# An Efficient Strategy for Deploying Deception Technology

Noora Alhosani, Saed Alrabaee(✉), and Ahmed Al Faresi

Department of Information Systems and Security, CIT,
United Arab Emirates University, Al Ain, United Arab Emirates
{202170216,salrabaee,ahmed.alfaresi}@uaeu.ac.ae

**Abstract.** This article introduces a methodology for maximizing the effectiveness of deception technology in detecting sophisticated cyber attacks and overcoming the limitation of intrusion detection systems' ability. The proposed methods implement multi-layered deception techniques at different network, system, and application levels to enhance coverage and improve attack detection by using decoys that mimic real systems to attract and identify potential attackers. The method proposes dynamic adaptation to changes in the network environment and employs obfuscation to maintain the effectiveness of the proposed techniques. Implementing this method can provide organizations with an early warning system to respond quickly and mitigate potential damage from cyber attacks, and we shall prove that by performing multiple cyber attacks towards a network with an intrusion detection system and decoys, then compare the detection capability on both technologies.

**Keywords:** Deception · Deception attributes · Network Attacks · intrusion detection system

## 1 Introduction

Deception technology is one of the security solutions that aims to distract and mislead attackers from their actual targets, such as critical systems and data [1]. It does this by deploying decoys, traps, or honeypots that imitate real systems or services and lure attackers into engaging with them. The decoys are typically configured to appear as legitimate targets with specific operating systems, applications, network attributes, and vulnerabilities that the security admin of the network has chosen. They are placed within a network of real servers, workstations, and devices, to blend in and increase the chances of attracting attackers [2] and detect them.

Once an attacker engages with a decoy by scanning, probing, or attempting to exploit it, the deception technology collects valuable data and alerts security administrators. This data may include the attacker's IP address, location, tactics, and the timestamps and sequence of events leading up to the engagement [3]. Deception technology can display all the activities reported from decoys, giving the security administrator insight into all the engagements. This can assist


D. Perakovic and L. Knapcikova (Eds.): FABULOUS 2023, LNICST 542, pp. 177–194, 2024.
https://doi.org/10.1007/978-3-031-50051-0_13

the security administrator to correlate and identify anomalies that traditional security systems can miss. In addition, security administrators can gain insights into the attacker's motives, methods, and potential targets and take appropriate actions to prevent future attacks or mitigate and encounter the attack that has been detected.

Security administrators' overall mitigation actions may include blocking the attacker's IP address, quarantining the infected systems, and patching the vulnerabilities [4]. Details of mitigation actions do not fall within the research scope. Deception technology has several advantages over traditional security mechanisms, such as intrusion detection systems (IDS), firewalls, or antivirus solutions. It is more proactive, flexible, and adaptable and can detect unknown or zero-day attacks that evade traditional defenses. Moreover, it provides a safe environment for researchers and penetration testers to simulate real-world attacks and enhance an organization's security posture.

Achieving efficient and adaptable cyber deception involves ongoing network monitoring to observe adversary activities, replace decoys based on strategic planning, and have feasible implementation without disrupting the integrity of existing system [5]. The primary goal is establishing an effective strategy based on best practices for maintaining deception technology. Due to the growing complexity of cyber attacks and their ability to evade numerous security measures, utilizing this technology can offer improved detection capabilities and align with a defense-in-depth approach, ensuring multiple layers of protection for information systems. This article aims to answer the following questions:

- What is an efficient method to define decoy attributes?
- What is the significance of network mapping in the deployment phase of decoys?
- How can we detect false negatives in logs?

Our proposed approach will help define realistic decoy attributes and fingerprints and identify the most suitable locations to deploy the decoys within the network for improved visibility and enhanced detection capabilities. Additionally, we will conduct lab demonstrations to simulate various types of attacks and demonstrate how deception technology can successfully detect them compared to intrusion detection system deception capability.

## 2 Background

### 2.1 Problem Statement

While the numbers and the sophistication of cyber attacks is increasing, improving detection and prevention mechanisms has become crucial to ensure higher security. Even with multiple monitoring devices and security controls such as web application firewalls, intrusion detection and prevention systems, and SIEM, sophisticated attacks can bypass these controls and result in significant losses.

To address this problem, researchers over time has developed deceptive methods including honey pots and deception technology, which adds an extra layer of

defense to the network. honey pot and Deception technology is more of a strategy that needs to be tested, verified, and periodically changed to avoid exposure. It is crucial to establish clear set of objectives and prepare the prerequisites to ensure successful deployment.

### 2.2 Deception

Deception technology is a relatively new approach to cyber security that has gained much attention in recent years [6–8]. It is based on using deception to detect and prevent cyber attacks. The approach involves creating a controlled and monitored environment with decoys, traps, and honeypots that mimic real network assets and systems. Attackers are lured into engaging with these decoys, and their activities are monitored to gather intelligence on their tactics and techniques, and to detect malicious activity. One of the key benefits of deception technology is that it can provide an early warning system for detecting cyber attacks, allowing organizations to respond quickly and mitigate damage [9]. By creating a false sense of security for attackers, deception technology can also help gather intelligence on their tactics, techniques, and procedures, improving overall security posture. Additionally, deception technology can augment traditional security measures, providing an additional layer of defense against advanced threats that may bypass other security controls [9]. Deception technology is a comprehensive solution that has overcome some limitations of the traditional honey pot and honeynet technologies. It can create, customize, and manage decoys around the network to enhance the monitoring and detection of suspicious activities. The technology requires continuous network monitoring to observe adversary activities, optimal planning for feasible implementation, and safe deployment without breaking the system's integrity [5].

### 2.3 Honeynet

The project honeynet.org is a major source of information regarding honeynets [10]. The project's participants conduct research and run honeynets themselves, which serve as benchmarks for constructing and assessing honey nets [11]. They have also developed Open-Source honey net tools known as Honeywell [12]. Security enterprises, such as firewall/IDS manufacturers and antivirus manufacturers, maintain many honeypots in a honey net. These honeypots are distributed worldwide and provide threat information and an attack map. Organizations also use honeypots to hide their critical infrastructure and improve their security efforts with the obtained information. Honey nets are a type of honeypot, a security resource used to detect probes, scans, or attacks. Honey nets are more complex than honeypots and consist of multiple honeypots networked together to mimic a larger network. Deception technology is similar to honeypots but has several key differences.

### 2.4 Honeypot

Honeypots offer various capabilities and can mimic the operations of real systems, making them appear to be a part of the network. However, honeypots are actually isolated and closely supervised, allowing them to record the activities of attackers and gather information about their tools and operating procedures. Initially, honeypots were used as a trap method that is created and set up to be hacked to waste the attacker's resources and time on honeypots rather than attack real systems that actually exist [9]. Honeypots are classified based on the level of interaction they offer to the attacker, which results in low-, medium-, and high-interaction honeypots. The first two categories offer different levels of protocol emulation, while the high-interaction describes real-world systems. High-interaction honeypots are more expensive to maintain and are used significantly less than low/medium-interaction honeypots [13].

On a summary, honey nets, honeypots, and Deception share some similarities, there are also significant differences between them that are important to understand. Please refer to A comparison between Honey net, Honeypot, and Deception for comparing the main features of honey nets, honeypots, and deception technology (Table 1).

**Table 1.** A comparison between Honey net, Honeypot, and Deception

| Feature | Honeynet | Honeypot | Deception |
|---|---|---|---|
| **Scalability and administration** | Honeynets utilize virtualization to create entire network topologies using pre-set or custom images on a single physical or virtual machine | Setting up a honeypot involves a separate physical or virtual system placed within the network, and administrators must access each honeypot individually for modifications | Deception technology solutions are prepackaged and scalable, allowing for rapid decoy deployment and central management without direct access to the decoy itself |
| **Level of interaction** | Decoys imitate active hosts, responding to basic network commands while sharing the same operating system and MAC address, but may not react to all packet types | - Low-interaction honeypots simulate specific services, such as FTP or web servers, with attacker interaction determining the emulation depth. - High-interaction honeypots offer real operating systems and services like FTP or web servers instead of just emulating them | High-interaction decoys on real operating systems gather extensive attacker information, enabling automated incident response, accurate alarms, and forensic reports |
| **Risk of takeover/ compromise** | A compromised honeypot in a honeynet can become an attack source | It can be compromised | Modern deception platforms deploy out-of-band decoys to prevent attackers from launching network attacks |
| **Logging capability and monitoring** | They can be installed on the host system. However, there is no centralized dashboard to view logs and interactions | Honeypots don't have their own logs; they rely on capturing logs from the host OS. Logs can be forwarded to monitoring solutions or manually analyzed. There is no centralized dashboard to view honeypot logs and interactions | logs can be sent to the security monitoring tools like SIEM or viewed on the deception console's dashboard, without separate configurations for log forwarding or collection |

## 3 Methodology

The research methodology will have three steps. First will explore and collect honeypot and decoy attributes/fingerprinting methods from different resources and summarize them in order to to apply test their efficiency using an open source deception technology. Second, the study will also explore effective network mapping techniques to locate existing network assets/components for optimal decoy placement. Third and last, the research testing shall conclude our methodology by performing multiple cyber attack scenarios, on a virtual lab, the lab will have an intrusion detection system, and we shall implement decoy attributes that we collected previously, then test the ability of deception technology to detect and alert about cyber-attack attempts compared to the installed intrusion detection system (Table 2) (Fig. 1).

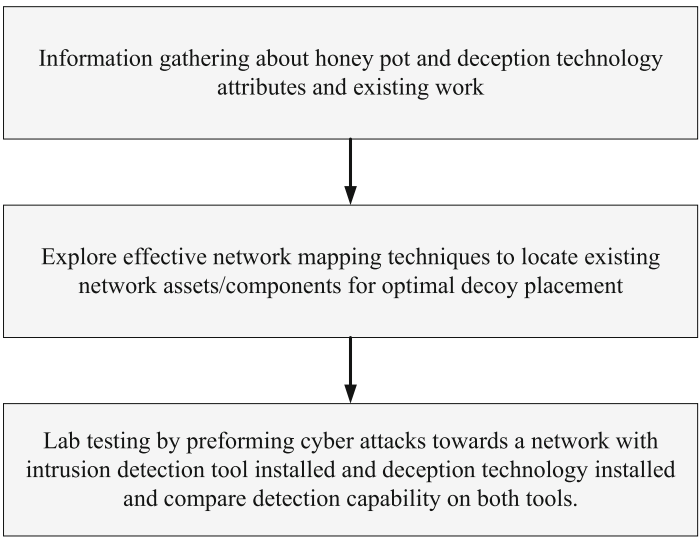


**Fig. 1.** Research methodology

### 3.1 Testing Lab Components

Our virtual testing lab consists of the following tools and components:

Each attack scenario will necessitate a unique network design and components tailored to the specific use case under examination and the type of cyber attack to be conducted. The following details the primary components that will be involved during the testing phase:

**Central Management Console.** The central management console provides a unified interface for configuring, managing, and monitoring the Dejavu system.

**Table 2.** A comparison between Honey net, Honeypot, and Deception

| Tool name | Tool description | Usage for experiment |
|---|---|---|
| Oracle Virtual box | Oracle VM VirtualBox is a versatile virtualization software that operates across various platforms. It provides users the capability to expand their current computer system to simultaneously run multiple operating systems, such as Microsoft Windows, Mac OS X, Linux, and Oracle Solaris | Hosting on virtual machines |
| Snort (NIDS) | Snort is an open-source Network Intrusion Detection System that employs a set of rules to identify harmful network activities. It scans network packets to detect matches with these rules and subsequently triggers alerts for users | Testing detection capability |
| Kali Linux virtual machine | Kali Linux is a Linux distribution, based on Debian, specifically crafted for conducting digital forensics and penetration testing tasks | Performing cyber attacks and scanning activites |
| nmap | "Network Mapper," commonly known as Nmap, is an open-source tool that provides network discovery and security auditing capabilities for free | Network scanning |
| Docker | Docker, as incorporated within the open-source Dejavu platform, is tasked with generating new decoys based on our designated attributes | Decoy creation |
| responder | Responder, a built-in tool in Kali Linux, caters to Link-Local Multicast Name Resolution (LLMNR) and NetBIOS Name Service (NBT-NS). It generates responses to specific NetBIOS queries according to the file server request | Perform LLMNR attack |
| Dejavu | Dejavu is an open-source deception platform capable of deploying decoys on the cloud (currently supporting AWS) and internal networks. It can simulate various operating systems and network services | management console and decoy control |

**Decoy.** Decoys are fake assets (e.g., servers, workstations, databases, etc.) that are strategically placed throughout the network to mimic the real assets.
**Attacker Machine.** This machine is based on kali linux image hosted on virtual box and shall be used to perform network scanning towards the decoys and other virtual machines on the lab.

The proposed framework aims to maximize the efficiency of deception technology in detecting sophisticated attacks by focusing on the following key aspects:

- Multi-layered Deception: The framework employs deception techniques at multiple layers of the network, system, and application to provide comprehensive coverage and increase the likelihood of detecting attacks.
- Strategic Decoy Placement: Decoys and honeypots are strategically positioned throughout the network to maximize their visibility to potential attackers and minimize the likelihood of accidental interaction with legitimate users.
- Dynamic Adaptation: The framework continuously adapts to changes in the network environment, modifying decoys and honeypots as necessary to maintain their effectiveness.
- Confidentiality: Ensuring the confidentiality of the deception process is crucial to prevent attackers from identifying and avoiding the decoys and honeypots. This can be achieved through obfuscation, encryption, and access control.

### 3.2 Phases

The virtual lab will simulate ten cyber-attacks, covering various decoy attributes and locations in the network to demonstrate the need for specific settings to enhance cyber-attack detection. The research will include the following phases:

- Phase one
  - Virtual lab setup and tool installation: We first choose Virtual Box to host kali linux and dejavu deception open source platform along with snort intrusion detection system which shall be inline mode to inspect all traffic on our internal network used for the lab.
- Phase two
  - Configure dejavu and deploy decoys: To configure DejaVu, we should install the software on a virtual machine. Then, we can follow the instructions provided by dejavu authors to set up the software and configure it to our network's requirements. Once DejaVu is configured, We shall determine the best deployment method based on our network's infrastructure and the type of decoys we want to use for testing.
  - Decoy customization and finger printing: Customization involves creating realistic decoy attributes such as open ports, services, and operating systems, and defining their behavior to appear as if they are legitimate.
- Phase three
  - Test cyber attack scenarios on decoys and perform result analysis: To test the effectiveness of the deployed decoys, various attack scenarios need to be simulated on them. During the simulation, we will compare the detection capability of deception technology and intrusion detection system and list the observations.

### 3.3 Steps

The following are the steps involved in deploying deception technology.

- Define objectives: Before implementing deception technology, map the network to help defining your objectives and desired outcomes.
- Identify the target areas: after network mapping, The next step is to identify the high-value areas in your network that need to be protected. These could include critical applications, databases, and other sensitive resources.
- Deployment: Deploy the decoys and honeypots in the identified areas of your network. Ensure they are configured correctly and blend in with the rest of your network to avoid detection.
- Monitor and manage: This includes collecting and analyzing data from the decoys, updating and refreshing them regularly, and integrating them with other security solutions if available.
- Incident response: In a breach or attack, deception technology can provide valuable insights into the attacker's tactics, techniques, and procedures. Ensure an incident response plan is in place to respond to such incidents effectively.

– Continuous improvement: Deception technology is not a one-time solution but requires continuous improvement and refinement. Regularly review and update your objectives, decoys to ensure that they remain effective.

### 3.4 Proposed Framework

Our proposed framework is based on covering multiple layers of attributes, including the network, system, and application, as well as the strategic positioning of the decoys. Most importantly, the entire process must maintain confidentiality:

#### Maintain Confidentiality

Security professionals should maintain the confidentiality of deception platforms, even from an organization's employees, such as IT staff. They ought to evaluate different solutions available in the market and may consult Gartner's [14] annual rankings for top choices. Furthermore, if a deception platform is acquired, its implementation shouldn't be broadly publicized or shared with IT companies. The involvement should be restricted, involving only government-approved companies with local offices who sign NDAs (Non-disclosure agreements) and have limited knowledge of the project and deployment.

### 3.5 Consider Data Ex-Filtration Detection Using Deception

During data ex-filtration, an attacker, whether an internal employee or an external threat, may gain unauthorized access to data and leak it externally. Determining the type of information leaked or the specific files targeted can be challenging. Placing honey files among sensitive files makes it possible to identify the source IP address when the file is opened in case of a leak, even if it is accessed outside the organization's network. This strategy enhances the detection and tracking of unauthorized data access and ex-filtration (Fig. 2).

#### Perform Network Mapping

To effectively integrate a deception platform into the network, comprehensive network mapping is crucial, covering all entry points like public websites and services across zones such as DMZ, VPN, webmail, and mobile apps. Decoys need to be tactically positioned in suitable quantities, offering diverse enticing services. Each decoy generates alerts, either false or true positives, which can be tracked and examined. Excessive decoys may lead to confusing and inundating log information [15]. Seek advice from the organization's network engineer or IT architect for a network diagram to aid in planning. Nonetheless, it's advisable to conduct independent mapping to confirm the accuracy of the provided network diagram. Take into account the following factors to maximize the benefits of this activity:

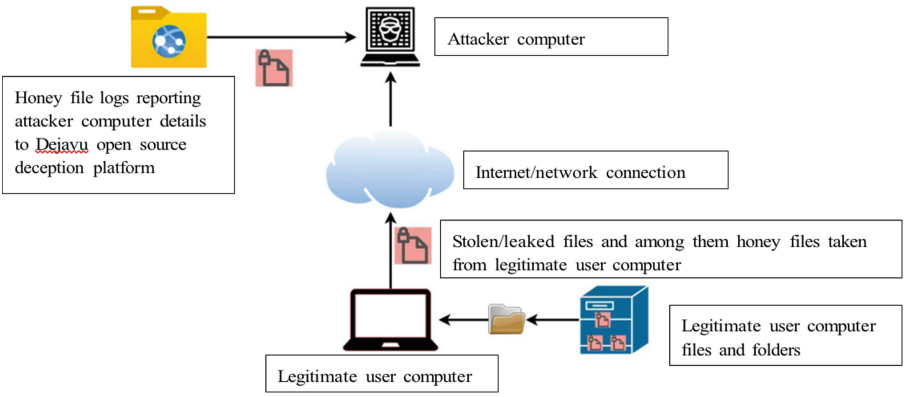


**Fig. 2.** Data Ex-filtration detection method

- Develop and uphold a list of essential active application servers: Organizations rely on particular services for maintaining their operations; for example, banks depend on payment systems, educational institutions on student and course registration systems, and healthcare facilities on patient databases. Each vital application comprises a list of servers and databases, with assigned IP addresses for incorporation into the information system/network. The list should connect the server to its respective IP address and outline the service it operates. It is best to designate an owner responsible for the routine upkeep of the server and application.
- To map the network, we can set up a host with full access from the organization's firewall, disable its internet connection, and scan the network using tools like nmap or masscan. These tools identify open ports in the IT system, which are then assessed for vulnerabilities. Nmap offers a wide assessment range, while Masscan focuses on quick, broad scans. For full network mapping with Masscan, use this command: masscan $0.0.0.0/0 - p0 - 65535$.

After obtaining the output, it will include all the hosts that responded to our host, allowing us to determine the number of subnets and live hosts, and gain further insight into the internal network.

### Customization of Decoy Attributes

Deploying a decoy aims to make it appear realistic to mislead attackers into wasting their resources trying to compromise it. If a network contains HP devices, the decoy should mimic these with similar specifications, such as MAC addresses, default credentials, and hostnames. A Windows machine decoy should join the domain with a matching hostname. Periodically changing the decoy's IP address, vulnerabilities, and hosted services can increase its effectiveness. The CONCEAL research emphasizes anonymization through frequent decoy IP, attributes, and

fingerprinting changes. However, our approach focuses on maintaining a consistent and authentic network appearance. We incorporate the fingerprinting principle and IP changes but reject the concept of frequent changes. Our proposal advocates for decoys to blend seamlessly into the network with accurate attributes and carefully tuned fingerprints. Manageable IP changes are preferable to maintain network integrity and minimize confusion.

**Define Network Attributes**

We consider the following attributes:

- TTL (Time to Live): TTL values vary for each operating system. Adjusting the TTL value to correspond with the decoy's specified OS type is crucial. For example, the TTL value for Windows OS is 128, whereas, for Linux, it is 64. When creating a decoy with a Windows operating system assigned, ensure the TTL value is modified to 128.
- MAC Address: When creating a decoy that mimics Fujitsu devices, it is essential to use Fujitsu-specific MAC addresses with their unique starting values (MAC prefix). According to CONCEAL, this principle applies to any device and operating system type; the MAC address prefix should correspond with the chosen operating system type. Simply copy and paste the appropriate prefix and complete the rest with random values.
- IP Address/Network VLAN: When deploying a decoy in user subnets, consider using a decoy with the same operating system as the users to make it blend in better. If you deploy a decoy in a server subnet, avoid using Windows 10 or Windows 7 (workstation operating systems), as placing a user workstation in a server subnet would be illogical. Such a strategy might raise suspicions among attackers.

After considering all these attributes, the likelihood of an attacker discovering the decoy on the network level is significantly reduced.

**Define System Attributes**

We consider the following attributes:

- OS: Select an operating system that is commonly used in your environment. If you primarily have Windows 10 machines, deploy a Windows 10 decoy rather than a Fujitsu decoy that stands out. If you have more Windows servers, use a Windows server decoy instead of RHEL. However, you can vary the decoy types if your environment includes a mix of operating systems. The main goal is to ensure the decoy blends in and doesn't stand out or appear unique among the other devices, as this could raise suspicion.
- Services: When deploying a decoy on a server subnet/VLAN, enable server-related services such as (but not limited to) FTP, SSH, and IIS. These services should also match the operating system type, as IIS service would not be

found on a RHEL server but on Windows-based hosts. Additionally, network administrators typically do not have user workstations with Apache or Tomcat within the regular users' VLAN, as they often separate the developers' environment.

**Define Application Attributes**

We consider the following attributes:

- Application protocols: When using deception technology, creating a fake webmail page with login capability is possible. However, such a decoy running on the intranet may be useless. It should be strategically placed on a target network, like the internet, to determine who is attempting to breach our network. Legitimate users are unlikely to use the fake page, as they already know the organization's main page.
- If we have a SWIFT network and decide to deploy a decoy within it, it's essential to ensure that all machines in this network are related to SWIFT and no other services. By doing so, an attacker roaming within this network will have difficulty distinguishing between legitimate devices and decoys.
- Consider enabling services such as Apache or Tomcat on application decoys, as application servers commonly use these services.

### 3.6 Network Scanning

When services are published online, attackers may compromise credentials and gain account access. Since these services generate large amounts of logs and users may lock their accounts accidentally, detecting attacks becomes challenging. To mitigate this, decoys can be used. These decoys, unknown to legitimate users, help to detect unauthorized attempts faster, as any login attempt indicates a possible security breach. Upon login attempts, users are looped back to the same page repeatedly. Meanwhile, the decoy reports to the Dejavu open source deception platform, triggering attack alerts.

Imagine an unknown, potentially compromised account within an organization in a realistic attack scenario. By placing a decoy of a webmail service, any login attempt becomes noticeable. Since internal employees are uninformed about this decoy webmail service, their chances of accessing or even seeking it are low because A) they already know how to access the organization's actual mail service and B) the organization may not even have a directly internet-accessible webmail service. Organizations with robust security controls usually avoid publishing their webmail services directly online, offering alternative access methods, thereby reducing their attack surface and limiting exposure. Entities considering deception implementation might opt to publish a decoy webmail service online without making any announcements. They can then monitor the type of traffic it attracts, which may alert them to an attacker's information-gathering activities. So considering one or both of the conditions, there is a high chance that an attacker is behind our network. To overcome the attack scenario, We shall

place decoys on the internet replicating our online services, where it simulates a login page for employees. In the below section, we shall simulate the attacks on different services and how to detect them using deception and, test the ability of the intrusion detection system to detect the attacks, then compare the results (Fig. 3).

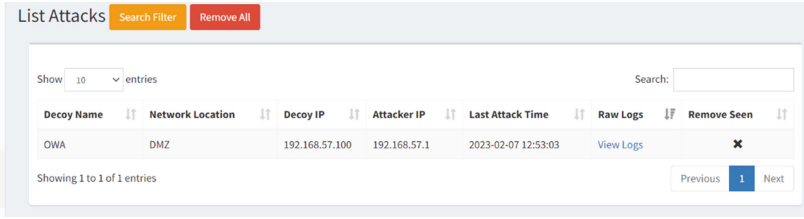


**Fig. 3.** Web mail decoy Log summary shown in dajavu open source deception platform console

## 4 Lab Testing Use Cases

Commonly, cyber attack phases include information gathering (reconnaissance), enumeration, and compromise of the system. We shall include recon and enumeration. The compromise phase is out of our scope. The test cases will be conducted via various cyber attack scenarios. The below table shows the tested attacks and their phases according to common cyber attack phases (Table 3):

**Table 3.** Test Cases

| Use case | Attacker target | Security control |
|---|---|---|
| Scanning the network using nmap (two types of network scan: normal nmap scan and silent scan) | Reconnaissance | Decoy network positioning |
| Password spray was not identified by Snort IDS but got detected by Dejavu open-source deception platform | Enumeration | Decoy customization |
| LLMNR poisoning was not detected on Snort IDS but got detected in Dejavu open-source deception platform | Enumeration | Decoy customization and network positioning |
| Simulate the importance of decoy attributes | Reconnaissance | Decoy customization |

### 4.1 Scenario 1: Scanning the Network Using Nmap (two Types of Network Scan: Normal Nmap Scan and Silent Scan)

**Normal Nmap Scan.** Port scanning is a method used to identify open network ports that may be sending or receiving data. By sending packets to specific ports on a host and analyzing the responses, vulnerabilities can be detected. Before port scanning can take place, it is necessary to first identify and map active hosts to their corresponding IP addresses. This process, called host discovery, starts with a network scan (Fig. 4).

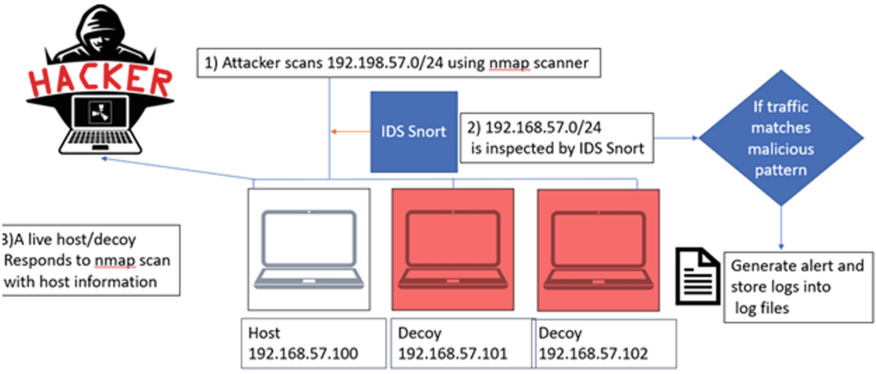


**Fig. 4.** Nmap Network scan

Port and network scanning is designed to identify the structure of IP addresses, hosts, and ports, enabling the detection of open or vulnerable server locations and assessing security levels. Network and port scanning can also uncover the existence of security measures to attackers. To identify port scanning activity, we can deploy a decoy on the internet that imitates a service (such as webmail) or a VPN service and then monitor the incoming traffic. Using kali linux machine we launched a scan towards the web mail service (Fig. 5).

On the other side, we open the Snort IDS log file and see the following logs that has been generated from the monitored network 192.168.57.0/24 and we can see that the scan has triggered alerts on NIDS side.

Observation: snort detected normal scan attempts on the subnet using Nmap, which has been presented in a series of logs. This was performed to confirm Snort is operating and inspecting the traffic (Fig. 6).

**Silent nmap Scan.** We prepared the following setup and used Nmap scanning tool. We added -ss to nmap command to enable the silent scan option and specified port numbers 80, 21, 443,22 for the scan. Nmap will scan 192.168.57.0/24 silently and search for the specified ports (Figs. 7 and 8).

Description: The log file is showing ipv6 traffic, which was generated from scenario 1.1 and not from the current silent scan. The screenshot shows that no

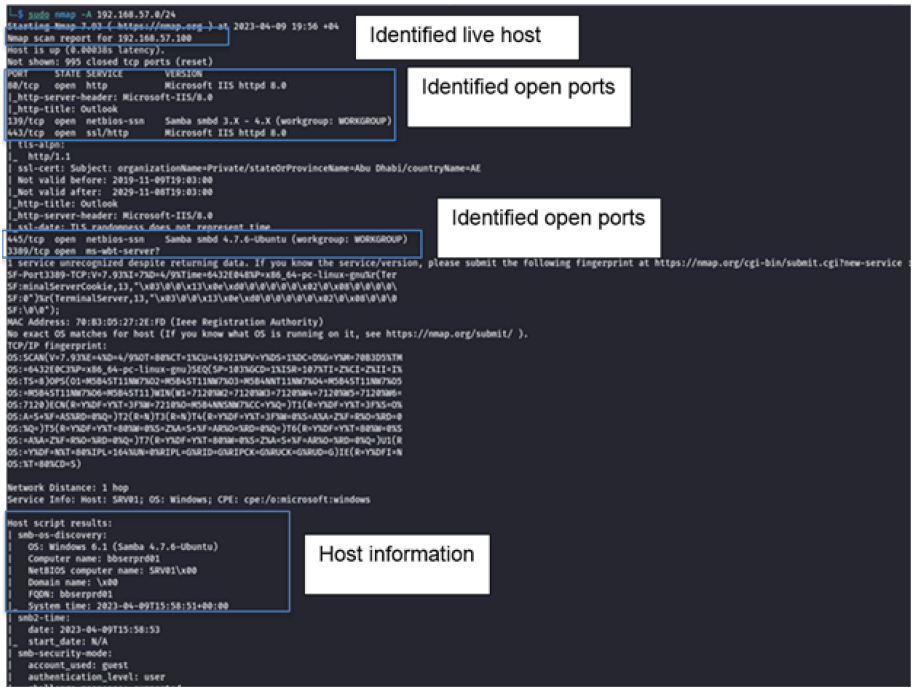


**Fig. 5.** Scan result from attacker side showing host information that has responded to nmap scan.

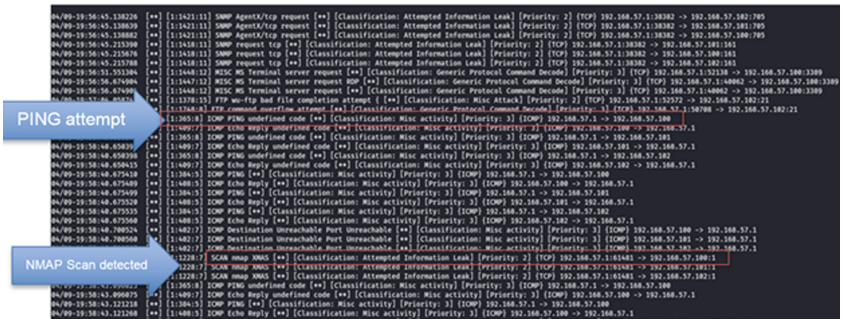


**Fig. 6.** scan result from IDS Snort side/terminal showing ping attempts and nmap scan traffic pattern which indicated as malicious

alerts have been triggered for this host, indicating that this is a false negative. On the other side, we can view the management console of deception technology to check if there were any alerts:

Description: We can see on the decoy IP, which is 192.168.57.101, has captured the scan attempt towards it, and it is showing the port numbers (on the

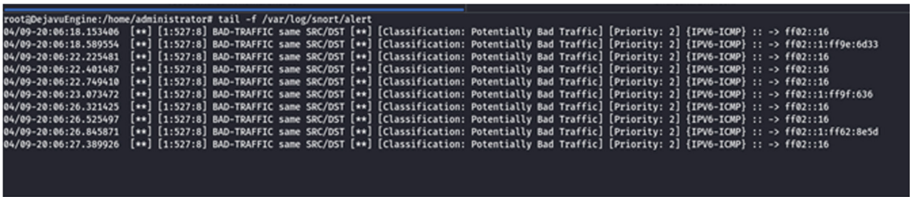


**Fig. 7.** Intrusion detection unable to detect silent scan

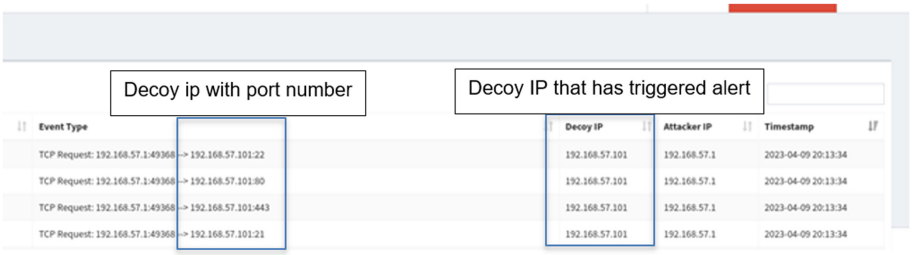


**Fig. 8.** silent scan detected on deception technology

event type) port 22, port 80, port 443, and 21, which we have specified on the silent scan option (refer to Fig. 20: scan result from attacker side/terminal showing the nmap command along with the subnet, specified port numbers and silent option (-ss)).

The console shows that alerts have been flagged for the targeted host of the scan (Decoy). The alerts have been generated and reported from the decoy to the management console.

### 4.2 Scenario 2: Password Spray Was Not Identified by Snort IDS but Got Detected by Dejavu Open-Source Deception Platform

67assword spraying is a type of brute force attack in which an attacker attempts to gain access to multiple accounts on an application using a list of usernames and default passwords. To avoid account lockouts that typically result from repeatedly trying multiple passwords on a single account, an attacker uses a single password (e.g., Secure@123) against numerous accounts on the application. This attack is common when an application or administrator sets a default password for new users.

**Creating a Decoy Without Setting Customized Attributes (Default):** Testing the case with no assigned attributes and relying on the default configuration of DejaVu as shown in Fig. 9.

**Credential Stuffing/Password Spray Not Identified by Snort IDS but Got Detected by Dejavu Open Source Deception Platform.** In this scenario, we have placed multiple decoys, and each runs a different service with

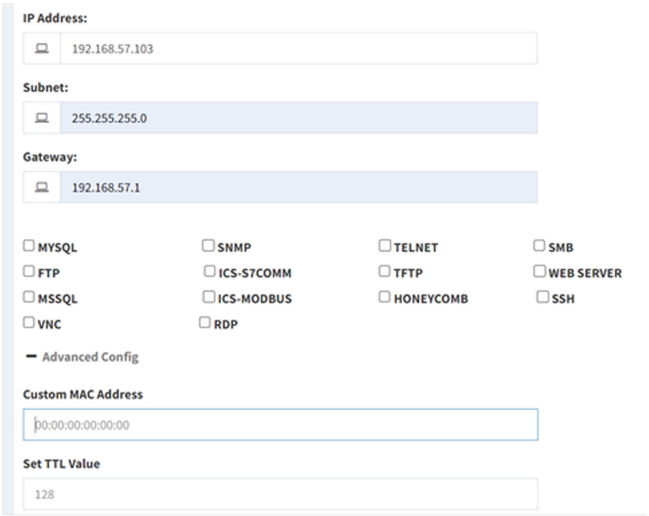


**Fig. 9.** Creating a decoy

login capability to test password spray detection. the created services (decoys) are below:

- Web mail login page decoy
- tomcat login page as a decoy
- Basic authentication application decoy

After attempting to login to the decoy services multiple times from the browser, we checked the logs on both IDS and Deception management console and identified that the attempt to login to the decoy from the browser had been alerted on deception technology and was not identified not alerted by IDS (Figs. 10, 11 and 12).

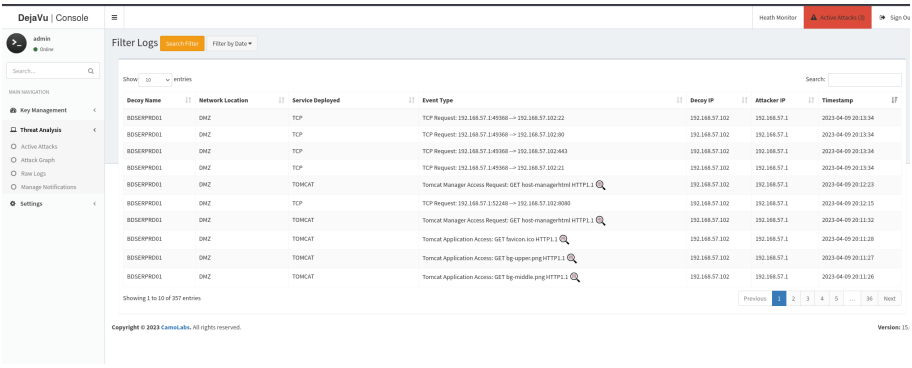


**Fig. 10.** Attack attempt detected on deception technology: example 1

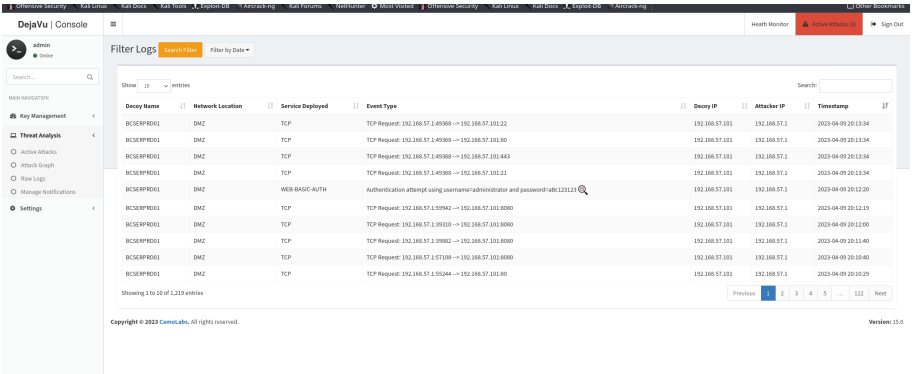


**Fig. 11.** Attack attempt detected on deception technology: example 2

On the other hand, we have the log file of the intrusion detection system that has not shown any attempt or alert:

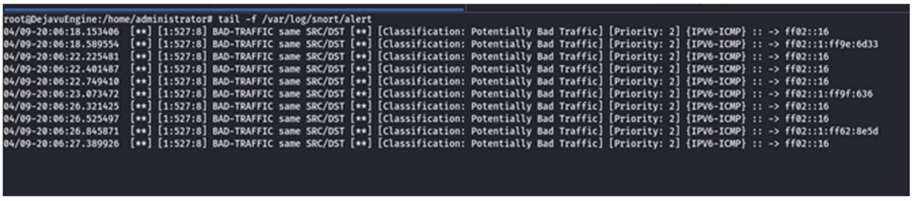


**Fig. 12.** Intrusion detection unable to detect password spray on the network

## 5 Conclusion

Overall, Deception technology is a comprehensive solution that has overcome some limitations of the traditional honey pot and honeynet technologies. It can create, customize, and manage decoys around the network to enhance the monitoring and detection of suspicious activities. The technology requires continuous network monitoring to observe adversary activities, optimal planning for feasible implementation, and safe deployment without breaking the system's integrity. This article summarized some of the best practices to follow to have a good implementation of the technology. We have tested multiple attack scenarios on a virtual lab and compared the results between a network with deception technology deployed and a network with a traditional intrusion detection system installed.

We conclude our observation on below:

– Placement of decoys can help identify potential information gathering attempts that can bypass intrusion detected system
– Deception technology can be used to identify false negatives and detect cyber attacks which were not identified on traditional intrusion detection systems.
– Decoy attribute is one of the important aspects to make the decoy blended and look more realistic to attacker.

**Acknowledgement.** This work was supported by grant number 12R170.

## References

1. Dickinson, K.: Implementer's guide to deception technologies, SANS Institute Information Security Reading Room, P. 16 (2020)
2. Major, M., Souza, B., DiVita, J., Ferguson-Walter, K.: Informing autonomous deception systems with cyber expert performance data, arXiv preprint arXiv:2109.00066
3. Han, X., Kheir, N., Balzarotti, D.: Deception techniques in computer security: a research perspective. ACM Comput. Surv. (CSUR) **51**(4), 1–36 (2018)
4. Chiang, C.-Y. J., et al.: Acyds: an adaptive cyber deception system. In: MILCOM 2016–2016 IEEE Military Communications Conference, pp. 800–805. IEEE (2016)
5. Srinivasa, S., Pedersen, J.M., Vasilomanolakis, E.: Deceptive directories and "vulnerable" logs: a honeypot study of the ldap and log4j attack landscape. In: 2022 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), pp. 442–447. IEEE (2022)
6. Andrews, K.T.: Deception techniques and technologies in the role of active cyber defense, Ph.D. thesis, Utica College (2020)
7. Bushby, A.: How deception can change cyber security defences. Computer Fraud Sec. **2019**(1), 12–14 (2019)
8. Xu, Y., Chai, S., Shi, P., Zhang, B., Wang, Y.: Resilient and event-triggered control of stochastic jump systems under deception and denial of service attacks. Int. J. Robust Nonlinear Control **33**(3), 1821–1837 (2023)
9. Melhem, H., Dayoub, Y.: A hybrid honeypot framework for ddos attacks detection and mitigation
10. Spitzner, L.: The honeynet project: trapping the hackers. IEEE Sec. Privacy **1**(2), 15–23 (2003)
11. Stumpf, F., Görlach, A., Homann, F., Brückner, L.: Nose-building virtual honeynets made easy. In: Proceedings of the 12th International Linux System Technology Conference, Hamburg, Germany, Citeseer (2005)
12. Lackner, P.: How to mock a bear: honeypot, honeynet, honeywall & honeytoken: a survey. In: ICEIS (2), pp. 181–188 (2021)
13. Srinivasa, S., Pedersen, J.M., Vasilomanolakis, E.: Gotta catch'em all: a multistage framework for honeypot fingerprinting, arXiv preprint arXiv:2109.10652
14. gartner (2019). https://www.gartner.com/peer-insights/search?text=deception
15. Duan, Q., Al-Shaer, E., Islam, M., Jafarian, H.: Conceal: a strategy composition for resilient cyber deception-framework, metrics and deployment. In: 2018 IEEE Conference on Communications and Network Security (CNS), pp. 1–9. IEEE (2018)