uncertainty can lead to lack of motivation or over-motivation. Some testers get fixated on a vulnerability discovered by a scanner and the extent of the test can suffer, since most tests are performed in a defined timeframe. If too much time is spent on a single vulnerability, testing the probability of other exploits can be neglected. Lack of motivation can lead to loss of testing quality as well. If possible vulnerabilities are not investigated, the confidence of the find will be low. This is a topic that needs more attention when using CTF type challenges for educational purposes.

References

All About Hack The Box. Hack The Box. 2022. Accessed on 4 October 2022. Retrieved from https://www.hackthebox.eu/about-us

Baloch, R. 2015. Ethical hacking and penetration testing guide. Boca Raton, Florida: CRC Press.

Black Hat. Black Hat. Accessed on 19 November 2022. Retrieved from https://www.blackhat.com

Bradbury, D. 2019. Thousands of API and cryptographic keys leaking on GitHub every day. Naked Security. Accessed on 11 November 2022. Retrieved from https://nakedsecurity.sophos.com/2019/03/25/thousands-of-coders-are-leaving-their-crown-jewels-exposed-on-github/

Breuer, P., Bowen, J., Palomar, E., Liu - 刘志明, Z. 2017. On Obfuscating Compilation for Encrypted Computing.

Brown, E. 2016. Learning Javascript, 3rd Edition. O'Reilly Media Inc.

Buchanan, C. 2014. Kali Linux CTF Blueprints. Olton: Packt Publishing, Limited. Accessed on 11 November 2022. Retrieved from http://ebookcentral.proquest.com/lib/jypoly-ebooks/detail.action?docID=1688607.

Chacon, S., Straub, B. 2014. Pro Git. USA: Apress.

Clark, B. 2013. RTFM : Red Team field manual. United States: Createspace.

Cloud Object Storage – Amazon S3 – Amazon Web Services. Amazon Web Services, Inc. Accessed on 10 October 2022. Retrieved from https://aws.amazon.com/s3/

Cloud Products. Amazon Web Services, Inc. Accessed on 5 October 2022. Retrieved from https://aws.amazon.com/products/

Create a bot for your workspace. Slack Help Center. Accessed on 5 October 2022. Retrieved from https://slack.com/help/articles/115005265703-Create-a-bot-for-your-workspace

CTF 101. CTC 101. Accessed on 3 October 2022. Retrieved from https://ctf101.org/

CTF. Disobey. Accessed on 17 November 2022. Retrieved from https://disobey.fi/2023/ctf

Davis, A., Leek, T., Zhivich, M., Gwinnup, K., Leonard, W. 2014. The Fun and Future of CTF. San Diego, CA: USENIX Association. Accessed on 11 November 2022. Retrieved from https://www.usenix.org/conference/3gse14/summit-program/presentation/davis.

Definition of RABBIT HOLE. Merriam-Webster dictionary. Accessed on 7 November 2022. Retrieved from https://www.merriam-webster.com/dictionary/rabbit+hole

Definition of STEGANOGRAPHY. Merriam-Webster dictionary. Accessed on 7 November 2022. Retrieved from https://www.merriam-webster.com/dictionary/steganography

Disobey. 2020. Craving for that #disobey2021 ticket? We know you do. Disobey 2021 Hacker Challenge is produced in cooperation with @KouvostoTelecom . It seems they've made some #OPSEC fails down the road.. can you find what those fails are? 😉 The Hacker ticket sale starts NOW!. Twitter. Accessed on 7 November 2022. Retrieved from https://twitter.com/Disobey_fi/status/1316287633631440901

Dubey, S. 2020. An Introduction to Cybersecurity, Capture the Flag Contests, and Basic Security Concepts. Blogpost. Accessed on 11 May 2022. Retrieved from https://betterprogramming.pub/an-introduction-to-cybersecurity-capture-the-flag-contests-and-basic-security-concepts-80f3fbf62bbc

Eilam, E. 2005. Reversing : secrets of reverse engineering. Indianapolis, IN: Wiley.

Enabling interactions with bots. Slack API. Accessed on 5 October 2022. Retrieved from https://slack.com/bot-users

example_hashes [hashcat wiki].[TYPE OF PAGE ON NAME OF WEBSITE]. Accessed on 13 November 2022. Retrieved from https://hashcat.net/wiki/doku.php?id=example_hashes

Fuzzing. OWASP Foundation. Accessed on 10 October 2022. Retrieved from https://owasp.org/www-community/Fuzzing

g0tmi1k. About ~ VulnHub. About page. Accessed on 4 October 2022. Retrieved from https://www.vulnhub.com/about/

Gite, V. 2006. Understanding /etc/passwd File Format. nixCraft. Accessed on 8 November 2022. Retrieved from https://www.cyberciti.biz/faq/understanding-etcpasswd-file-format/

GitHub commands 27.44% market share in Software Configuration Management. Enlyft. Accessed on 7 November 2022. Retrieved from https://enlyft.com/tech/products/github

GitHub's products. GitHub Docs. Accessed on 7 November 2022. Retrieved from https://ghdocs-prod.azurewebsites.net/en/get-started/learning-about-github/githubs-products

Gofore Oyj. 2020. Our story: 'Go for e-' business. Gofore. Accessed on 16 November 2022. Retrieved from https://gofore.com/en/our-story-go-for-e-business/

Gofore Oyj. 2021. Gofore was awarded the Employer Brand of the year in Finland. Gofore. Accessed on 28 October 2022. Retrieved from https://gofore.com/en/gofore-was-awarded-the-employer-brand-of-the-year-in-finland/

Gofore Oyj. 2022. Gofore awarded the Engineer Employer of The Year in Finland. Gofore. Accessed on 28 October 2022. Retrieved from https://gofore.com/en/releases/gofore-awarded-the-engineer-employer-of-the-year-in-finland/

Gonzalez, H., Llamas-Contreras, R., Montaño, O. 2019. Using a CTF Tournament for Reinforcing Learned Skills in Cybersecurity Course.

Halunen, K. 2012. Hash function security:cryptanalysis of the Very Smooth Hash and multicollisions in generalised iterated hash functions. Oulun yliopisto. Accessed on 11 November 2022. Retrieved from http://urn.fi/urn:isbn:9789514299667.

Hamari, J. 2015. Gamification : motivations & effects. Helsinki: Aalto University.

Hirsjärvi, S., Remes, P., Sajavaara, P. 2004. Tutki ja kirjoita. Helsinki: Tammi.

HITCON CTF 2022. HITCON CTF 2022. Accessed on 19 November 2022. Retrieved from https://ctf2022.hitcon.org

How to Password Protect a Zip Folder. WinZip blog. 2022. Accessed on 8 November 2022. Retrieved from https://winzip.com/blog/enterprise/how-to-password-protect-a-zip-folder/

How to Play Challenges. Hack The Box Help. Accessed on 11 November 2022. Retrieved from https://help.hackthebox.com/en/articles/5185436-how-to-play-challenges

Huotari, K., Hamari, J. 2012. Defining Gamification - A Service Marketing Perspective. Accessed on 6 November 2022. Retrieved from https://www.researchgate.net/profile/Juho-Hamari/publication/259841647_Defining_Gamification_-_A_Service_Marketing_Perspective/links/0c96052e13e865be00000000/Defining-Gamification-A-Service-Marketing-Perspective.pdf

Identifying resources on the Web - HTTP | MDN. MDN Web Docs. Accessed on 7 November 2022. Retrieved from https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/Identifying_resources_on_the_Web

JavaScript Deobfuscator. deobduscate.io. Accessed on 7 November 2022. Retrieved from https://deobfuscate.io/

Khamis, H. 2021. Studies on Image Steganography. Itä-Suomen yliopisto. Accessed on 11 November 2022. Retrieved from http://urn.fi/urn:nbn:fi:uef-20210974.

Koek, W., Yu, V., Chen, V. 2022. Meaningful Gameplay Design and the Effect on Eudaimonic and Hedonic Gaming Experience. Accessed on 19 November 2022. Retrieved from http://hdl.handle.net/10125/79533

Logos. Press Center. Accessed on 11 November 2022. Retrieved from https://press.aboutamazon.com/logos

Lonvick, C., Ylonen, T. 2006. The Secure Shell (SSH) Protocol Architecture. Internet Engineering Task Force. Accessed on 8 November 2022. Retrieved from https://datatracker.ietf.org/doc/rfc4251

Manky, D. 2019. Malware Payloads Hide in Images: Steganography Gets a Reboot. threatpost. Accessed on 7 November 2022. Retrieved from https://threatpost.com/steganography-combat/143096/

Media Kit. Slack. Accessed on 11 November 2022. Retrieved from https://slack.com/media-kit

Mittal, S. 8 Best Java Decompilers in 2022 | Java Hungry. Blogpost. Accessed on 8 November 2022. Retrieved from https://javahungry.blogspot.com/2018/12/8-best-java-decompiler-in-2019.html

Ndichu, S., Kim, S., Ozawa, S. 2020. Deobfuscation, unpacking, and decoding of obfuscated malicious JavaScript for machine learning models detection performance improvement. Accessed on 11 November 2022. Retrieved from https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/trit.2020.0026.

Network Monitor — Firefox Source Docs documentation. MDN Web Docs. Accessed on 7 November 2022. Retrieved from https://firefox-source-docs.mozilla.org/devtools-user/network_monitor/

Nguyen, D. 2019. Serverless Architecture on AWS. Oulu University of Applied Sciences. Accessed on 11 November 2022. Retrieved from http://www.theseus.fi/handle/10024/227904.

Nosrati, M., Karimi, R., Hariri, M. 2011. An introduction to steganography methods.

Panta, D. 2015. Web crawling and scraping : developing a sale-based website. Turun ammattikorkeakoulu. Accessed on 11 November 2022. Retrieved from http://www.theseus.fi/handle/10024/93110.

Panwar, S., Kumar, M., Sharma, S. 2020. Digital Image Steganography Using Modified LSB and AES Cryptography. Cham: Springer International Publishing. Accessed on 7 November 2022. Retrieved from http://link.springer.com/10.1007/978-3-030-39875-0_39

Penetration Testing, Test the AWS environment against defined security standards. Amazon Web Services, Inc. 2022. Accessed on 25 February 2022. Retrieved from https://aws.amazon.com/security/penetration-testing/

Pinna, E., Cardaci, A. GTFOBins. GFOBins project web page. Accessed on 13 November 2022. Retrieved from https://gtfobins.github.io/

Postel, J., Reynolds, J. 1985. File Transfer Protocol. Internet Engineering Task Force. Accessed on 8 November 2022. Retrieved from https://datatracker.ietf.org/doc/rfc959

Red Team - Glossary | CSRC. CSRC Home Page. Accessed on 11 October 2022. Retrieved from https://csrc.nist.gov/glossary/term/red_team

Red Teaming Handbook, 3rd Edition. 2021. UK Ministry of Defence. Accessed on 14 November 2022. Retrieved from https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/1027158/20210625-Red_Teaming_Handbook.pdf

Resources and URIs - HTTP | MDN. MDN Web Docs. Accessed on 7 November 2022. Retrieved from https://developer.mozilla.org/en-US/docs/Web/HTTP/Resources_and_URIs

Secure and resizable cloud compute – Amazon EC2 – Amazon Web Services. Amazon Web Services, Inc. Accessed on 5 October 2022. Retrieved from https://aws.amazon.com/ec2/

Serverless Architectures. Amazon Web Services, Inc. Accessed on 30 September 2022. Retrieved from https://aws.amazon.com/lambda/serverless-architectures-learn-more/

SFTP protocol, clients, servers etc. Page by the original author of SFTP. SSH corporation homepage. Accessed on 11 November 2022. Retrieved from https://www.ssh.com/academy/ssh/sftp

sh3llm4g1ck. CTF Sites - Biggest Collection Of CTF Sites. GitHub Docs. Accessed on 4 October 2022. Retrieved from https://ctfsites.github.io/

shamane. 2021. Red, Blue, Purple, White, Black & Gold Team. Privasec Global. Accessed on 20 November 2022. Retrieved from https://privasec.com/blog/coloredteams/

Shannon, S. 2020. Gamification and work motivation. Accessed on 11 November 2022. Retrieved from https://janet.finna.fi/Record/theseus_jamk.10024_346837.

Subresource Integrity - Web security | MDN. MDN Web Docs. Accessed on 7 November 2022. Retrieved from https://developer.mozilla.org/en-US/docs/Web/Security/Subresource_Integrity

The National Cyber Security Centre (NCSC). 2022. Penetration Testing. The National Cyber Security Centre (NCSC). Accessed on 11 October 2022. Retrieved from https://www.ncsc.gov.uk/guidance/penetration-testing

The Structure of the European Education Systems 2018 / 19 : Schematic Diagrams. 2022. Eurostat. Accessed on 18 November 2022. Retrieved from https://eacea.ec.europa.eu/national-policies/eurydice/sites/default/files/the_structure_of_the_european_education_systems_2018_19.pdf

TryHackMe | Cyber Security Training. TryHackMe. Accessed on 4 October 2022. Retrieved from https://tryhackme.com

Turpin, K., Pietrowski, W., Spencer, C., McGary, C., Causey, B., Petit, L., Scovetta, M., Manico, J., Coleman, J., Agarwal, A., Petukhov, A. 2010. Secure Coding Practices - Quick Reference Guide. Accessed on 10 October 2022. Retrieved from https://owasp.org/www-pdf-archive/OWASP_SCP_Quick_Reference_Guide_v2.pdf

vulc@n. CTF History. DEF CON® Hacking Conference. Accessed on 9 May 2022. Retrieved from https://defcon.org/html/links/dc-ctf-history.html

What Does 'Pwn' Mean?. Merriam-Webster dictionary. Accessed on 11 November 2022. Retrieved from https://www.merriam-webster.com/words-at-play/pwn-what-it-means-and-how-you-say-it

What is a Brute Force Attack? | Definition, Types & How It Works. Fortinet. Accessed on 10 October 2022. Retrieved from https://www.fortinet.com/resources/cyberglossary/brute-force-attack

What is AWS. Amazon Web Services, Inc. Accessed on 4 October 2022. Retrieved from https://aws.amazon.com/what-is-aws/

What is SSH Public Key Authentication?. SSH corporation homepage. Accessed on 8 November 2022. Retrieved from https://www.ssh.com/academy/ssh/public-key-authentication

Wortley, F., Alison, F., Thompson, C. 2021. Log4Shell: RCE 0-day exploit found in log4j, a popular Java logging package | LunaTrace. Lunasec Blog. Accessed on 7 November 2022. Retrieved from https://www.lunasec.io/docs/blog/log4j-zero-day/

Appendices

Appendix 1. Questionnaire

Gofore Treasure Hunt CTF Questionnaire

Questions about the treasure hunt. What did you do and how did you feel? The questions will be handled anonymously and the data collected used for my thesis.

1. Did you finish the hunt?

- ◯ Yes
- ◯ No

2. Was the hunt

- ◯ Too hard
- ◯ Too easy
- ◯ Just right

3. What is your highest level of education? (https://eacea.ec.europa.eu/national-policies/eurydice/sites/default/files/the_structure_of_the_european_education_systems_2018_19.pdf)

- ◯ ISCED 5 or lower: Lower secondary education like Elementary school (Peruskoulu), Highschool, Tradeschool, Vocational School
- ◯ ISCED 6: Bachelors degree or equivalent
- ◯ ISCED 7: Master's degree or equivalent

- ◯ ISCED 8: Doctorate or equivalent

4. Is your position .. (Pick closest one that applies) or use other

- ◯ Developer / Sr Developer / Architect or similar
- ◯ Managerial / Accounting / Sales or similar non technical
- ◯ Test automation / QA / Tester or similar
- ◯ UI Designer, Graphical Designer or similar
- ◯ Service Architect
- ◯ Change leader / consultant
- ◯ Infosec / Cybersec consultant
- ◯ Infra consultant / admin / Cloud consultant
- ◯ Other

5. What puzzles did you solve?

- ☐ The login on the "Sepposorsa Webservice" by guessing or bruteforcing the password
- ☐ The login on the "Sepposorsa Webservice" by fuzzing and finding the password file
- ☐ Deobfuscated the javascript
- ☐ The image url leading to another server
- ☐ Anonymous login FTP service
- ☐ SSH Key in Github repository
- ☐ Cracked the SSH key password with John or similar software
- ☐ Found the SSH key password hidden in an image
- ☐ Reverse engineered the Java application

- [ ] Reverse engineered the ELF binary
- [ ] Found password hash file by fuzzing
- [ ] Cracked Overlord password hash
- [ ] Escaped rbash
- [ ] Found the leaked data
- [ ] Other

6. Did you attend the teaching session?

- ◯ Yes
- ◯ No

7. Previous experience in CTFs

- ◯ None
- ◯ Some
- ◯ Alot

8. Open feedback

Appendix 2. Questionnaire answers

	The login on the "Seppo-sorsa Web-service" by guessing or bruteforcing the password ;The login on the "Senno-	Some	None	The image url leading to another server ;Anonymous login FTP service ;SSH Key in Github repository ;Found	Some	None	Some	None	Some
	Yes	Deobfuscated the javascript ;The image url leading to another server ;Anonymous login	No	Deobfuscated the javascript ;SSH Key in Github repository ;Found the SSH key	Yes	Yes	Yes	Yes	Yes
	Developer / Sr Develop-	Developer / Sr Develop-	Developer / Sr Develop-	Developer / Sr Develop-	UI Designer, Graphical	Developer / Sr Develop-	Developer / Sr Develop-	Developer / Sr Develop-	Infrastructure consultant / admin /
	ISCED 5 or lower: Lower secondary education like Elementary school Basic	ISCED 6: Bachelors degree or equivalent	ISCED 6: Bachelors degree or equivalent	ISCED 7: Master's degree or equivalent	ISCED 7: Master's degree or equivalent	ISCED 7: Master's degree or equivalent	ISCED 6: Bachelors degree or equivalent	ISCED 7: Master's degree or equivalent	ISCED 6: Bachelors degree or equivalent
	Just right	Just right	Just right	Just right	Too easy	Just right	Just right	Just right	Just right
	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	9/16/21	9/16/21	9/16/21	9/16/21	9/16/21	9/16/21	9/16/21	9/16/21	9/17/21
1	9/16/21	9/16/21	9/16/21	9/16/21	9/16/21	9/16/21	9/16/21	9/16/21	9/17/21

None	None	None	None	None	None
No	No	No	Yes	Yes	Yes
Deobfuscated the javascript;SSH Key in Github repository;Found the SSH key	Deobfuscated the javascript;Anonymous login FTP service	The login on the "Seppo-sorsa Web-service" by guessing or bruteforcing the	The login on the "Seppo-sorsa Web-service" by guessing or bruteforcing the	Deobfuscated the javascript;The image url leading to another server;SSH Key in Github	
Developer / Sr Develop-	Infra consultant / admin /	Test automation / QA /	Developer / Sr Develop-	Developer / Sr Develop-	
ISCED 7: Master's degree or equivalent	ISCED 6: Bachelors degree or equivalent	ISCED 7: Master's degree or equivalent	ISCED 7: Master's degree or equivalent	ISCED 7: Master's degree or equivalent	
Just right	Just right	Just right	Just right	Just right	
Yes	Yes	No	No	No	
9/18/21	9/18/21	10/1/21	10/1/21	10/1/21	
9/18/21	9/16/21	10/1/21	10/1/21	10/1/21	
10	**11**	**12**	**13**	**14**	

Appendix 3. Questionaire answer summary

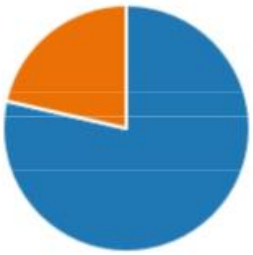
Forms(https://www.office.com/launch/forms?auth=2&from=FormsDomain) ? HH

Gofore Treasure Hunt CTF Questionnaire

14	199:55	Active
Responses	Average time to complete	Status

1. Did you finish the hunt?

Yes	11
No	3



2. Was the hunt

Too hard	0
Too easy	1
Just right	13

3. What is your highest level of education? (https://eacea.ec.europa.eu/national-policies/eurydice/sites/default/files/the_structure_of_the_european_education_systems_2018_19.pdf)

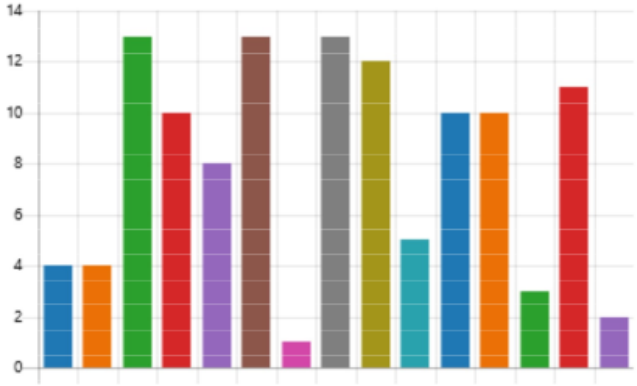
ISCED 5 or lower: Lower seco...	1
ISCED 6: Bachelors degree or ...	5
ISCED 7: Master's degree or e...	8
ISCED 8: Doctorate or equival...	0

4. Is your position .. (Pick closest one that applies) or use other

Developer / Sr Developer / Ar...	10
Managerial / Accounting / Sal...	0
Test automation / QA / Tester ...	1
UI Designer, Graphical Design...	1
Service Architect	0
Change leader / consultant	0
Infosec / Cybersec consultant	0
Infra consultant / admin / Clo...	2
Other	0

5. What puzzles did you solve?

The login on the "Sepposorsa ...	4
The login on the "Sepposorsa ...	4
Deobfuscated the javascript	13
The image url leading to anot...	10
Anonymous login FTP service	8
SSH Key in Github repository	13
Cracked the SSH key passwor...	1
Found the SSH key password ...	13
Reverse engineered the Java a...	12
Reverse engineered the ELF bi...	5
Found password hash file by f...	10
Cracked Overlord password h...	10
Escaped rbash	3
Found the leaked data	11
Other	2



6. Did you attend the teaching session?

Yes	8
No	4

7. Previous experience in CTFs

None	8
Some	4
Alot	0

8. Open feedback

9
Responses

Latest Responses

"Was fun exercise and will definitely see the playthrough"

"It was great! More of these please! I had too little available time this t...

3 respondents (**33**%) answered **tools** for this question.

Appendix 4. Source code of Slackbot lambda

```
import json
from urllib import parse as urlparse
from urllib import request
import boto3
from datetime import datetime

now = datetime.now()
dt_string = now.strftime("%d/%m/%Y %H:%M:%S")
print("now: " + dt_string)


def put_flag(user, flag):
    dynamodb = boto3.client('dynamodb')
    response = dynamodb.put_item(
        TableName='flag_returns',
        Item={
            'user': {'S': user},
            'flag': {'S': flag},
            'id': {'S': user + flag},
            'time': {'S': dt_string}
        }
    )
    return response


def lambda_handler(event, context):
    body = json.loads(event['body'])

    slackevent = body['event']

    options = {
        "Authorization": 'Bearer [redacted]'
    }

    message = slackevent['text']
    channel = slackevent['channel']
    user = slackevent['user']

    channels = {
        '4de4f6b8aa2497b51f7e331efb39fc3bdd106202cb99e228e829e07ddcb52d19':
            {
                "channel": "G01QCCH855W",
                "message": "You found out where the leak started! Keep on inves-
tigating!"
            },
        '129396e81354516c096913b50e3fa8cb1f0b554b9183516465819e8fd8cc9702':
            {
                "channel": "G01Q1SNVCNM",
                "message": "Those were some bad-ass credentials! Keep on inves-
tigating!"
            },
        'qfVCuyVF0ZjYv9HZZhfRWkonDqIUN5xZA5vTuXXOSOJaRD1YIc':
```

```
            {
                'channel': "G01PFRWSVD4",
                'message': "You found some bad project keeping. Keep going!"
            },
        'XC0PJThzBCQUgZwmZ8to1txCqY5bbz1rXy2m9j00YQAh8tk9sj':
            {
                'channel': "G01PGPBRF0W",
                'message': "So you got in? Noice! Keep going!"
            },
        'a0fb9bda78f139d5b65562110afafd0c57325a7794b5824bb697b6a8bb777944':
            {
                'channel': 'G01PPEL7B4N',
                'message': 'Who still uses ftp?'
            },
        '130a3aed935c442ef8c83af1dd806e9bac14cd09c499fadda100abacdfc25999':
            {
                'channel': 'G01Q2CMMX9S',
                'message': 'Sesam open!'
            },
        'e0ebdec34d6322c67ffb4a3c54cf56daa3fda152c905af33e740839999655cc2':
            {
                'channel': 'G01Q0MQ0WEQ',
                'message': 'Servers, servers servers ...'
            },
        'a5e8d9dce352299d4f93eb3038a226467c3458d4c6345f5072d722e14390a79f':
            {
                'channel': 'G01PPFBB2ES',
                'message': 'Weeeeeeee aaare the champpions,  MYYY FRIEE-
EEENDS!!!'
            }

    }

    if user == 'U01P1UX1N5D':
        return {
            "isBase64Encoded": False,
            "statusCode": 200,
            "body": "ok"
        }

    channelToJoin = None
    try:
        channelToJoin = channels.get(message)
        answer = channelToJoin['message']
        url = 'https://slack.com/api/conversations.invite'

        myobj = {
            'channel': channelToJoin.get('channel'),
            'users': user
        }

        data = urlparse.urlencode(myobj).encode()
        print(data)
        x = request.Request(url, data, options)
        resp = request.urlopen(x)
```

```
        print(resp.read())

    except:
        answer = "You said " + message + ". That is not a valid flag. Try hard-
er!"
        return {
            "isBase64Encoded": False,
            "statusCode": 200,
            "body": "ok"
        }

    url = 'https://slack.com/api/chat.postMessage'
    myobj = {
        'channel': channel,
        'text': answer
    }

    data = urlparse.urlencode(myobj).encode()
    x = request.Request(url, data, options)
    resp = request.urlopen(x)

    print(resp.read())

    put_flag(user, message)

    return {
        "isBase64Encoded": False,
        "statusCode": 200,
        "body": "ok"
    }
```

Appendix 5. Training presentation slide 2

GOFORE

Gofore systems were breached and personal information of at least Seppo Sorsa and Teppo Sorsa were stolen. The Information was shared in Ylilauta.

The leak has been removed from the site, but we need to find out what happened and what data was leaked. The hacker had slipped a note under the Gofore Tampere office door …