e-palli

AMERICAN JOURNAL OF INNOVATION IN SCIENCE AND ENGINEERING (AJISE)

ISSN: 2158-7205 (ONLINE)

VOLUME 1 ISSUE 1 (2022)

Indexed in

doi | Crossref | Google Scholar | Directory of Research Journals Indexing DRJI | INDEX COPERNICUS INTERNATIONAL | OCLC WorldCat

PUBLISHED BY: E-PALLI, DELAWARE, USA

American Journal of Innovation in Science and Engineering (AJISE)

Volume 1 Issue 1, Year 2022
ISSN: 2158-7205 (Online)
DOI: https://doi.org/10.54536/ajise.v1i1.453
https://journals.e-palli.com/home/index.php/ajise

DNS Cache Poisoning/Honeypot Analysis Based on Data Exfiltration Using Stochastic Petri Nets Method to Enhance Cyber Security Hygiene

Akhigbe-Mudu Thursday Ehis[1*]

Article Information

Received: August 02, 2022

Accepted: August 21, 2022

Published: August 24, 2022

Keywords

Domain Name System, Deadlock, Exfiltration, Security Chain, Stochastic Petri Nets

ABSTRACT

According to some experts, organizations make investments of huge sum of dollars concerning firewalls, encryption, and other tightly closed access devices, yet it is all naught, considering that none regarding these solutions tackle the weakest link within the safety chain. That speech perfectly captures the present day stress experienced by skilled network protection specialists. Researchers' threat detection techniques can pick up a lot of false positive alarms throughout the detection process; false positive alarms are accountable for company's unneeded system lock down. A stochastic model is necessary to represent a communication system since the nature of the traffic between them is unpredictable. SPN was utilized in this study to build statistical model for networks with security chains. By permitting the formalization of both real-time and non-Markovian behavior, the new stochastic Petri net formalism improves model fidelity. This allowed us to see special structures within the stochastic processes produced by SPN models. We have applied this principle by proposing an effective simulation method that supports deadlock detection and easy-to-compute point estimates and confidence intervals. The method is novel because it can automatically detect hidden regenerative structures that do not conform to different simple conditions, and can be easily determined by analytical methods.

INTRODUCTION

Let's take a look look at what the Domain Name System (DNS) is before discussing DNS poisoning. DNS is the online equivalent of the phone book. When looking for a company address in the past, one would consult the yellow pages. DNS is similar to that, with the exception that your computer's internet connection does all the searching for you. A computer must have an IP address in order to communicate with another computer across an Internet Protocol (IP) network (Ellard et al., 2015). Consider an IP address to be similar to a street address; in order to find another computer, one must know its number. Since most people can remember names better than anything else,

A DNS transforms an IP address into a name that can be read by humans, such as www.geeksforgeeks.org (Manuel et al., 2018). When your computer connects to a website using one or more IP addresses, the DNS system responds (such as geeksforgeeks.org). A number of DNS servers resolve the domain name. According to (Barkri et al., 2021), (Freeman et al., 2020), DNS employs cache to operate effectively so that it may swiftly refer to DNS lookups already completed rather than repeatedly executing a DNS lookup. The domain name resolving procedures are carried out more quickly, thanks to DNS cache. Attackers take advantage of DNS flaws to take over the system and route visitors to a dangerous website. The major responsibility of a recursive server is to build and manage a sizable cache of DNS responses. The goal of cache poisoning is to tamper with the responses kept in the cache such that any subsequent lookups from other clients would receive the tampered-with response.

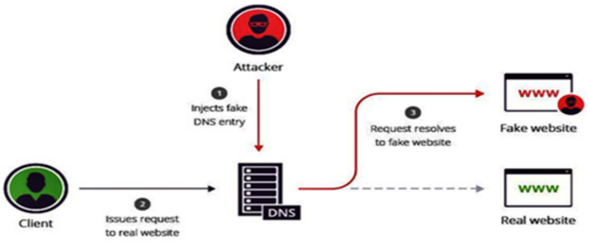


Figure 1: DNS Cache Poisoning

Experts have labeled it the most deadly DNS hack. User Datagram Protocol (UDP) on Port 53, which is often constantly open on systems, firewalls, and clients, is used to transmit DNS requests. The protocol does not include any security features. The fact that DNS is widely used and trusted is known to all cyber criminals. An information technology (IT) specialist would be aware that many businesses do not examine their DNS traffic for malicious activity because DNS is not intended for data transport. As a result, numerous DNS-based attacks against corporate networks can be effective.

This protocol is used by DNS tunneling attacks to bypass your firewall and send malicious traffic through. DNS can be used by an attacker to circumvent network security measures and carry out data exfiltration by utilizing malicious domains and DNS servers. Attackers can circumvent network administrators' rules by setting up covert channels using DNS to communicate secretly (Deoyer et al., 2020), (Frederick and Matthew 2016). This characteristic is exploited by attackers, who use DNS tunneling to create a command and control (C&C) channel for malware. While inbound DNS traffic can provide commands to the virus, outbound DNS traffic can exfiltrate your private information or provide answers

[1] African University of Science Administration and Commercial Studies Lome, Togo
[*] Corresponding author's e-mail: akhigbe-mudut@iaec-university.tg

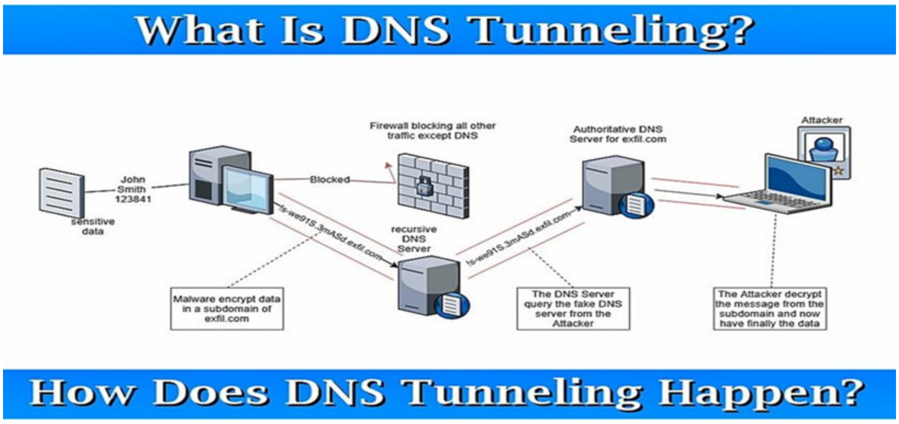

What Is DNS Tunneling?
John Smith 123841
sensitive data
Blocked
Firewall blocking all other traffic except DNS
recursive DNS Server
ls-we91S.3mASd.exfil.com
ls-we91S.3mASd.exfil.com
Authoritative DNS Server for exfil.com
Attacker
Malware encrypt data in a subdomain of exfil.com
The DNS Server query the fake DNS server from the Attacker
The Attacker decrypt the message from the subdomain and now have finally the data
How Does DNS Tunneling Happen?

Figure 2: Showing DNS Tunneling and The Probability of it happening

to the malware operator's questions. If the DNS server is compromised, hackers could access your database and steal information about your clients, including their social security numbers, names, addresses, phone numbers, mobile phone numbers, and emails (Mohammed 2020) (see figure 2).

What is a Honeypot?

A honeypot is a type of activity that enables you to compile all pertinent data regarding an attacker's movements (A. Paradise et al., 2017). A honeypot is a special security tool that is incorporated into your company's overall security plan. The goal is for all potential attackers to only interact with your honeypots and not with actual systems. If the attacker does not interact with the honeypot, they are useless. By storing data and applications that are exactly like those on genuine targets, honeypot traps malicious parties to attack these artificial networks, servers, or other devices (Leyi et al., 2018). The honeypot provides administrators with crucial details about the type of attacker, the activity he was attempting, and, in some cases, even the attacker's identity when an attacker falls victim to this trap. In order to gather information on new threats before they destroy your networks or important assets, they can be used as early warning systems, slowing down automated attacks and catching new exploits. The ability of honeypots to act as a genuine computer, a virtual machine, a whole (fake) network, or even an application is a benefit. Even without a computer, honeypots can be used. Examples include username and passwords, Excel spreadsheets, and credit card details (known as honey tokens). There are many different sizes and styles of honeypots (See figure 3). A real-time Windows or Linux operating system that may be deployed on a physical or virtual machine makes up the honey pot (Vishuevsky and Klyucharey 2020). The honeypot is set up to send logs to the data stream analyzer. In the Test Bed, a honey pot sensor is installed in the network's active directory. Since all systems connected to the network communicate with one another using active directory, convenient logging of all system activity is possible. The malicious system calls are forwarded to a read data stream analyzer, which is installed on a different machine and displays data streams from various sensors graphically according to their level of threat. A system log that is present in the data streams aids in identifying which harmful behavior is taking place on the system.

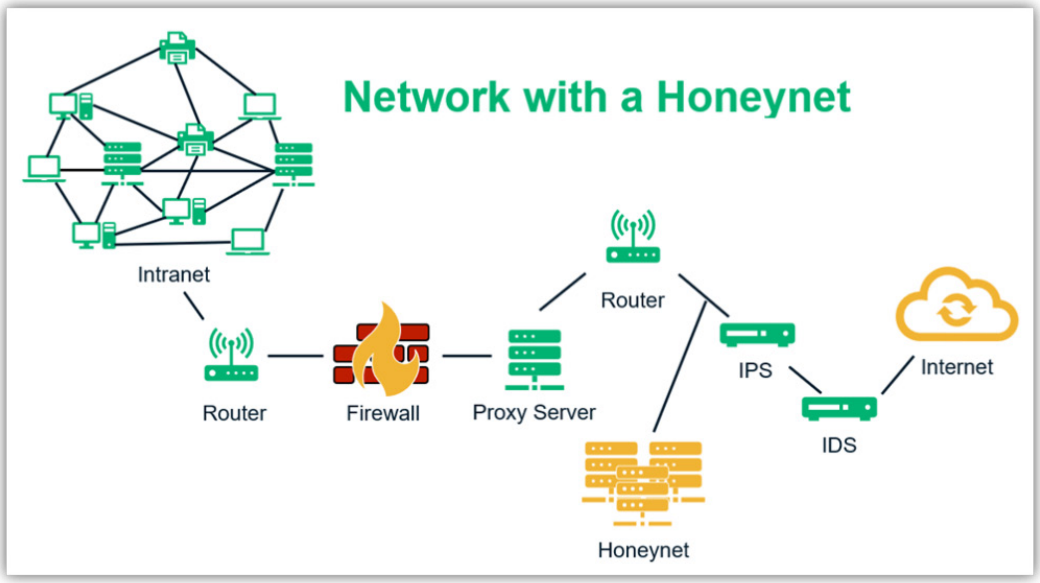


Figure 3: Honeypot deployment in a System

How do attackers tamper with DNS caches?

By posing as DNS name servers, sending a request to a DNS resolver, and then faking the response when the resolver queries a name server, attackers can poison DNS caches. Figure 4's illustration illustrates the procedure in great detail. This is allowed because DNS servers employ User Datagram Protocol (UDP) rather than Transmission Control Protocol (TCP) and because DNS information is not currently subject to verification (Faheem et al., 2017). Figure 4 in this article's illustration makes it very obvious how it occurs.

The primary perpetrator in this situation is UDP. Instead than using TCP, DNS requests and answers use UDP, which necessitates the execution of a "handshake" by both communicating parties in order to establish communication and confirm the identity of the devices. With UDP, it is impossible to determine whether a connection is active, whether the recipient is prepared to accept, or whether the sender is who they say they are. Because UDP is vulnerable to forging, a hacker may send a message over UDP and alter the header information to make it seem like a genuine server response. A DNS resolver accepts and caches false responses without question because there is no way to determine if the information is accurate and originated from a reliable source (Liang et al.,2020).

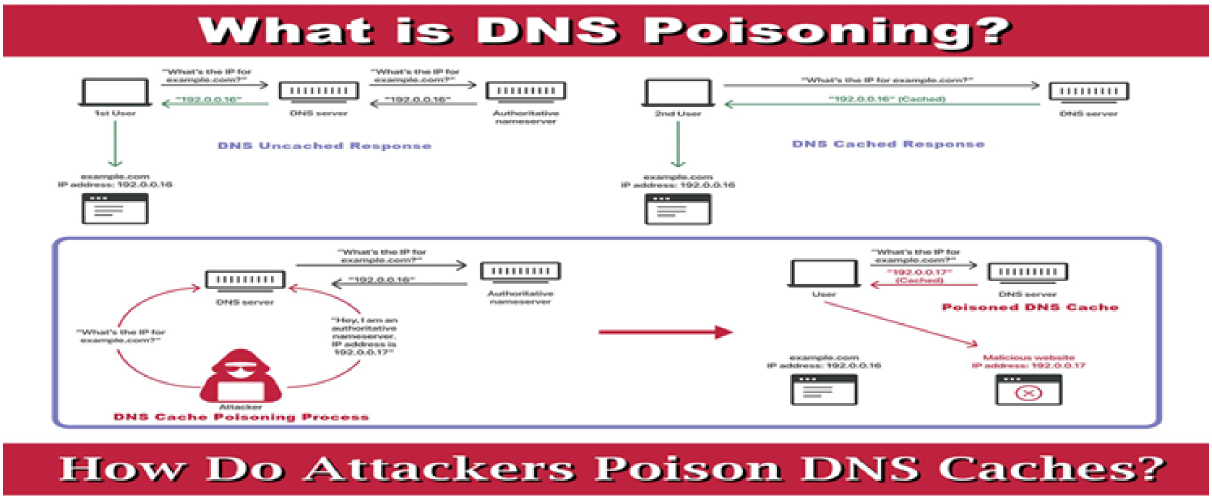


Figure 4: Shows the How Attackers Poison DNS Caches.

Statement of the Problem

According to some experts, organizations make investments of huge sum of dollars concerning firewalls, encryption, and other tightly closed access devices, yet it is all naught, considering that none regarding these solutions tackle the weakest link within the safety chain (Gammal et al., 2021). That sentence sums up the stress that network security professionals face nowadays. The weakest links in the security chain are those with authorization who work on secure networks. They may intentionally or accidentally pose a threat to the company, cause damage to networks, or steal sensitive data. These people are referred to as insiders, as stated by (Shin 2019). Both kinds of attackers are able to take advantage of the system's weaknesses. Therefore, it is crucial to create an efficient and effective security policy to get rid of such dangers.

The company's sensitive assets and confidential data are accessible to authorized users or employees, according to (Datta et al., 2021), there is always a chance that they would exploit this access for bad. Methods for detecting insider threats have a number of shortcomings. The limitations are related to personnel profile verification because the researchers' (Andrey and Pater 2020) developed insider threat detection technique can generate several false positive alarms while it is being used to detect threats. System lock down that was unnecessary for the organization was caused by false positive alerts (Cazenaye et al., 2020). Consequently, it is crucial to implement an effective security policy.

Second, authoritative name servers have the ability to accept dynamic updates, which essentially allows them to instantly produce new DNS records. This capability could be used by attackers to introduce unauthorized entries into the DNS zone. Insider Threat can be mitigated by using adequate security measures. Among the security measures are the implementation of an ethical policy for Internet users, the deployment of various spy product evaluation programs, anti-infection initiatives, firewalls, and a strong information security check and record administration.

Motivation

One of the most frequent reasons for cyberattacks, whether carried out by state actors, business rivals, organized crime, or even "hacktivists," is data theft (also known as data exfiltration). Due to two important causes, it is becoming more and more difficult to prevent data exfiltration. To start, during the past few years, cybercrime has changed from a personal to a collective act. As a result of this change, attackers (also known as hackers) now have more money, resources, and sophisticated data exfiltration techniques. Second, the current data infrastructure includes several mechanisms (shown in Fig. 5a) that were created for legal data exchange but can also be used for data exfiltration (Barkri et al., 2020).

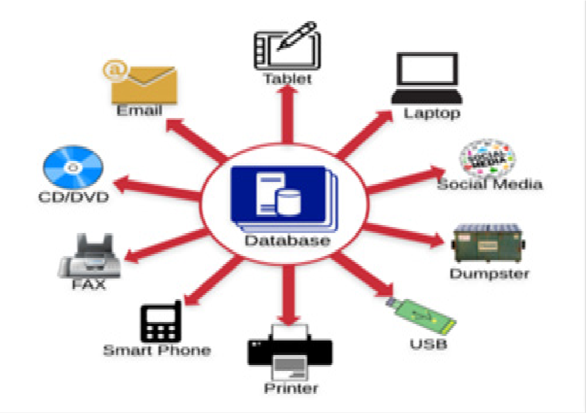


Figure. 5a: Some of the means of Data Exfiltration, **5b:** Data Exfiltration Scenario

Stage 1: An attacker searches via an organization's numerous channels for vulnerabilities that can be used to steal data.

Stage 2: After identifying the weakness, the attacker launches an attack to take advantage of it. Attacks could be physical or based on a network. In a network-based attack, an attacker may use a variety of tactics, including phishing emails, malware, or injections. In a physical attack, an attacker may employ a variety of tactics, such as copying data to a portable device or stealing printed materials.

Stage 3: The data exfiltration process starts once the attacker has accessed the private data, as shown in Fig. 5a.

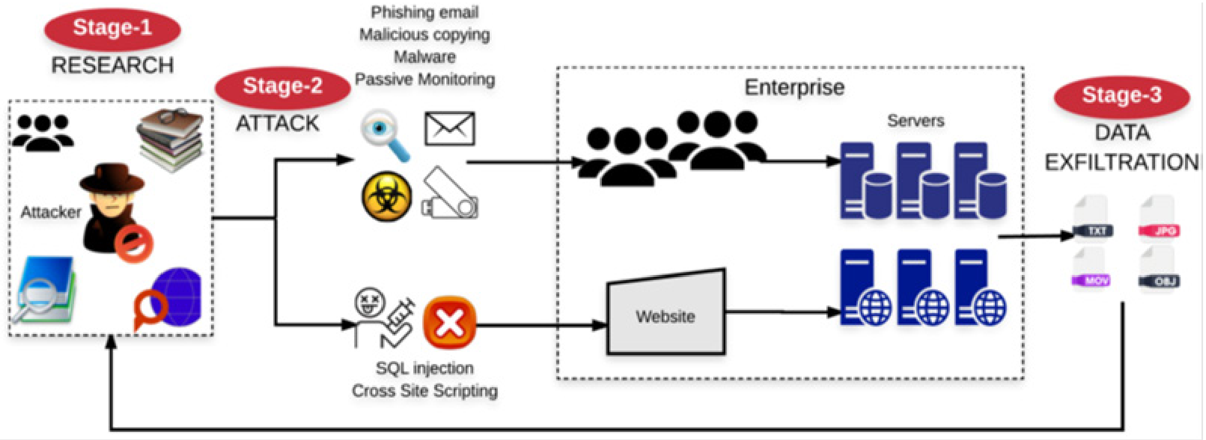


Figure 5b: shows one scenario of how these legitimate channels can be leveraged for data exfiltration.

LITERATURE REVIEW

Panangiotis et al., (2020) Built a trustworthy encryption method using fractional-order chaotic systems. The speech encryption system came with the substitution boxes and other building elements needed to generate the encryption. Sensitivity, statistical analysis, and a number of other criteria were considered in a number of security studies. According to Foneseca et al., (2020), Advanced Encryption Standard (AES) methods and hashing operations like the Message Digest Algorithm (MD5) can be utilized to protect audio file steganography. The AES algorithm was used to encrypt the data and MD5 was used to scramble passwords. Moreover, the MP3 file was encoded. On the other end, decoding is performed by removing the private message's sensitive information and decrypting it to recover the original data. The limitation of this technique is that it can only be used on mp3 files with a homogeneous frame.

The steganographic technique utilized a Cuckoo search accompanied by other optimization procedures, in contrast to Razzaq and Ahmad (2015) which utilized a Cuckoo search with other optimization strategies to scramble mp3 information inside a picture. The result of their strategy was greatly noteworthy since it had great precision. In spite of the broad utilize of pictures, the steganographic strategy was as it were appropriate to sound recordings. There have been at slightest two complementary strategies for analyzing information from cyber-attacks that have been captured by honeypots. One strategy for visualizing cyber-attack information is to utilize neural projection strategies to show the ports found in honeypot information (Souravlas and Roumeliotis 2015). This method's downside is that it can as it were be connected to mp3 records with homogeneous outlines.

The steganographic method employed a Cuckoo search followed by other optimization strategies, in contrast to (Qin et al., 2015) which used a Cuckoo search followed by other optimization techniques to encrypt mp3 data within an image. The outcome of their method was extremely impressive because it overall had good accuracy. Despite the extensive use of images, the steganographic method was only applicable to audio recordings. There have been at least two complementary methods for analyzing data from cyber-attacks that have been captured by honeypots. One method for visualizing cyber-attack data is to use neural projection techniques to display the ports found in honeypot data (Dalla and Dheida 2020). On the other hand, statistical analysis is a technique that is frequently used. With the ultimate goal of understanding and predicting cyberbullying, our study of predicting cyberbullying (in terms of attack rate) should be particularly significant. Regarding the use of honey jars as a form of self-defense, we point out that honeypots have been employed to assist in the detection of numerous attacks, such as DoS (denial-of-service) (Alhathaly et al., 2020), worms (Dlamini et al., 2020), (Davison et al., 2020), botnets (white et al., 2020), (Wisniewski et al., 2020), Internet messaging threats (Xiaoyang et al., 2019), the creation of attack signatures (Gokhale et al., 2020), (Zareef Mohammad 2020), and targeted targeting (Consuelo 2020). Although these lessons are significant, the current paper's main focus is not on them (Knuk et al. 2018). A honeypot was proposed as a security architecture called Japonica that can be used to detect and stop unidentified aggressive attempts. Komenda et al., (2020) describe the model framework put forth by the Colored Petri Nets (CPN) to choose the structure of messages sent between organizations that are similar. The outcomes demonstrated the framework's potential for success. Other activities connected to Petrol Nets exist than those mentioned above. A software system model known as an Intelligent Petri Net (PN) was presented (Shin 2019), in which the operational time domain and system behavior may be characterized by fusing conventional Petri nets with ambiguous rules. I-PN includes adaptability in its design. With Siphon and proteins involved in the targeted detection of medicinal medicines that contain numerous components in showing pathways, Petri nets were employed in Barylska et al., (2020) for a physiochemical model. A similar control by Balogh and kucharik (2019) has been described using translations of Petri nets. During the polynomial era, the system prototyping method was put out. In terms of network security, the aforementioned tasks are nevertheless tightly tied to login access, web service firewall, spoofing Address resolution Protocol (ARP), and program structure. In this article, we emphasize the use of honey pots and Petri nets (Marcin et al., 2021). Only in Dimitrio et al., (2019), as opposed to the Stochastic Petri Nets, a honeypot was utilized as part of the suggested structure (SPN). As a result, to our knowledge, no work has been done on both honeypot and SPN. This study suggests such a strategy to evaluate the honeypot's effectiveness.

Wisniewski et al., (2020) too bargain with information burglary on remote systems; in spite of the fact that their center is on versatile advertisement systems sent for military operations. They utilize the integration strategy to discover the confounding, utilizing IP and data of the transport theme as highlights. These strategies center on observing information that crosses the organizations organize boundary and avoiding it from assault like SQL infusion. Their beginning approval of this strategy in recreation works, but the adaptability prerequisites of the wiring organize don't show up to be fully spoken to within the test environment. Recognizing unusual activity on a quickly changing ad-hoc arrange would appear to be a troublesome issue for any dividing framework.

Wisniewski et al., (2019) introduced a strategy for monitoring network log files using data mining methods to identify password guessing attempts and DoS attacks. The data is recorded and kept in a file once the log file has been processed (Xiao and Li (2021). The clustering method is created and applied to find connections that happen repeatedly. The connection that frequently appears denotes irregular behavior. The key flaw in the suggested approach is its inability to identify attacks as they happen, which makes the system less effective. Due to the lack of clarity surrounding a few related factors, including acquisition rate and performance implications, the authors do not offer job evaluation tests.

METHODOLOGY

Sensor Data Parser

This module converts the incoming honeypot sensor data streams into readable format. Strings represent the data produced by the honeypot sensor. The module separates the strings into understandable forms for the source IP, destination IP, event name, username, Date, and Time, among many other significant processes. Data is sent to the threat analyst to identify potential insider threats after gathering all pertinent event data. On the same server, the threat analyzer is installed as a distinct module.

Threat Assessment

Three stages of data analysis are used by Threat Analyst. Each step is built to weed out any false alarms. Only when it detects event activity in any of the level details listed in the algorithm below does it issue a notification:

Algorithm 1: False Positive Insider Threat Detection

```
Step 1: There won't be an alarm if the user event function is normal.
Step 2: Alternatively, look at a software and event-based program.
Step 3: There is no alarm when user behaviour is linked to software and app. events.
Step 4: Continue to monitor system hardware alterations,
Step 5: If the user has enabled the hardware changes, there will be no alarm.
Step 7: if the user edits the files according to each user's rights then No alarm.
Step 8: Alternatively raise the alarm.Step 6: Verify user rights once more.
```

Figure 6's algorithm is easy to apply and effective enough to catch any false alarms caused by a user's system function. The three levels of user system function are divided, and each level first determines whether the event is a software-produced system or a system-produced system, as previously identified in the profile training part. If it is not, the event is then ignored. The next step is to determine if any hardware modifications to the system that caused the suspicious events have occurred. If so, it is important to determine whether these modifications constitute routine events brought by human behavior. Assesses modifications to user-generated data at the end. If user-adjusted data complies with the specified user rights, those events are disregarded. To identify insider threats, any suspicious events produced by random sequences, read, write, network penetration, data transfer, modifications to computer hardware, and changes to

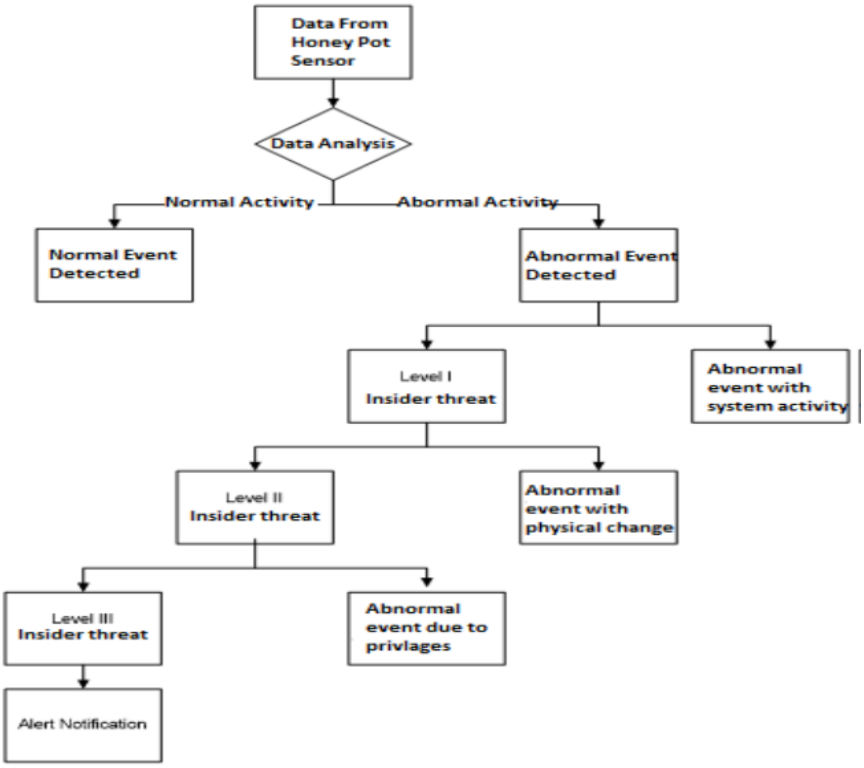


Figure 6: Three level classification of threat analyzer algorithm

software are immediately identified. Figure 7 depicts the suggested threat analysis algorithm's three-stage partition of its sequence. Only system administrators have access to the web-based, user-friendly GUI where the discovered insider threat is shown (shown in Figure 7). The figure shows that the three concepts that make up SOAP specifications are protocol concepts, encapsulation concepts, and network concepts. The use of online services has led to the structure of the data transmission protocol. TCP Dump is a program with a visible command line for network data packet analysis. Enables viewing of TCP/IP and other export packets by the user. In this illustration, the website's query is examined using the optimizer components. Every query must be thought of at least once; hence it is utilized in this context for decision-making.

RESULT AND DISCUSSION

Result

The honeypot contains a window operating system that runs on a virtual machine. The honeypot is designed to transfer logs to the data distribution analysis. In Test Bed, the honeypot sensor is installed in the active network directory where all systems are connected to the network and use the active communication interface of their system. With network, all activity in the system can be easily accessed. Sensor Data Parser is a module for analyzing incoming data transmission from honeypot sensors in readable format. The honeypot generated data is divided by the module into IP source threads, local IP, event name, username, date, time and many other important features represented in a readable manner. After collecting all relevant event data, the data is transferred to the threat analyst to identify potential Insider Threat. The threat analyzer is a separate module and is installed on the same server. At the beginning of the trial, participants were directed to create a normal behavioral profile. Prior to the creation of standard user profiles, the researcher created 40 MB random references within user programs, and created 32 references in the system. After creating a standard profile, each participant created a different version of the modified, deleted, and modified files on drivers that are often inaccessible. Unusual process work can be a prelude to the fact that malicious application, malicious code, or any other malicious program is present in the program. As established earlier, an organization's internal attack can be exceedingly hazardous and dangerous if left unchecked. The researcher has created a different experiment that is incredibly useful for identifying odd process behaviors like deadlock. The profile part is followed by the subsequent examinations.

Deadlock

A scenario known as a deadlock includes the interaction of multiple resources and processes (Yang et al., 2018). If there are two persons on the stairs, we can see a deadlock as that circumstance (figure 8). While the other descends the stairs, one does so. There is enough room on the stairs for one person at a time. One has to take a step back while the others proceed and use the stairs as a result. After that individual is done, another person may utilize those steps. However, no one is prepared to stay and wait for someone else to take a step back in this situation. They are all unable to use the stairs. The stairs

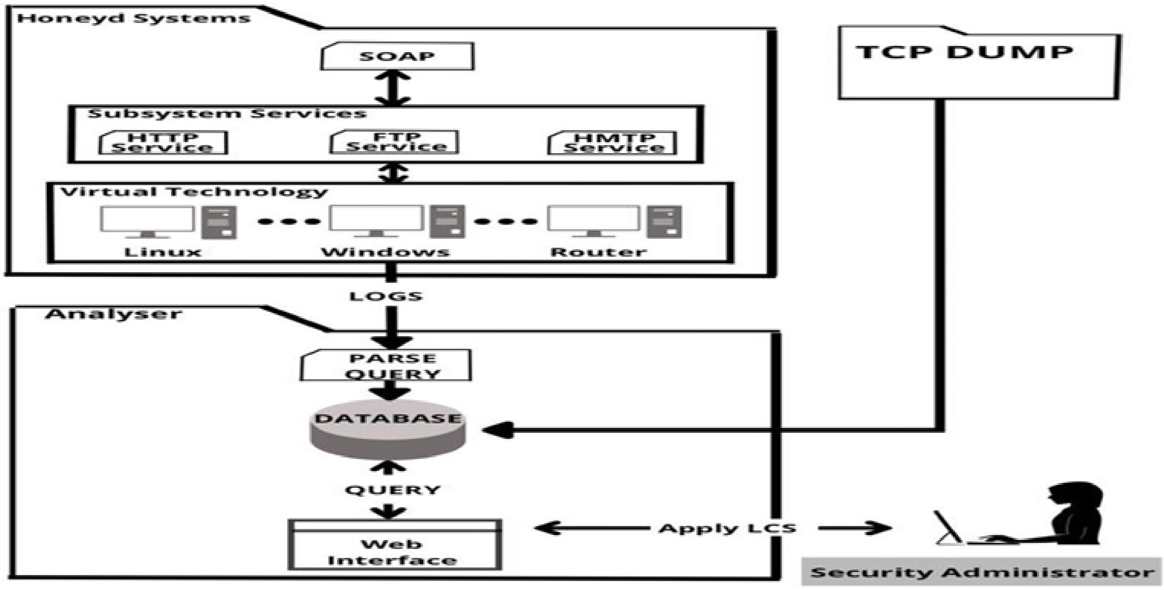


Figure 7: User Friendly Web Based GUI threat analyzer algorithm

are the source, and the individuals present are the process. Deadlock occurs when a process requests an application hosted by another process that needs another program to proceed but is halted by the first process (Routian et al., 2020), (Hou and Barkaoui (2017).

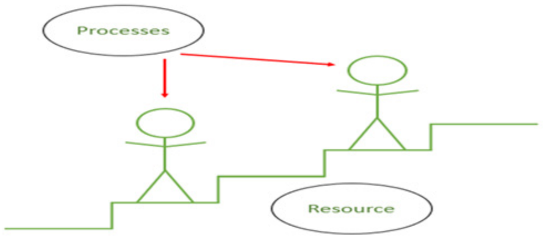


Figure 8: Deadlock Condition

The simultaneous holding of the four instances listed below (Required Conditions) can result in deadlock.

1. Multiple resources cannot be shared in a shared release (Only one process can be used at a time)
2. Hold and Wait: The process holds one or more applications while it awaits resources.
3. No Release: Unless the system uninstalls the software, the app cannot be downloaded from the system.
4. Waiting for a round: A number of processes are awaiting a round.

Safety Algorithm / Deadlock Detection Algorithm: The algorithm for detecting safe state condition in a system can be defined as follows:

Algorithm steps:

```
Assume Task and Finish to be vectors of length 'm' and 'n' respectively.
Get Started Task = Available for i=0,1,2,...,n-1
 if Request , then Finish [i] = true; otherwise, Finish [i] = false.
   Get an index 'i' such that both:
(a).Finish[i]=false
(b)Request 'i'<=Task
   If not then go to step 4.
   Task = Task + Allocation
   Finish [i] = true
   Go to step 2.
   If Finish [i ] == is false for some 'I' ,it means the system is in a deadlock state.,
 In addition, if Finish [i] == false, the process is deadlocked.
```

Petri Nets Definitions

Three different item kinds make up a two-dimensional graph called a Petri net. The locations, transitions, and directed arcs are as follows (Yi-Nan et al., 2022). Directed arcs link locations to transitions or from transitions to locations. The Petri net can be represented by a transition together with an input and output location in its most basic form. A variety of modeling system components can be represented using this fundamental net. Tokens are a dated Petri nets notion for locations and transitions. If a token is present or absent, it may be possible to determine if the status corresponding to that location is true or false.

For Example

A Petri Net is formally defined as a 5- tuple $N=(P,T,I,O,M)$ where:

1. $P=\{p_1,p_2,...,p_n\}$ is a finite set of places;
2. $T=\{t_1,t_2,...,t_n\}$ is a finite set of transactions,
3. $I:P\times T\rightarrow N$ is an input function that defines directed arcs from places to transitions, where N is a set of nonnegative integers: $P\cup T\neq\phi$...and...$P\cap T=\phi$
4. $O:T\times P\rightarrow N$ is an output function defined by directed arcs transitions and;
5. M_o:P→N is the initial marking.

Tagging on the Petriet net is the assignment of tokens in the Petri net area. Tokens are located in the places of Petri's net (Chakraborty 2019). The number and status of tokens may change during the creation of the Petri net. Tokens are used to describe the processing of Petri net. The Petri net graph is the structure of the Petri net as a directed multigraph for bipartite. Consistent with the definition of Petri nets, the Petri net graph has two types of nodes. The circle represents the location (Place); bar or box represents transition. Targeted (directed) arcs (arrows) link places and transitions, with other arcs directed from places to transitions and other arcs directed from transition to locations. An arc directed from a location to the transition defines the input location of identified as. The arc directed from the transition to the place defines as an output place of which is defined as or then there are k-directed (parallel) arcs that link the location to the transition (or link the transition to the location) represented by a single pointed arc labeled by its quality, or weighted K. A circle contains a dot representing a place containing a token (Su and Qui 2019).

A Simple Petri Net Illustration

Figure 9 shows a simple Petri net. In this Petri net, we have

$P=\{p_1,p_2,p_3,p_4\};$

$T=\{t_1,t_2,t_3\}$

$I\{t_i,p_i\}=2,I(t_2,p_2)=0\forall i=1,2,3,4;$

$I(t_2,p_2)=1,I(t_2,p_i)=0\forall i=1,3,4;$

$O(t_1,p_2)=2,O(t_1,p_2)=1,O(t_1,p_i)=0\forall i=1,4;$

$O(t_2,p_4)=1,O(t_2,p_i)=0\forall 1=1,2,3;$

$O(t_3,p_4)=1,O(t_3,p_i)=0\forall i=1,2,3;$

Stochastic Petri net

A Stochastic Petri Net (SPN) [39] is defined by 6 − tuple (P,T,F,W,M,Ω) where (P,T,F,W,M) are similar as defined in the definition of standard petri net and Ω represents the function Ω:T→R which assigns rate to the transition t∈T according to the negative exponential distribution function. The emergence of Stochastic Petri Net as defined by Continuous Time Markov Chain (CTMC) and a state of a CTMC represents a single Petri Net tag. In other words, CTMC represents the Petri Net accessibility graph (Su and Qui 2019). The example in Figure 9, shows the definition of Stochastic Petri Net. There are a few behaviors of Petri Nets (Xia and Li 2021), (Su and Qui 2019) and (Yang et al., 2017) and some of them are described below:

- **Reachability:** This property is used to study dynamic properties of the system. A marking M_k is reachable from an initial marking M_0 if there exists a firing sequence from M_0 to M_k.
- **Liveness:** A live Petri Net is a deadlock frees Petri Net and from any marking, there exists a firing sequence which contain all transitions.
- **Reversibility:** This property ensures that there will always be a way back to the initial marking Mo from all reachable markings commencing from M_o.

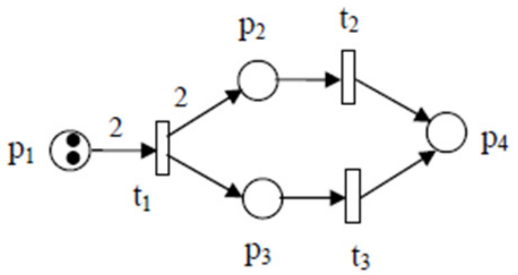


Figure 9: A simple Petri net.

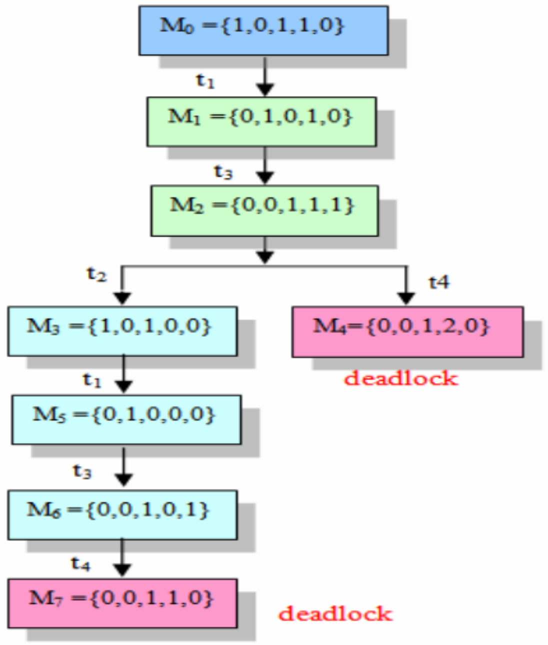
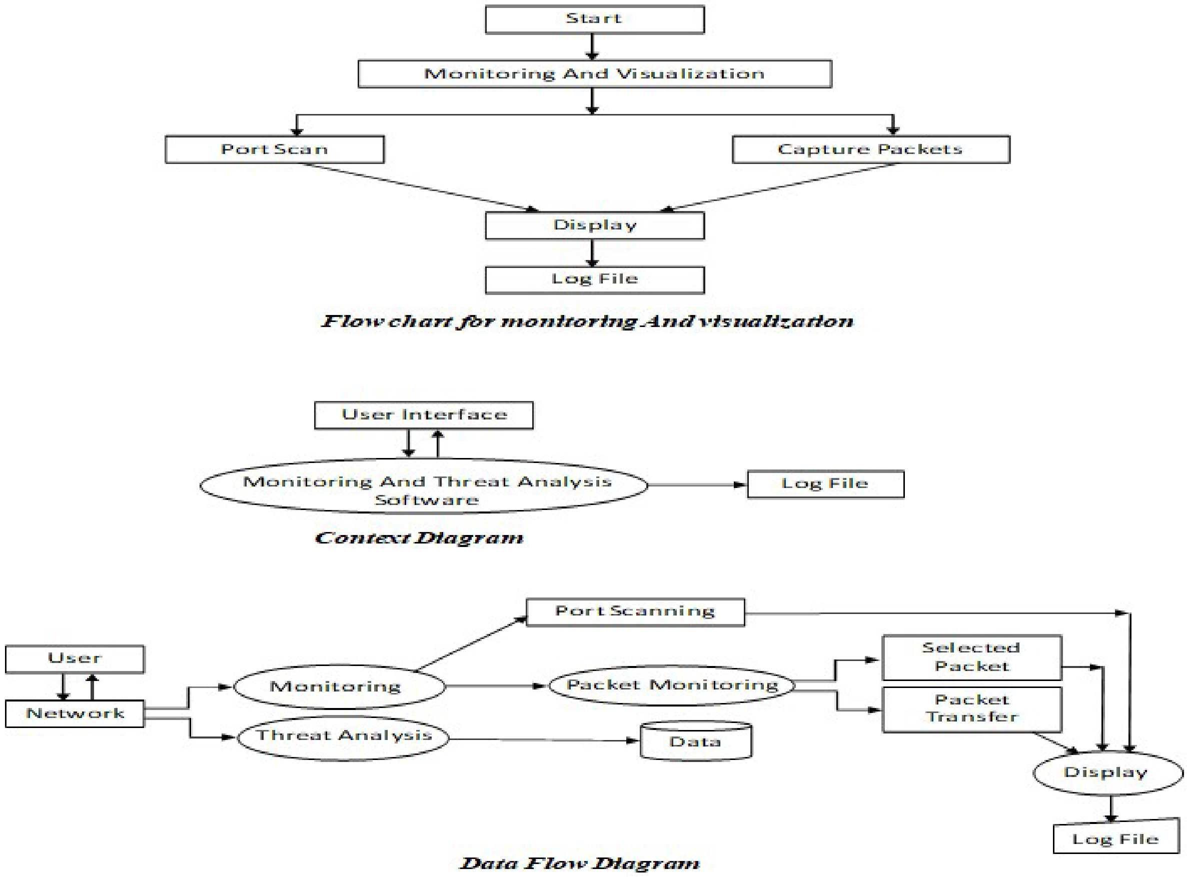


Figure 10: The Reachability Flowchart of Figure 9

A marked PN(ψ,m_0) is said to be (marking) k-bounded iff each of its place is k-bounded. A 1-bounded net is called safe. A marked PN(ψ.m_0) is bounded if there exists (k∈N) such that (ψ,m_0) is k-bounded. A net ψ is structurally bounded iff M, the marked PN(ψ,m_0)arek-bounded for some k∈N.

Performance Evaluation

The theoretical aspect of Petri nets allows for accurate modeling and behavioral analysis, while graphical representation of Petri networks enables the realization of structural changes in the system (Yi-Nan et al., 2022). This combination is the main reason for the selection of performance tests for this study. Simulation contains applications and tasks with the required skill, depending on the package routes. Passive monitoring which is the subject of performance testing is a way to track the performance and behavior of a broadcast package by measuring user traffic without creating new traffic or modifying existing traffic. This is used by integrating additional intelligence into network devices so that they can detect blocked processes, record features and the number of packets flowing through them. The IP header of the passive monitoring package contains the destination address of the receiving node and the source address of the sending



Flow chart for monitoring And visualization

Context Diagram

Data Flow Diagram

Figure 11: Flowchart for monitoring and Visualization

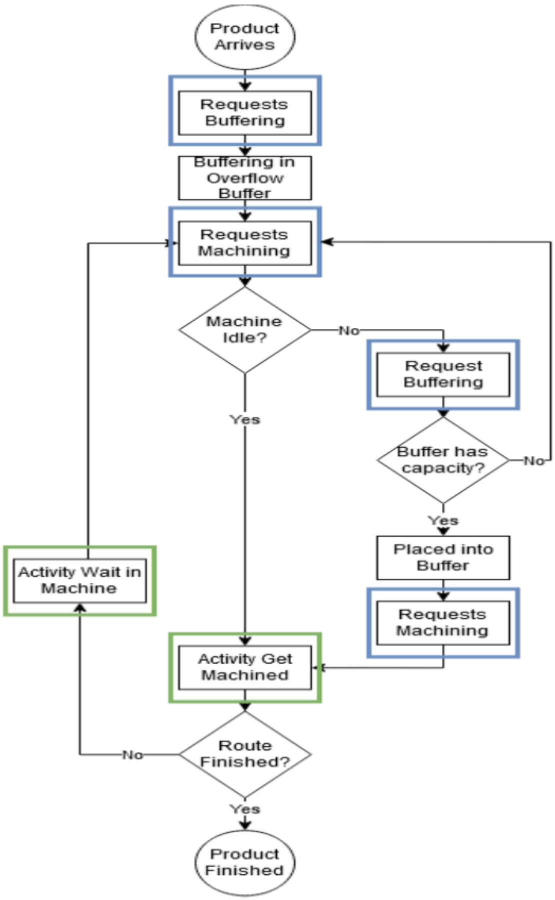


Figure 12: Flowchart of Requests and Activities

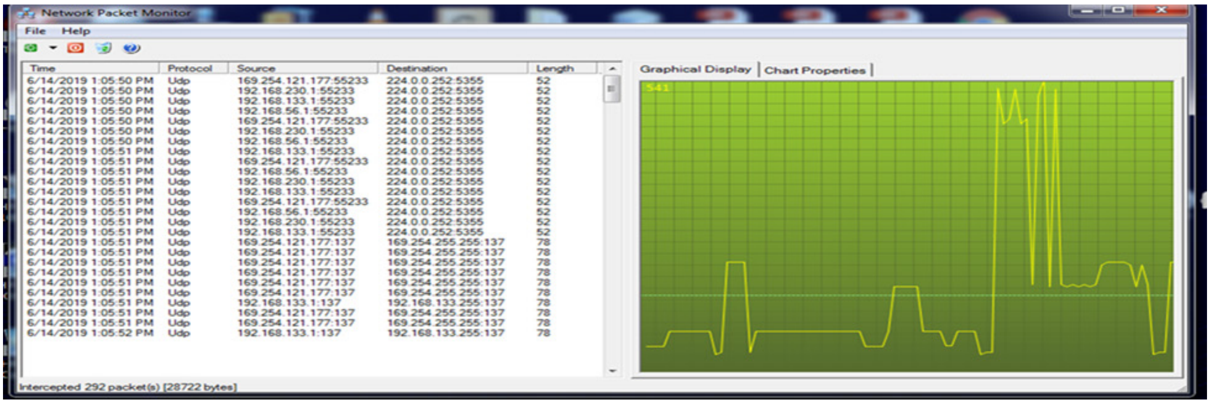
node. The reception area sets the total (aggregate) total number of packets received and the current time in the corresponding header fields. The Augmented counting Algorithm (fig. 12) looks at the sender on the receiving package and seeks to identify any blocking. As soon as the work starts, the machine is set (not idle) and stores information. To better understand these behaviors, figure 13 shows a simplified flowchart from process perspective to focus on tasks and applications.

DISCUSSION

This paper shows the effective siphon structure for PN analysis. This tool is very useful for finding a siphon and setting up an easily accessible tree, deadlock detection, and liveness of petri nets. The Petri net model as a tool that helps us to take a deeper look at ethical and structural investigations. Table 1 shows the image representation of that collection. On the left side, from top to bottom is the main simulation time and at the top is the transfer protocol used (UDP), Source (IP address), Location (Internet address) and data size. Simulation as mentioned earlier includes Applications and Tasks. This behavior is used to identify the state of the system when the process is idle, separating the blocked machine. However another required request machine is filed and operations are performed successfully and this may result in increased packet transfer. This is clearly shown in the table on 06/14/2019 at 1.05.50 pm and 1.05.51 pm with data lengths of 52 packets and 78 packets respectively.

The Augmented counting Algorithm of (figure 12) looks at the sender on the receiving package and seeks to identify any blocking. As soon as the work starts, the machine is set (not idle) and stores information. To better understand these behaviors, figure 13 shows a simplified flowchart from process perspective to focus on tasks and applications. The theoretical aspect of Petri nets allows for accurate modeling and behavioral analysis, while graphical representation of Petri networks enables the realization of structural changes in the system (Yi-Nan et al., 2022).

The outlined steps conclude the basic function of the structures that PN is a powerful and widely used simulation analysis strategy in which deadlock control mechanisms will be developed.



Time	Protocol	Source	Destination	Length
6/14/2019 1:05:50 PM	Udp	169.254.121.177:55233	224.0.0.252:5355	52
6/14/2019 1:05:50 PM	Udp	192.168.230.1:55233	224.0.0.252:5355	52
6/14/2019 1:05:50 PM	Udp	192.168.133.1:55233	224.0.0.252:5355	52
6/14/2019 1:05:50 PM	Udp	192.168.56.1:55233	224.0.0.252:5355	52
6/14/2019 1:05:50 PM	Udp	169.254.121.177:55233	224.0.0.252:5355	52
6/14/2019 1:05:50 PM	Udp	192.168.230.1:55233	224.0.0.252:5355	52
6/14/2019 1:05:50 PM	Udp	192.168.56.1:55233	224.0.0.252:5355	52
6/14/2019 1:05:51 PM	Udp	192.168.133.1:55233	224.0.0.252:5355	52
6/14/2019 1:05:51 PM	Udp	169.254.121.177:55233	224.0.0.252:5355	52
6/14/2019 1:05:51 PM	Udp	192.168.56.1:55233	224.0.0.252:5355	52
6/14/2019 1:05:51 PM	Udp	192.168.230.1:55233	224.0.0.252:5355	52
6/14/2019 1:05:51 PM	Udp	192.168.133.1:55233	224.0.0.252:5355	52
6/14/2019 1:05:51 PM	Udp	169.254.121.177:55233	224.0.0.252:5355	52
6/14/2019 1:05:51 PM	Udp	192.168.56.1:55233	224.0.0.252:5355	52
6/14/2019 1:05:51 PM	Udp	192.168.230.1:55233	224.0.0.252:5355	52
6/14/2019 1:05:51 PM	Udp	192.168.133.1:55233	224.0.0.252:5355	52
6/14/2019 1:05:51 PM	Udp	169.254.121.177:137	169.254.255.255:137	78
6/14/2019 1:05:51 PM	Udp	169.254.121.177:137	169.254.255.255:137	78
6/14/2019 1:05:51 PM	Udp	169.254.121.177:137	169.254.255.255:137	78
6/14/2019 1:05:51 PM	Udp	169.254.121.177:137	169.254.255.255:137	78
6/14/2019 1:05:51 PM	Udp	169.254.121.177:137	169.254.255.255:137	78
6/14/2019 1:05:51 PM	Udp	169.254.121.177:137	169.254.255.255:137	78
6/14/2019 1:05:51 PM	Udp	192.168.133.1:137	192.168.133.255:137	78
6/14/2019 1:05:51 PM	Udp	169.254.121.177:137	169.254.255.255:137	78
6/14/2019 1:05:51 PM	Udp	169.254.121.177:137	169.254.255.255:137	78
6/14/2019 1:05:52 PM	Udp	192.168.133.1:137	192.168.133.255:137	78

Intercepted 292 packet(s) [28722 bytes]

Figure 13: Shows a Graphical Representation

CONCLUSIONS

The petri nets method described here has allowed us to make system modeling easier and faster compared to other analytical methods. We have concluded from this study that honeypot techniques are very effective in reducing the risk of deadlock and, in fact, have a significant effect on controlling systems against hackers. What has been presented in this article is a brief review of rich information in the field of petri nets. It is not possible to discuss all aspects of the field on a single page. Therefore, emphasis is placed on the area known as transformation zones / programs, as well as the theory of petri nets used. Stochastic petri nets, high quality nets and their application models deserve more space, as there is a growing interest in these areas. This field is new and there is still a lot of work to be done. We hope that this paper will help to mimic further research and development in the emerging field of petri nets.

Acknowledgement

This paper and the research behind it would not have been possible without the exceptional support of my supervisor Professor Akinwale A.T. of Federal University of Agriculture Abeokuta, Ogun State Nigeria. His enthusiasm, knowledge and exacting attention to detail have been an inspiration and kept my work on track.

Most importantly, I wish to thank my loving and supportive wife, and my Four wonderful children (Pamela, Ohi baby, Ebehitale and Ehisoyayan) who provide unending inspiration. They are the ultimate role models.

REFERENCES

Andrey Vishnevsky. Peter Klyucharev (2020). A Survey Game- Theoretic Approaches to Modeling Honeypots. Conference: Secure Information Technologies 2017 (BIT 2017): At Mouscow Russia.

Bakri, A., Alkbir, M. F. M., Awang, N., Januddi, F., Ismail, M. A., Ahmad, A. N. A., & Zakaria, I. H. (2021). Addressing the issues of maintenance management in SMEs: towards sustainable and lean maintenance approach. *Emerging Science Journal, 5*(3), 367-379.

Barylska, K., Koutny, M., Mikulski, Ł., & Piątkowski, M. (2018). Reversible computation vs. reversibility in Petri nets. Science of Computer Programming, 151, 48-60.

B. Camiña, R. Monroy, L. A. Trejo, and M. A. Medina-Pérez. (2016). "Temporal and Spatial Locality: an Abstraction for Masquerade Detection. *IEEE Transactions on Information Forensics and Security 11*(9), 2036–2051.

Balogh, Z.; Kuchárik, M.(2019). "Predicting student grades based on their usage of LMS moodle using Petri inets." *Appl. Sci., 9*, 4211.

Chakraborty, S. (2019). "Analyzing Peer Specific Power Saving in IEEE802.11s Through Queuing petri nets: Some Insights and Future Research Directions". *IEEE Transactions on Wireless Communications, 15*(5), 3746–3754. https://ieeexplore.ieee.org/document/7404028.

Consuelo, N. (2020). "Advanced Design for Manufacturing of Integrated Sustainability "Off-Shore" and "Off-Site" Prototype - MVP "S2_HOME.". *Civil Engineering Journal, 6*(9), 1752–1764.

Davison, P., Cameron, B., & Crawley, E. F., et al. (2020). Technology Portfolio Planning by Weighted Graph Analysis of System Architectures. *Systems Engineering, 18*(1), 45–58. https://doi.org/10.1002/sys.21287.

Dwyer, M., Cameron, B., & Szajnfarber, Z., et al. (2020). A framework for Studying Cost Growth on Complex Acquisition programs. *Systems Engineering, 18*(6), 568–583. https://doi.org/10.1002/sys.21328.

D.Dalla and J. Dheiba (2020). Exploration of Various

Attacks and Security Measures related to the Internet of Things *International Journal of recent Technology and Engineering, 9*(2), 175-184

Dlamini, M.T., Venter, H.S., eloff, J.H., Eloff, M. (2020, September 8– October1). *An Information Behaviour Lens.* In proceeding of the Information Behavior Conference, Pretoria South Africa.

Datta D., Garg L., Srinvasan K., Inoue A., Reddy G.T., Reddy ,M.P.K., Ramesh K., Nasser N. (2021). Efficient Sound and Data Steganography Based secure Authentications System. *Computers, Materials, and Continua, 67*(1), 723-751. https://doi.org/10.32604/cmc.2021.014802.

Ellard D., Jones C., Manfredi V., Strayer W.T., Tapa B., Van welle M.and Jackson A. (2015). *A Rebound: Decoy Routing on Symmetric Routes Via Error Messages*". In IEEE 40th Conference on Local Computer Networks (LCN) 2015, (pp. 91-99).

Faheem Ullah, Matthew Edwards, Rajiv Ramdhany, Awais Rashid (2017). Data Exfiltration: A Review of External Attack Vectors and Counter Measures. *International Journal of Networks and Computer Applications, 101*(2). https://doi.org/10.1016/j.inea.2017.10.016

Frederick Weigang Pan and Matthew Caesar (2016). Salmon: Robust Proxy Distribution for Censorship Circumvention. Proceedings on Privacy Enhancing Technologies. 2016(4), 4-20. https://doi.org/10.1515/popsets-2016-0026.

Freeman, R.E., Phillips, R. and Sisodia, R. (2020), Tensions in Stakeholder Theory, *Business and Society, 59*(2), 213-231.

Gammal E.I Selim; Ezz El-Din Hemdan; Ahmed M. Shehatta; Nawal A. El-Fishawy (2021). An Efficient Machine Learning Model for Malicious Activities Recognition in Water-Based Industrial Internet of Things. *Journal Security and Privacy, 4*(3), 1-14. https://doi.org/10.1002/spy2.154.

Jan Komenda; Aiwen Lai; Jose Godoy-Soto; Sebastian Lahaye; Jean-Loius Boimond (2020). Modeling of Safe Time Petric Nets by Internal weighted Automata. *IFAC paper online, 53* (4), 187-192. https://doi.org/10.1016/j.ifaco.2021.04.018.

Konuk, F.A. (2018). "Price fairness, satisfaction, and trust as antecedents of purchase intentions towards organic food, *Journal of Consumer Behavior, 17*(2),141-148

K.A. Shin (2019): Universal Forgery Attacks on remote Authentication Schemes for Wireless Body Area Networks Based on Internet of Things. *IEEE Internet of Things Journal, 6*(5), 9211-9212.

Liang, X.; Zhang, S.; Liu, Y.; Ma, Y.(2020). Information Propagation Formalized Representation of Micro-blog Network Based on Petri Nets. *Sci. Rep., 2020*(10), 1–20.

Leyi Shi; Yang Li; Haijie Feng (2018). Performance Analysis of Honeypot with Petri Nets. *Information Theory and Methodology, 9*(10), 245. https://doi.org/10.3390/info9100245.

Lama Alhathally; Mohammed A. Alzain; Jchad Al-Amri; Mohammed Baz; Mehedi Masud (2020). Cyber Security Attacks: Exploiting Weaknesses. *International Journal of Recent Technology and Engineering (IJRTE), 8*(5), 906-913.

Marcin Wojnakowski; Remiguisz Wisniewski; Grzegorz Bayzydio and Mateusz Poplawskwi (2021): " Analysis of Safeness in a Petri Nets Based Specification of the Control Part of Cyber-physical systems. *International Journal of Applied Mathematics and Computer Science, 31*(4), 647-657. https://doi.org/10.34768/amcs-2021-0045.

Mohammed Y.F (2020). Network – Based detection and prevention System Against DNS-Based Attacks. https://doi.org/scholarworks.uark.etd/3970..

Manuel Cheminod; Luca Durante; Lucia Seno; Adriano Valenzano (2018). Performance Evaluation and Modelling of an Industrial Applications Layer Firewall. *IEEE Transactions on Industrial Informatics, 14*(5), 2159-2170. https://doi.org/10.1109/TII.2018.2802903.

Paradise, A., Shabtai, A., Puzis, R., Elyashar, A., Elovici, Y., Roshandel, M., & Peylo, C. (2017). Creation and management of social network honeypots for detecting targeted cyber attacks. *IEEE transactions on computational social systems, 4*(3), 65-79.

Panagiotis Radoglou-Gammaliks; Panagiotis Sariagiannidis, Eider Iturbe; Erkuden Rios, et al., (2021). Spear Siem: A security Information and Event Management System for the Smart Grid. *Computer Networks, 193*, 1-26. https://doi.org/10.1016/j.comnet.2021.108008.

P. Cazenave; M. Khifi-Bouassida; A. Togueyeni (2020). S3PMR Deadlock and Control with Partial Controllability and Observability. *Journal of International Federation of Automatic Control. 15th IFAC Workshop on Discrete Event Systems WOOES 2020-Rio de janeiro, Brazil, 53*(4), 173-179. https://doi.org/10.1016/j.ifaco.2021.04.017.

Pau Fonseca Casas; Daniel Lijia Hu: Antoni Guasch I Petit and Jaume Figueras Jove (2020). Simplifying The Verification of Simulation Models through Petri Nets to Flexsim Mapping, *Applied sciences, 10*(4), 1395. https://doi.org/10.3390/app10041395.

Qin M, Li ZW and Al-Ahmari AM (2015). Elementary-Siphon Based Control Policy for Flexible Manufacturing Systems with Partial Observability and Controllability of Transitions. *Asian J. Control, 17*, 327–342.

Ruotian Liu; Rabah amour: Leonardo Brener; Isabel Demongidin (2020). Event Driven Control for Reaching a Steady State in Controlled Generalized Batches Petri nets. *Journal of International Federation of Automatic Control, 53*(4), 180-186. https://doi.org/10.1016/j.ifaco.2021.04.063

Sheetal Gokhale; Ashwini Dalvi and Irfan suddavatam (2020). Industrial Control Systems Honeypot: A formal Analysis of Conpot. *International Journal of Computer Networks and Information Security, 12*(6), 44-56. https://doi.org/10.5815/ijcnis.2020.06.04

Souravlas, S. I., & Roumeliotis, M. (2015). Petri net modeling and Simulation of Pipelined Redistributions for a Deadlock-Free System. *Cogent Engineering, 2*(1), 1–22.

Su, Z.; Qiu, M. (2019). Airport Surface Modeling and Simulation Based on Timed Coloured Petri net. *Promet-Traffic -Traffico, 31*, 479–490.

White, A., Karimoddini, A. and Karimadini, M. (2020). Resilient Fault Diagnosis Under Imperfect Observations—A need for Industry 4.0 Era, IEEE/CAA. *Journal of Automat-ica Sinica, 7*(5), 1279–1288

Wisniewski, R., Grobelna, I. and Karatkevich, A. (2020). Determinism in Cyber-Physical Systems Specified By interpreted Petri nets, *Sensors*, 1–22.

Xiaoyang Chen; Hongwei Huo; Jun Huan; Jeffrey Scott Vitter (2019). An Efficient Algorithm for Graph Edit Distance Computation. *Knowledge Based Systems, 163*, Retrieved 1st January 2019, 762-775. https://doi.org/10.1016/j.knosys.2018.10.002

Xia, C. and Li, C. (2021). Property Preservation of Petri Synthesis net Based Representation for Embedded Systems, IEEE/CAA. *Journal of Automatica Sinica, 8*(4), 905–915.

Yi-Nan Lin, Cheng-Ying Yang, Gwo-Jen Chiou, Sheng-Kuan Wang, Victor R.L. Shen, Yu-Ying Wang, Hai3 Hoang Bui & Jianzhi Wang. Caggiano Alessandra (Reviewing editor) (2022). *Smart selection from petri net modeling tools for fast developing a manufacturing system, Cogent Engineering, 9*(1). https://doi.org/10.1080/23311916.2021.2020609.

Yifan Hou and Kamel Barkaoui (2017). Deadlock Analysis and Control Based on Petri nets: A Siphon Approach Review. *Advances in Mechanical Engineering, 9*(5), 1-30. https://doi.org/10.1177/1687814017693542.

Yang, F., Wu, N., Qiao, Y., Zhou, M., Su, R. and Qu, T.(2018). Petri net-Based Efficient Determination of Optimal Schedules for Transport-Dominant Single-Arm Multi-cluster Tools, IEEE Access, 6, 355–365.

Zareef Mohammed (2022). Data Breach Recovery Areas: An Exploitation of Organization's Recovery Strategies for Surviving Data Breaches. *Organizational Cyber Security Journal, Practice, Process and People*, 2(1), 41-59. https://doi.org/10.1108/OCJ.05.2021.0014