Honeypots: Architecture, Detection and Prevention

PARTHIVI SAHU

July 2022

1 Introduction

Organisations constantly confront challenges from sophisticated cyber attackers looking to compromise sensitive data in the quickly changing world of cybersecurity. It's critical to have strong defences in place to properly battle these threats. Honeypots are one such defence method that is receiving a lot of attention. In order to monitor and analyse the activities of cyberattackers, honeypots are specialised decoy devices used to entice them. These systems imitate worthwhile targets, luring attackers to engage with them while safeguarding the security of the real production systems and data.

The goal of the research paper "Honeypots: Architecture, Detection, and Prevention" is to give readers a thorough understanding of honeypots, their architecture, and their function in identifying and thwarting cyberattacks. This study aims to shed light on how organisations might use these deceptive technologies to improve their cybersecurity posture by examining the subtleties of honeypots.

The fundamentals of honeypot architecture, including types and deployment methods, will be covered in detail in this paper. In order to trick attackers into disclosing their strategies and goals, it will examine the underlying technologies that allow honeypots to imitate real systems and services.

In honeypot environments, detection systems are essential for differentiating between trustworthy users and malevolent actors. In order to get important insight into the strategies, weaknesses, and attack routes used by attackers, this paper will examine several methodologies and tools used to identify and analyse their actions.

The research report will also go over preventative methods based on the information gathered from honeypot deployments. It will examine how businesses may strengthen their defences, fix vulnerabilities, and proactively mitigate new threats by using the information gleaned from honeypots.

This study attempts to provide readers with a detailed understanding of honeypots, their architecture, and their consequences for cybersecurity through extensive research and analysis. Organisations can decide whether to use honeypots into their cybersecurity plans in an informed manner by learning about the potential advantages and difficulties connected with these devices.

A summary of the main findings that emphasises the importance of honeypots as a pro-active defence mechanism will be included in the research paper's conclusion. Additionally, it will point out topics for future study and development, highlighting the necessity of constantly improving honeypot technology to successfully counter growing cyber threats.

2 Related Works

Honeypots are designed to trap the attacker and investigate its malicious behaviour. As the risk of cyber attacks is increasing.Therefore the need of new honeypot architecture has been increased in order to improve its functioning. For example, Fan et al.[2] have proposed a novel architecture method named HoneyDOC to make improvements in all-around honeypot design and its implementation. A proof-of-concept system was implemented to validate its feasibility and effectiveness. This architecture mainly used the Decoy-Orchestrator-Captor perspective to dissect and decouple the honeypot. SDN-enabled architecture has also enabled the demonstrated design . This architecture improved the efficiency of data reduction and traffic redirection for data analysis. The experimental results have shown the feasibility and efficiency of the proposed architecture. Noaman et al.[9] have also proposed an architecture which relies on virtualization and various software agents. This prototype has been implemented and validated which achieved a success rate of 98 % .The architecture has been designed using software agents, a secure proxy server, and a real-time communication (RTC) monitoring and cluster server. The advantage of this has been the adoption of RTC servers to reduce the delay in transferring data from a production server. The secure decoy port mapping and the secure forwarding of malicious traffic by the software agent to honeynet servers had other notable advantages of the proposed architecture. As deception has been considered a method for strengthening software security, Papalitsas et al.[10] have discussed the idea of deceiving attackers through fake services, which has helped them to find out more about malware's functionality and to hamper cyber intelligence. The properties required from a tool generating fabricated entities have also been discussed. The final design for a honeypot proxy which has generated fallacious content for fake services for deceiving, they have also tested the implementation's accuracy and performance. Experiments with example plugins that they have implemented for the proxy shows it can recognize entities accurately, succeeds well in keeping the original entity values safe and does not produce an unacceptably large performance overhead. Kostopoulos et al.[4] presented the Honeypot architecture named YAKSHA and investigated its deployment within an IoT pre-commercial environment. YAKSHA has enhanced the levels of cybersecurity readiness for its end users, help in preventing cyberattacks, reduced cyber-risks, and improved the whole cybersecurity process. The essential goal of YAKSHA was to enable end-users, whether they were from government, organizations or companies, to easily set up customized honeypots, which has allowed them to understand how their systems are being attacked autonomously. Kandanaarachchi et al. [3] have introduced a new framework called Honeyboosts that operates on the LAN rather than the perimeter and focuses on internal traffic. For example, to predict malicious insider attacks. Honeyboost incorporated Network Anomaly Detection System (NADS) and honeypots within an internal network. The anomaly detection technology has been integrated into Honeyboost. It was an unsupervised method comprising of two approaches namely horizontal and vertical. The horizontal approach consisted of communications via ARP, TCP, or UDP whereas the vertical approach found anomalies in each protocol. Both approaches were combined to identify anomalies. The results indicated the efficacy of the framework.Surnin et al.[15] proposed software architecture and implemented various problems of a specific honeypot. They considered that the only way to fix the honeypot was by changing its implementations. They implemented open-source software based on their methodology. It was a fully-functional prototyping protocol, which allowed honeypot detections with a certain degree of probability. The approach provided the opportunity for developers to extend their software with additional methods. The proposed architectural model has extended its support for other types of honeypots.

Honeypots have been used extensively, but their development has a rare focus on understanding how attackers can detect them. So, Tsikerdekis et al. [17] have presented a classification of honeypot characteristics along with honeypot detection evasion strategies which minimize the detection rates of honeypots. As we know honeypots are mainly intelligence-gathering devices; their designs and implementations must also change with time to make them more secure and to catch new attacks which will make them compatible within the ever-changing world of cyber security. To make it happen we need to have different ideas and architectures. They also provide some recommendations to make honeypot software more adaptable, and modular and to incorporate a dynamic intelligence design. Zobal et al.[19] discuss honeypot's history, and the role it plays in the domain of cybersecurity. They also defined honeypots and their abilities. They also discussed various taxonomy of honeypots, namely low, medium and high interaction honeypots. They also divided honeypots into research honeypots, production honeypots, server and client honeypots, and physical and virtual honeypots. Here, they discuss various advantages and disadvantages of honeypots. Different examples were compared simultaneously to give us a better understanding including the challenges that were faced by honeypots and then concluding it with data analysis and interpretation. From a data analysis perspective, honeypots provided easy-to-understand visualization and intelligent attack clustering. Zulkurnain et al. [20] improved Thug, which has managed to capture malicious viruses and malware that the previous Thug was not able to detect. As information gets exploited and certain communities like black hat hackers, frequently use networks and servers that are commonly used to gain unauthorized information and data on the victim. The new Thug had improved scanning speed; the new scanning time was negligible enough for the benefits to outweigh the scanning speed. The Thug had also improvised the ability to add cypher machine code to Thug features as the current one wasn't able to decode the machine code. Ren et al.[14] have proposed and analyzed a new dynamic model for assessing the efficacy of honeypots. Their work was a heuristic to assess honeypot due to its universality of honeypot deployment by mathematical analysis is the maximum characteristic value of the deployed network. Especially the number of metrics clearly defines the boundaries of a spread between two explicit branches. Below this boundary, the Honey Web worked best until the malware disappears, and beyond that, the malware survives at a certain level. Evaluating the effectiveness of the formulated model required a large collection of real data in a network deploying a honeynet and a series of observations of network behaviour, which demanded long-term steady efforts. Once malware-propagated dynamics were revealed under the honeypot background, it usually took a lot of time to improve defence strategies. The results showed that taking practical measures explicitly enhanced honeypot potency. As the network is filled with attackers which were both automated and live sought to identify and compromise vulnerable devices. For a better understanding of these events, Thom et al.[16] discussed that honeypots can be deployed to monitor and log activity by simulating actual Internet-facing services such as SSH, Telnet, HTTP, or FTP, and malicious activity. The data was then correlated across honeypots to compare attack and traffic patterns to learn more about the tactics being employed. By gathering data gathered from geographically separate zones over a long period a greater understanding was developed regarding attacker intent and methodology, which was an aid in the development of effective approaches to identifying malicious behaviour and attack sources, and served as a cyber-threat intelligence feed. Wang et al.[18] observed that attackers tend to organize different botnets that target the same victim host competitively or cooperatively. Furthermore, they monitored and analyzed large and complicated networks, which was challenging. Their study presented a data-driven study to analyze IoT botnets and attack patterns. Besides, they also proposed a recursion-based algorithm, which identifies botnets based on control periods. It was also said that bots tend to show time-related attack patterns within the same botnet control period. Experiments have demonstrated the method's efficacy. They also presented eight interesting findings that helped the security community to understand better and fight botnet attacks then and in the future.

Defending Industrial Control System (ICS) in the cyber domain has both helped and hindered by bespoke systems that integrated heterogenous devices for unique purposes. Because of this fragmentation, observed attacks against ICS have been targeted and skilled, making them difficult to identify prior to initiation. Dodson et al.[1] in their research exposed four zero-day attacks against devices that were running on common ICS protocols such as S7comm and Modbus, which were then disclosed to the applicable manufacturers. They also demonstrated the gap between ICS-aware and IoT-aware hosts, which was narrowed, and also showed that IoT malware was co-located with ICS devices and scanners. Bridging this gap was the first major hurdle in attacking ICS devices at scale. Finally, they discussed the limitations of previous ICS honeypot studies and provided recommendations. Rashid et al.[12] deployed three low-interactive multi-platform honeypots in three different locations to lure cyber criminals to attack the networks. They performed large-scale analysis and observed the attack trends toward Industrial Control System, and captured adversaries like malicious activities and techniques for adaptive threat defense systems in the future. They knew the attacker's processes and methods in which honeypot was used to help dor taking early measures to protect ICS devices from cyber-attacks. During their experiment, they also used default templates which were able to be recognized by advanced attackers. Maesschalck et al. [6] reviewed previous ICS honeypot deployments in their literature outlines ICS-focused threat vectors and discusses how they integrated honeypots into their organization's defense strategy. It also described that the relevant legislation, including the UK Cyber Assessment Framework, the US NIST Framework for Improving Critical Infrastructure Security, and related industry-based standards and guidelines, supports operator compliance. It was used to formulate discussions on the role of honeypots in investigating the implementation of existing ICS honeypots and supporting regulatory objectives. It turns out that many of the less interacting honeypots have limited use. Matin et al.[7] studied and analyzed 11 research papers. Therefore, from the results of the literature, it was concluded that the use of honeypot in malware detection-based learning has increased from 2017 to 2019. The techniques used by most researchers are utilizing available honeypot datasets. Meanwhile, based on the type of malware analyzed, a honeypot in machine learning is mostly used to collect IoT-based malware. Significant research trend improvement occurred in 2019. According to the method of use, virtualization techniques were more frequently used with the honeypot. In other words, a honeypot was set up to collect data, and then modelling-based data was collected. The model was directly based on the given dataset by using a dataset rather than installing a honeypot.

An intrusion detection system (IDS) basically monitors network traffic for abnormal activity and sends out warnings when it is found. It is software that checks a system or network for malicious activities or policy violations. Any illegal activity or violation is often recorded either centrally using a security information and event management (SIEM) system or notified to an administrator. Kyriakou et al.[5] deployed multiple honeypot sensors to monitor and study the actions of the attackers. Multiple honeypots such as Cowrie, Dionaea and Glastopf, were presented as a Linux host, a Windows host and a Web application, respectively, which enabled them to have a diverse and broad environment. The creation of a single dashboard presents the captured data effectively in real-time and both at macroscopic and microscopic levels. The sensors were running on a containerization platform, Docker, and in this way, they were lightweight, resilient and easily deployed and managed. The main purpose of honeypot systems is to collect more and more attack traffic. Park et al.[11] proposed Dynamic Virtual Network Honeypot (DVNH) architecture which leverages the emerging technologies, Network Function Virtualization and Software-Defined Networking. DVNH redirected the attacks to the honeypot system, thereby protecting the targeted system. To overcome the existing virtual honeypot deployment problem, the DVNH system dynamically managed the virtual honeypot and provided a honeypot when Network IDS detected a new attack. Their prototype implementation and experimental results demonstrated that the mechanism was efficient to manage and for the provision of virtual honeypot in Software Defined Networking environment. Their experiments show that DVNH enables efficient resource usage and dynamic provisioning of the Honeypot system. Mayorga et al. [8] studied the mechanisms of computer security concerning the protection of a data network. An attack is generally directed towards the data network or the services it provides, they sought to determine the origin of the attack by simulating a network trap that allowed them to obtain data from the attacker and the type of attack. They also have analyzed the results of the Honeypot as activities of a hacker or attacker who was accessing the network like brute force attack, Denial of service attack, and man-in-the-middle attack from these alerts the system so that administrator can take preventive measures. A Honeypot in production was configured to work as a virtual system or a physical system, using tools such as Honeywall, which acted as a firewall system, and detected all incoming and outgoing connections to the Honeypot. The study of various honeypots and their log records can reveal a lot about the origin of the attack along with other information. Ravji et al.[13] proposed a plan to deploy a Honeypot in the Intrusion Detection and Prevention System (IDPS) model to guarantee enhanced performance, an expanded level of security in the Cloud computing environment and a reduction in the dangers to the Cloud environment. The framework used both Anomaly Detection (AD) and Signature Detection (SD) in collaboration, which identified different attacks and denied access through the use of the proposed Intrusion Prevention System (IPS). Therefore, the honeypots can also be used for logging information on the threats, researching the threats, and the application of any defense measures to stop the attack.

The table below shows how the researchers have contributed to the study of honeypots in the mentioned parameters along with the types of honeypot.

Table 1: Comparative analysis of research papers .

Author, year	Key Contributions	Archit-ecture	IDS	ICS	Prev-enti-on	Dete-ction	Inter-action of honey-pot	Form of honey-pot	Purp-ose of honey-pot
Fan et al. , 2019	Improved efficie-ncy and good traffic control.	Yes	No	No	Yes	Yes	High	Virtual	Produ-ction
Tsikerdekis et al., 2018	Several approa-ches to ensure detection avoidance for honeypots.	Yes	No	No	Yes	Yes	High	Virtual	Produ-ction
Zobal et al., 2019	Advantages and disadvantages of honeypots were presented and they were classified into different metrics	No	No	Yes	Yes	Yes	All	Both	Both
Dodson et al., 2020	Demonstration of a network that has high-interaction honeypots which can identify previo-usly unknown profile and tar-geted ICS attacks.	No	No	Yes	No	Yes	High	Virtual	Produ-ction
Surnin et al., 2019	Software archi-tecture and implementation problems of the specific honeypots.	Yes	No	No	No	Yes	All	Both	Resea-rch
Matin et al., 2020	The research trends on the use of honeypot to detect malware fluctuation.	No	No	No	No	Yes	High	Virtual	Resea-rch
Kyriakou et al., 2018	The improved honeypots were successful in capturing a large number of connections and many indicative malware samples.	Yes	Yes	No	Yes	Yes	High	Virtual	Produ-ction

Author, year	Key Contributions	Archit-ecture	IDS	ICS	Prev-enti-on	Dete-ction	Inter-action of honey -pot	Form of honey- -pot	Purp-ose of honey -pot
Wang et al., 2022	A recursion-based novel algorithm, which identifies botnets based on control periods.	No	No	No	Yes	Yes	High	Virtual	Both
Thom et al., 2021	Identified patterns and activity which was useful in identifying malicious traffic in actual servers or IoT devices.	No	Yes	No	Yes	No	Medi-um	Virtual	Produ- -ction
Noaman et al., 2019	A novel honeynet architecture is designed using software agent, secure proxy server, and real-time communication (RTC) monitoring and cluster server.	Yes	Yes	No	Yes	Yes	All	Virtual	Both
Zulkurnain et al. , 2018	The improved honeypot has managed to capture malicious viruses and malware that the previous honeypot has not detected.	No	Yes	No	Yes	Yes	High	Both	Both
Papalitsas et al. , 2018	It provides one-way backend integration between the real service and the honeypots, to maintain the illusion of continuity in terms of content and session.	Yes	No	No	Yes	Yes	Low	Physical	Rese- -arch

Author, year	Key Contributions	Archit-ecture	IDS	ICS	Prev-enti-on	Dete-ction	Inter-action of honey -pot	Form of honey- -pot	Purp-ose of honey -pot
Park et al. , 2019	To overcome the existing virtual honeypot deployment problem, the DVNH system dynamic manages virtual honeypot and provides honeypot when Network IDS detects a new attack.	Yes	Yes	No	Yes	Yes	Low/ High	Virtual	Rese- -arch
Kostopoulos et al. , 2020	YAKSHA analytics capability is used to raise awareness and provide decision support in strengthening the cybersecurity posture of the product.	Yes	Yes	Yes	Yes	Yes	High	Virtual	Produ- -ction
Rashid et al. , 2019	They used default templates which could be recognized by advanced attackers. Honeypot was used to protect ICS devices from cyber attacks.	No	Yes	Yes	No	Yes	Low	Virtual	Rese- -arch

Author, year	Key Contributions	Archit-ecture	IDS	ICS	Prev-enti-on	Dete-ction	Inter-action of honey -pot	Form of honey- -pot	Purp-ose of honey -pot
Mayorga et al., 2019	Helped to improve the protection systems, the alerts allowed them to know what happened in the network traffic, they can monitor by means of a Honeypot administrator.	No	Yes	No	Yes	Yes	High	Virtual	Produ- -ction
Ravji et al. , 2018	Deployment of high-interaction honeypots increased the use of its bandwidth and the success rate of intrusion detection will also be more.	No	Yes	No	Yes	Yes	Low	Virtual	Rese- -arch
Kandanaa-rachchi et al. , 2022	A novel, hybrid framework was developed consisting of two complementary approaches horizontal and vertical to enhance honeypot aided NAD, it adds another tier of security to the LAN.	Yes	Yes	No	Yes	Yes	Low	Physical	Rese- -arch

Author, year	Key Contributions	Archit-ecture	IDS	ICS	Prev-enti-on	Dete-ction	Inter-action of honey -pot	Form of honey--pot	Purp-ose of honey -pot
Ren et al. , 2021	The honeypots deployed on various computer networks could play an appropriate role, confirming their effectiveness and this architecture was validated by numerical simulation.	Yes	No	No	No	Yes	Low	Physical	Rese--arch
Maesschalck et al. ,2021	Honeypots in an industrial control system environment, when used properly, could provide a wealth of data related to the threats faced by an organization.	No	Yes	Yes	Yes	Yes	All	Both	Rese--arch

3 Methodology

3.1 Background:

Honeypot refers to a network-attached system that creates a virtual trap for the cyber-attackers to lure them and redirect attackers from their intended target. It helps study different hacking attempts that are used to gain unauthorized access to the information systems with the attacker's behaviour pattern. Honeypots are made attractive to attackers by building different types of deliberate security vulnerabilities, like they may have ports that respond to a port scan or might have weak passwords. Unlike a firewall or anti-virus, it is an information tool that helps in understanding the existing threats and spotting new threats. They also reduce the risks of false positives compared to traditional cybersecurity measures because they can't attract attackers. Honeypots vary based on the design, deployment models, interaction levels and physicality. Honeypots are specialized decoy systems designed to lure cyber attackers, providing a controlled environment for monitoring and analyzing their activities. These systems mimic valuable targets, enticing attackers to interact with them while keeping the actual production systems and data secure. To combat these threats effectively, it is crucial to have robust defense mechanisms in place. One such defense mechanism gaining significant attention is honeypots.

3.1.1 Taxonomy of Honeypots :

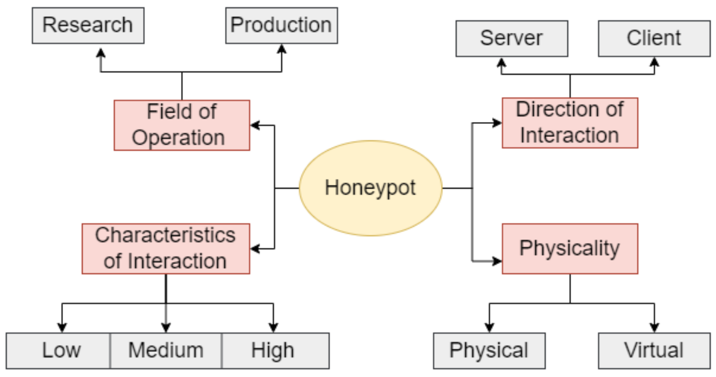

Figure 1: Fig 1: Categories of honeypots

Honeypots based on their field of operation:

1. Production Honeypots: The production honeypots are to emulate real production systems, used in the production environment of companies. As it involves the attackers spending their time and resource attacking the systems along with a way to learn how they exploit the vulnerabilities. It is therefore used to achieve a higher security level within a network. For example, we can examine the attacks by deploying production honeypots in the web server as they can be deployed internally to discover the loopholes.

2. Research Honeypots: The research honeypot is a type of honeypot used for collecting information about the specific pattern or method that hackers use. It collects information about different attacks and vulnerabilities. Unlike the production honeypot, research honeypots are used in government and research organizations. They are also more complex than the production honeypot.

Honeypots based upon the characteristics of interaction:

1. Low Interaction Honeypot: Low interaction honeypots give limited access to the operating system. They are only able to interact with a small number of internet protocols and network services to deceive the attacker. It generally deploys and does not give access to a real root shell. Examples of low-interaction honeypots are Nepenthes and Honeyd.

2. Medium Interaction Honeypot: Medium interaction honeypots provide a little better ability to interact than low-interaction honeypots. They are designed to give certain responses, like worm scanning for some vulnerabilities. It helps in getting some collections of software emulated so that the attacker can be convinced that it is the real system. Examples of medium-interaction honeypots are Mwcollect and honeytrap.

3. High Interaction Honeypots: High-interaction honeypots include an operating system. It is the most advanced type of honeypot, which can lure the attacker and get information about the attack type, source, as well as nature. These have no restrictions and can perform different tasks that are desired by the user, which also makes them time-consuming to configure and implement. An example of a high-interaction honeypot is Honeywell.

Honeypots based on the direction of the interaction:

1. Server Honeypots: The server honeypots generally wait until the attackers start communication. The traditional approach to honeypots' working is to maintain a server that waits for the attackers. A Windows Honeypot machine and a Linux Honeypot machine are examples of server honeypots.

2. Client Honeypots: Client honeypots are usually represented by web browser clients crawling malicious websites. They search for potential malicious items and then request interaction. The high-interactive honeypots use client-honeypots. HoneyC and SpyBye are a few examples of client honeypots.
Honeypots based on the physicality :

1. Physical Honeypots: These are actual machines with their own IP address. They use physically dedicated hosts for each honeypot. It is more costly compared to the virtual honeypot as it requires a proper setup. It is less feasible to use a physical honeypot for dynamic concepts as it is less efficient, inflexible and also requires a huge amount of investment in terms of cost, labour and hardware for implementation. A Honeyboost is an example of a physical honeypot.

2. Virtual Honeypots: A virtual honeypot is a machine that is virtually simulated and has modelled behavior. They are more feasible than physical honeypots because they require fewer computer systems, which eventually reduces maintenance costs. Certain virtualization tools like VMware and KVM have certain functions that can be used for creating a high-interaction honeypot. YAKSHA and HoneyDOC are examples of virtual honeypots.

3.1.2 Framework of Honeypots :

YAKSHA [4] has one of the most effective model architecture for honeypot software which can be used in Smart Home solutions also. YAKSHA consists some basic VMs :

- A VM which helps YAKSHA to handle the front-end. Vagrant manages the VMs. Honeypot has custom scripts which helps it to perform monitoring and data collection.
- A cuckoo host that helps in managing all the sandboxes which performs dynamic analyses.
- Reporting engines that generates reports.

The YAKSHA control and command center hosts the following events:

- Web administration for cuckoo monitored VMs
- web administration for VM of honeypot server.
- Mongo DB

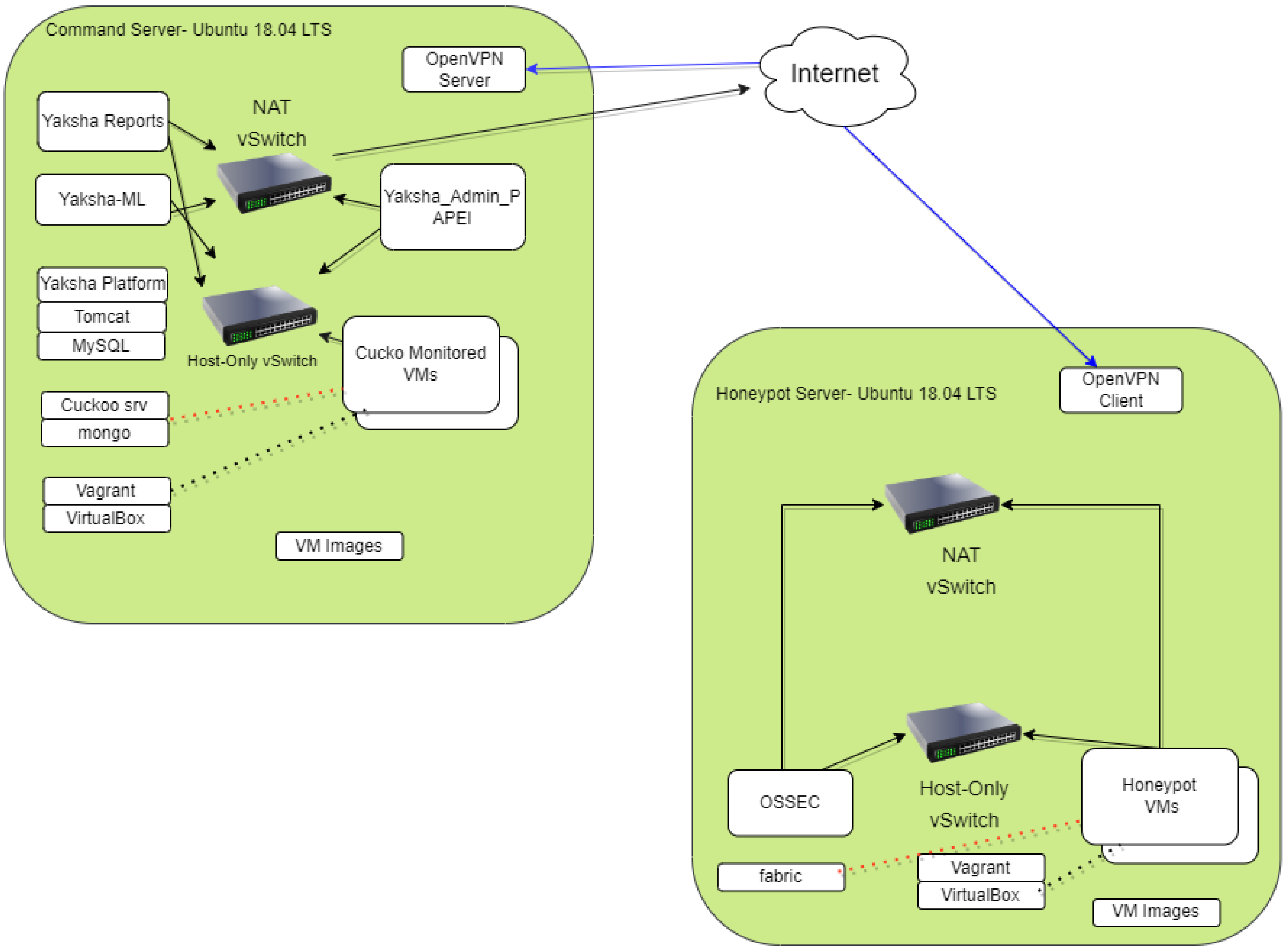

Figure 2: Fig 2: YAKSHA logical architecture

3.1.3 Features of Honeypots :

Honeypots have various features that are explained as follows:

Data Collection: Honeypots are usually used to collect data which is not disturbed by production activities and therefore of higher value. This helps in making data smaller and the analysis of data less complex.
Less Workload: Honeypots only need to process the traffic which is directed towards them, which means they are independent and have less workload in the experience of production systems.

Zero-Day-Exploit Detection: Honeypots are used to capture everything that is utilised against them, thus they can aid in the discovery of various undiscovered tactics and zero-day vulnerabilities.

Flexibility: The honeypots are very flexible as they can be adjusted and various tools can be used to modify them, making them mould themselves according to the needs.

3.1.4 Benefits of Honeypot :

Finding flaws in crucial systems is a great idea when using honeypots. For instance, a honeypot can show the real threat that attacks on IoT devices represent. Additionally, it could offer suggestions for ways to strengthen security. There are several advantages to comparing a honeypot to a genuine system while looking for intrusions. Any action that is logged is almost probably a probe or intrusion attempt, as a honeypot, for example, should never receive real traffic. The use of similar IP addresses (or IP addresses that are all from the same country) to conduct network sweeps is one example of a trend that is much easier to spot as a result. In contrast, when you are focusing on enormous volumes of genuine traffic on your core network, such warning indicators of an attack are simple to miss in the cacophony. The main benefit of adopting honeypot security is that you might only see these malicious addresses, which makes it much easier to spot an attack.

3.1.5 Legal and Ethical Issues

Honeypot deployment requires a huge amount of discussion along with legal and ethical liabilities. The structure of a honeypot needs to be defined according to the laws of the respective countries. It involves entrapment challenges and privacy challenges. The act of convincing a subject to commit a crime when no prior intention to do so exists is known as entrapment. One could contend that honeypots are exempt from this because they don't aggressively convince anyone. Server honeypots imitate production systems or services and wait for incoming connections. Unless other network users choose to approach those systems, they are essentially invisible to the rest of the network.The initial request is made by client honeypots. However, because the exploitation is already prepared server-side, which implies that the intention of a crime already exists, they cannot convince the victim to conduct a crime. The privacy concern goes into more detail about data acquisition. It is debatable whether it is appropriate for a honeypot to gather details about the attackers without their knowledge or consent and so breach their right to privacy.

3.2 Problem Statement :

As we have seen in the table above that many authors have given their contribution in developing a more perfect honeypot. A lot of work is done in building a good architecture,interaction of honeypots along with its forms and purposes but the parameters like detection and prevention of honeypots have lacked a little. The nearest to the perfect approach was given by Ahmed Noaman et al. [9], Byungju Park et al. [11] and Alexandros Kostopoulos et al. [4]. The architecture of the honeypot has been changing frequently with the development of technologies throughout the years. There have been various architectures have been applied to the honeypots to make their functionality better. The architecture of the honeypot comprises of Real-Time Communication Cluster Server(RTC), Software Agent, Security Proxy server, Honeywall, Load Balancer, Virtual honeynet server farms, database log server and malware analysis engine. In this research paper, we aim to find the most ideal architecture for a honeypot. To learn about the architecture of the honeypot, we first need to know about its history of architecture, the honeypot architecture that emerged with the deployment of Honeynets. [9] The honeynet project was a series of physical honeypot architectures which also included Gen I, Gen II and Gen III which was widely used by the organizations. Gen I honeypots were made to manage compromised computer systems across the network, it lacked the capability of large-scale deployment. In Gen II the virtualisation of the honeypot began which can make a single host run multiple guests. In Gen III we get the hybrid version of honeypots which can work as both physical and virtual honeypots. To find a novel architecture for honeypots we first need to see the problems surrounding it. [19] When we deal with the problems regarding honeypots, it can be divided into two fields. The primary field is regarding the development of honeypots, their deployment and sustainable maintenance. The other is based on the analysis of collected data and information, visualization, and decision-making based on the data.

3.3 Proposed Solution :

The current global trend is massive amounts of data, and investigating cyber-attacks is no exception. Although honeypots drastically reduce the amount of data, it is necessary for honeypot users to have the capability to quickly extract useful information[19]. The problems like detection, analysis of data and architecture which play major roles can be done by dealing with proper solutions like:

Honeypot Development and Detection:
Honeypot is vulnerable when exposed or identified by the attacker, therefore we need to protect its detection. Low-interaction honeypots should be more secured as they only simulate the servers, which makes it difficult to prevent from detection. Therefore,[17] some approaches that can help in preventing the detection are:

a) Automatic Honeypot Redeployment: As Low-interaction honeypots have limited scope and functionality in comparison to the high-interaction honeypots, therefore it becomes more expensive to built an anti-detection system. According to a study, the approach to avoid detection can be done through automated re-deployment of honeypot which can voluntarily reduce the need of anti-detection.

b) Honeypot delay reduction: This technique combines the use of IP address spoofing and false TCP three-way handshake to undermine the authentication process used to permit infected machines to join a botnet. It is used to prevent honeypot detection in P2P botnets. The Advanced Two-Stage Reconnaissance Worm was the writers' main emphasis (AT-SRW).

The other methods like Honeypot process transparency, dedicated hardware, and dynamic intelligence can also be used to avoid the detection of honeypots.

Data Analysis and Interpretation:
It explains how the attacked profile might provide specific information about the attacking vectors, which can be categorized as follows:[19]

- Attack Source: Information regarding the attacker's identity is gathered. This may include OS, URLs, user agents, and IP addresses with their geolocation.
- Attack Target:Similar details as in the prior item, but with regard to the victim. This is often the honeypot itself, but we can monitor which ports were attacked in server honeypot cases, what software in client honeypot cases, what IP address range or OS in honeynet cases, etc.

- Attack Frequency: These functions include information on statistics, such as the number of attacks per time unit, data/packets/messages received per time unit, the interval between attacks, the duration until the first attack, etc.

- Attack Evolution: Various measures for detecting anomalies and irrational patterns.

- Attack Patterns: The potential for grouping attacks is covered in this category.

- Exploit Detection: In this area, researchers explore methods for detecting exploitation on honeypots with high user interaction, such as operating system monitoring or data-driven techniques (Argos) (Capture-HPC).

  The ideal framework for the architecture is discussed above in section 3.1.2 framework of honeypots.

4 Evaluation

The proposed solutions were evaluated by reading 20 different research papers and doing analysis on the problem that makes honeypots detectable. After that we came up with approaches to make honeypots a little less detectable and to perform better.

5 Future work

The future work will have a comparison between 20 honeypot software from which we can detect their pros and cons. It can help us to learn more about the different architectures along with detection and prevention techniques. It will include honeypot software like HoneyBOT, honeyperl, HoneyD, Netbait, KFSensor, Honeytrap, SpamD, Jackpot, HoneyDroid, HoneySink, HoneyBow sensor and many more. This analysis will help us in designing a more knowledge-based system. We can also apply machine learning to the honeypots in future for advancing in terms of technology. The architecture will have few more artificial intelligence components which can help the honeypots to function more effectively. I t will focus on predicting the attacker's techniques. An architecture for honeypots which can be done by research is to build a virtual hooneypot like Byungju Park et al.[11] for the modern honeypots.

A basic overview of virtual honeypot is given below:

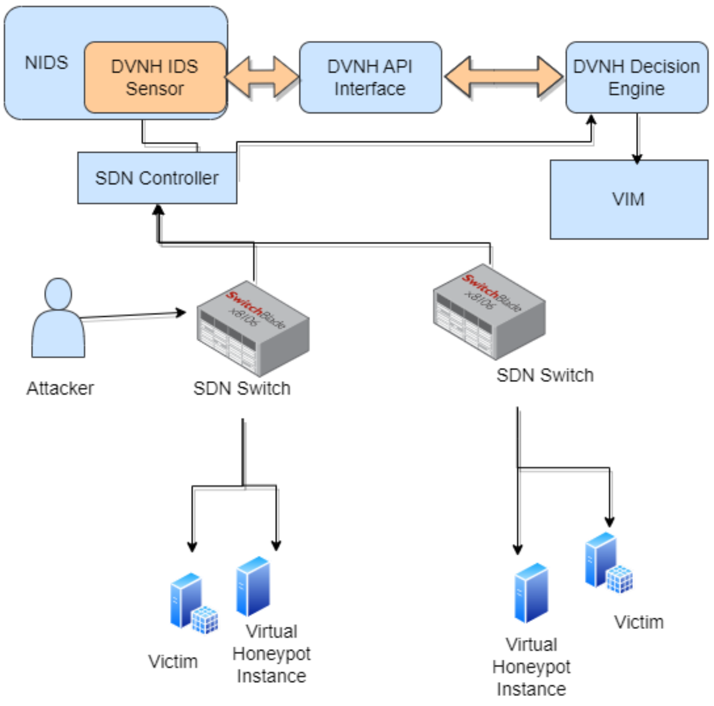

Figure 3: Fig 3: Dynamic Virtual Honeypot Network

6 Conclusion

In conclusion, honeypots have become effective cybersecurity tools, giving organisations a cutting-edge method for identifying, assessing, and preventing cyberattacks. The fundamental characteristics of honeypots, their architecture, and their function in bolstering cybersecurity defences have been examined in the research article titled "Honeypots: Architecture, Detection, and Prevention".

We have discovered the adaptability and potency of these systems by a thorough investigation of honeypot kinds and deployment techniques. Honeypots lure attackers by imitating real systems and services, offering a controlled setting for observing their actions and comprehending their strategies. The significance of detection techniques inside honeypots has been emphasised in the research report. Organisations can identify and examine the actions of hostile actors using a variety of approaches and technologies, learning important information about their methodologies, weaknesses, and attack vectors. With the use of this information, organisations may strengthen their defences, fix vulnerabilities, and neutralise new threats.

In addition, the research highlights the use of honeypots in preventative tactics. Organisations can proactively improve their cybersecurity posture, spot possible holes, and take preventive action to protect their vital systems and data by utilising the insights gained from honeypots. It is critical to understand the potential of honeypots and their continual growth as the cybersecurity landscape changes. For better threat identification and prevention, future research and development efforts should concentrate on developing honeypot technologies, evasion strategies, and combining honeypots with threat intelligence platforms and machine learning algorithms.

In conclusion, honeypots provide a distinctive and useful method for cybersecurity. The study paper has given readers a thorough grasp of honeypots, their design, methods of detection, and defences against them. Organisations can significantly improve their ability to keep ahead of cyberthreats, fortify their defences, and protect their vital assets by properly adopting honeypots.

References

[1] M. Dodson, A. R. Beresford, and M. Vingaard. Using global honeypot networks to detect targeted ics attacks. In *2020 12th International Conference on Cyber Conflict (CyCon)*, volume 1300, pages 275–291. IEEE, 2020.

[2] W. Fan, Z. Du, M. Smith-Creasey, and D. Fernandez. Honeydoc: an efficient honeypot architecture enabling all-round design. *IEEE Journal on Selected Areas in Communications*, 37(3):683–697, 2019.

[3] S. Kandanaarachchi, H. Ochiai, and A. Rao. Honeyboost: Boosting honeypot performance with data fusion and anomaly detection. *Expert Systems with Applications*, 201:117073, 2022.

[4] A. Kostopoulos, I. P. Chochliouros, T. Apostolopoulos, C. Patsakis, G. Tsatsanifos, M. Anastasiadis, A. Guarino, and B. Tran. Realising honeypot-as-a-service for smart home solutions. In *2020 5th South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference (SEEDA-CECNSM)*, pages 1–6. IEEE, 2020.

[5] A. Kyriakou and N. Sklavos. Container-based honeypot deployment for the analysis of malicious activity. In *2018 Global Information Infrastructure and Networking Symposium (GIIS)*, pages 1–4. IEEE, 2018.

[6] S. Maesschalck, V. Giotsas, B. Green, and N. Race. Don't get stung, cover your ics in honey: How do honeypots fit within industrial control system security. *Computers & Security*, page 102598, 2021.

[7] I. M. M. Matin and B. Rahardjo. The use of honeypot in machine learning based on malware detection: A review. In *2020 8th International Conference on Cyber and IT Service Management (CITSM)*, pages 1–6. IEEE, 2020.

[8] F. Mayorga, J. Vargas, E. Álvarez, and H. D. Martinez. Honeypot network configuration through cyberattack patterns. In *2019 International Conference on Information Systems and Computer Science (INCISCOS)*, pages 150–155. IEEE, 2019.

[9] A. Noaman, A. Abdel-Hamid, and K. Eskaf. A novel honeynet architecture using software agents. In *2019 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT)*, pages 1–6. IEEE, 2019.

[10] J. Papalitsas, S. Rauti, J. Tammi, and V. Leppänen. A honeypot proxy framework for deceiving attackers with fabricated content. In *Cyber Threat Intelligence*, pages 239–258. Springer, 2018.

[11] B. Park, S. P. Dang, S. Noh, J. Yi, and M. Park. Dynamic virtual network honeypot. In *2019 International Conference on Information and Communication Technology Convergence (ICTC)*, pages 375–377. IEEE, 2019.

[12] S. Z. U. Rashid, M. J. Uddin, and A. Islam. Know your enemy: analysing cyber-threats against industrial control systems using honeypot. In *2019 IEEE International Conference on Robotics, Automation, Artificial-intelligence and Internet-of-Things (RAAICON)*, pages 151–154. IEEE, 2019.

[13] S. Ravji and M. Ali. Integrated intrusion detection and prevention system with honeypot in cloud computing. In *2018 International Conference on Computing, Electronics & Communications Engineering (iCCECE)*, pages 95–100. IEEE, 2018.

[14] J. Ren, C. Zhang, and Q. Hao. A theoretical method to evaluate honeynet potency. *Future Generation Computer Systems*, 116:76–85, 2021.

[15] O. Surnin, F. Hussain, R. Hussain, S. Ostrovskaya, A. Polovinkin, J. Lee, and X. Fernando. Probabilistic estimation of honeypot detection in internet of things environment. In *2019 International Conference on Computing, Networking and Communications (ICNC)*, pages 191–196. IEEE, 2019.

[16] J. Thom, Y. Shah, and S. Sengupta. Correlation of cyber threat intelligence data across global honeypots. In *2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 0766–0772. IEEE, 2021.

[17] M. Tsikerdekis, S. Zeadally, A. Schlesener, and N. Sklavos. Approaches for preventing honeypot detection and compromise. In *2018 Global Information Infrastructure and Networking Symposium (GIIS)*, pages 1–6. IEEE, 2018.

[18] H. Wang, H. He, W. Zhang, W. Liu, P. Liu, and A. Javadpour. Using honeypots to model botnet attacks on the internet of medical things. *Computers and Electrical Engineering*, 102:108212, 2022.

[19] L. Zobal, D. Kolář, and R. Fujdiak. Current state of honeypots and deception strategies in cybersecurity. In *2019 11th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, pages 1–9. IEEE, 2019.

[20] N. F. Zulkurnain, A. F. Rebitanim, and N. A. Malik. Analysis of thug: A low-interaction client honeypot to identify malicious websites and malwares. In *2018 7th International Conference on Computer and Communication Engineering (ICCCE)*, pages 135–140. IEEE, 2018.