

Table of Contents

Table of Contents.....	1
1. INTRODUCTION.....	6
2. UNDERSTANDING THE CYBER THREAT LANDSCAPE IN THE LAST DECADE.....	8
2.1. Significant Developments in Cyber Threats in the Last Decade	10
2.2. Data Compromise	11
2.3. Understanding the Impact of An Attack	14
2.3.1. Attacks on Web Application.....	17
2.3.2. Email Threats	19
2.3.3. Exploits.....	21
2.3.4. Crypto Currency and Crime.....	22
2.3.5. Malware	24
3. AUTOMATED MALWARE INFRASTRUCTURE.....	27
3.1. Introduction to Cuckoo Sandbox Environment	27
3.2. History of Cuckoo Sandbox	28
3.3. Use Cases	28
3.4. Architecture	29
3.5. Malware analysis.....	30
3.6. Malware Analysis Setup.....	32
4. IMPLEMENTATION	33
4.1. Specifications	33
4.2. Architecture	34
4.3. Network Configuration	34
4.4. Preparing the Host Machine	37
4.4.1. Installing Support Software in Host Machine	37
4.4.2. Installing Cuckoo in Host Machine.....	40
4.5. Creating a user in cuckoo.....	40
4.6. Installing the Guest Machine	41
4.6.1. Installing Virtual Box	41
4.6.2. Preparing Guest Machine	41
5. SUBMITTING MALWARE SAMPLES FOR ANALYSIS	45

5.1. Starting Cuckoo	45
5.2. Submitting Samples in Command line Utility	45
5.3. Submitting Sample in Web Interface Utility	46
5.4. Rest API	48
5.5. Analysis of Sample File in Cuckoo Sandbox.	50
6. ANALYSIS RESULTS IN CUCKOO SANDBOX.....	53
7. CLUSTERING THE ANALYSIS RESULTS INTO MALWARE FAMILIES	58
8. CODE	62
9. BIBLOGRAPHY	68
10. APPENDICES	Error! Bookmark not defined.

Summary

Cybersecurity is emerging as top concern of any company or individual connected with the Internet in the present era. Cyber-threats have developed exponentially over the last decade and have become difficult to understand and tackle. A separate study must be charted to understand the current threat landscape and the associated risk of each type to develop a solution that is efficient enough to provide a sense of security in this internet connected world. This project is aimed to understand the trend in Cyber threats in the last decade, current threat landscape and working of various types of threats. This project also aimed to study the motivations of the Cyber criminals, different techniques used in implementing the attacks and the ease of use of the available attacking tools. Furthermore, this project aimed to understand the security concerns in Organisations and remediation techniques to mitigate the risks. An automated malware analysis infrastructure was designed and built to analyse suspicious malware samples and to perform comparative security tests. Automation of Malware system helped to submit large number of malware samples to analyse and reduced the amount of time in analysing a suspicious malware sample. The automated malware analysis system can analyse the samples by executing it in sandbox environment and store the different types of analysis results in the database. These dynamic analysis results are further analysed using open source libraries and frameworks and added value to the analysis results. Based on the analysis results, the tested suspicious samples are further clustered into malware families.

Table of Figures

figure 1: vulnerabilities in last 10 years. [1]	11
Figure 2: Data compromises by industry [1].....	12
Figure 3 : Top 10 Critical Web Applications [1]	19
Figure 4 : Cuckoo Architecture [15]	30
Figure 5: Implemented Cuckoo Architecture.....	34
Figure 6: Network Connection Between Host and Guest	36
Figure 7: Python Version	37
Figure 8: Installed YARA VERSION	38
Figure 9: Yara-Python Version	38
Figure 10: Pydeep Version	39
Figure 11: Volatility Version.....	39
Figure 12: Installed Cuckoo Version	40
Figure 13: Adding user in cuckoo.....	40
Figure 14: Installed Virtual Box.....	41
Figure 15: User Access Control Permissions	41
Figure 16: Firewall Settings.....	42
Figure 17: Windows Update Settings	42
Figure 18: Agent Installation.....	43
Figure 19: Installed Agent Version.....	43
Figure 20: Snapshot of the Machine.....	44
Figure 21: Initialising Cuckoo.....	45
Figure 22 : Submission of Samples in CLI	46
Figure 23: Initialising Web Server	46
Figure 24: Web Interface to Submit Samples	47
Figure 25: Customizing Web Server.....	47

Figure 26: Initializing Rest Api Interface	48
Figure 27: Customized Rest Api	48
Figure 28: Submitting Sample Using Rest Api	49
Figure 29: Tasks Submitted in Rest Api	49
Figure 30: Initialising Cuckoo	50
Figure 31: Submitting Sample File for Analysis	50
Figure 32: Restoring Virtual Machine	51
Figure 33: Agent is Initialized	51
Figure 34: Execution of Sample File in Guest Machine	52
Figure 35: Analysis Results Stored in Host Machine	53
Figure 36: Analysis Results in Web UI	53
Figure 37: Static Analysis in Web UI	54
Figure 38: Network Analysis in Web UI	54
Figure 39: Process memory in web UI	55
Figure 40: Volatility analysis in web UI	55
Figure 41: Compare analysis option in web UI	56
Figure 42: Export analysis in web UI	56
Figure 43: Reboot analysis in web UI	57
Figure 44: Malware analysis output	58
Figure 45: Interesting information	59
Figure 46: Pandas data frame	59
Figure 47: Data frame in CSV file	60
Figure 48: TFIDF Matrix	60
Figure 49: K-Means clustering	61
Figure 50: Clustered malware samples into malware families	61
Figure 51: Cuckoo Working Directory	Error! Bookmark not defined.

1. INTRODUCTION

“Are we protected?” is the general question when we talk about the exponential growth of internet connections. Cyber threats are more sophisticated now and Malware is the key weapon of the attackers to exploit the existing vulnerabilities or to develop potential malicious intent. Cyber Attacks are more attractive as they are cheaper, convenient and less risky without constraints of geography and distance. The anonymous nature of the internet makes it even more difficult to identify the attackers. Rapidly developing technologies has not only generated huge profits to the organisations but also brought in major security threats such as fraud, network infrastructure attacks, identity theft and data breaches. Industries are at great risk as the trend is going to increase more and more devices connected to each other with the adaption of the new technologies. As we are more dependent on computer networks and information technology solutions, there is an urgent requirement to identify the malware and develop an effective malware defensive mechanism.

To mitigate cybercriminal risks, it is important to have a clear understanding of quantitative and qualitative changes in the cybersecurity threat landscape. Malware can compromise computer functions, steal data and bypass access controls causing harm to the host machine. There various categories of Malware and the working, intent is different from each other. Malware has the highest proportion of computer security incident across many industries. ABACUS survey respondents ranked malware as the most significant computer security incident. The detection of Malware is very difficult as they obfuscate to avoid detection and attackers use persistence techniques to reload the malware even after a reboot.

Why do we need to Automate the Malware Analysis Process?

Automated malware analysis techniques have been developed to handle large number of malware samples. Automation of certain aspects of malware analysis is essential for organizations that process large numbers of malicious programs. When using Cuckoo as an automated malware analysis tool, the amount of time analysing a malware is expected to be reduced in a conventional manner. There are steps in dynamic malware analysis that take a lot of time; one of the instances is when we set up a virtualized malware environment to run. The process may seem easy, but it will be quite time consuming if we have several malwares to analyse. As malware became more sophisticated, we needed more technology to easily analyse malware without compromising our environment. So, we use Sandboxing environment to analyse the malware safely and securely without worrying about the damages that will occur during the analysis process.

2. UNDERSTANDING THE CYBER THREAT LANDSCAPE IN THE LAST DECADE

There is significant development in the Cyber threats over the last decade. Earlier threats are opportunistic as the motive of the attack was to steal money with card details, online login credentials and any other valuable information from the victim. But, now the threats are more advanced and persistent from the Cyber Criminals. Cyber Criminals are skilled professionals who can breach any large network possible and Organisations are investing in Security Intelligence Teams to protect their infrastructure. Security tools and event loggers are developed to identify the trends in cyber security landscape. Vulnerability scanning, and testing services internally is part of the operations in the organisations with huge network infrastructure to understand the malware development, phishing trends, exploit kits and collected critical information trading. We will go through the different types of attacks and their impacted industries details.

Data Compromise: Retail, Finance & Insurance and Hospitality are the most affected industries in the data compromise and data breach and the impact of these breaches is not immediately identified in most of the cases.

Web Attacks: The attackers gather preliminary data to identify the weak packages and tools on the targeted network. Attackers take advantage of these weakness in the network and try to compromise the devices or networks. The significant preplanning methods increased the success rate of an attack in the last decade. It is difficult to distribute the software updates to all the devices in the network and most of the devices are vulnerable as they are not properly patched.

Email Threats: The spam content in the emails is reduced significantly in the last decade, thanks to the various filters in Email Gateways but attackers are finding new ways to deliver malicious attachments, links to malicious web pages and enticements to perform transactions. Spam content delivered through emails has shown significant spike because of Necurs botnet. These botnets operate in short bursts of intense spamming.

Exploit Kits: Exploit kits provide an easy to use packages to silently exploit the vulnerabilities in the victim's machine while browsing the internet. The victims are convinced to click the malicious links that serves malicious content which is highly automated in nature. They range from web based, client based and even zero-day vulnerabilities. These exploit kits played a crucial role in the evolution of Malware.

Crypto Currency: The anonymous nature of the crypto currency transactions has made it a popular alternative medium of exchange for ransomware attacks. The popular ransomware attackers demanded payment in bitcoin. Nearly USD15 billion crypto currency is stolen from exchanges in the last 5 years. Some Attackers use the victim's computer resources to mine the crypto currency coins.

Malware: Malware is an umbrella term that describes any malicious code that is harmful to computer. There are different types of malware such as Virus, Trojan Horse, worm, spyware, ransomware, and all are unique in their traits and characteristics. The malware is highly sophisticated now and can obfuscate to avoid detection and these malwares can reload even after a reboot. The vulnerabilities in the machines is the general cause of attacks.

2.1. Significant Developments in Cyber Threats in the Last Decade

2008 – Storm botnet was developed to distribute spam and malware in the form of infected e-cards and attachments to the users for free. Command-and-control (C&C) spam also became popular in the same year.

2009 - Fake antivirus were developed to detect false infections and demand the users to pay ransom for the “full version” of software to clean the infected machines.

2010 – The spam botnets reached its peaks and the spam generated was mostly for fake advertisements of pharmaceutical products. Lecthic spam is notable as it generated more than 20 percent of all the spam.

2011 – Attackers tried to spread the Blackhole exploit kit in large scale campaign of link spam. Cyber criminals used large botnets to send out unsolicited emails containing malware to infect victim’s computers.

2012 – A series of service providers takedown dealt a fatal blow to many botnets.

2013 – Attackers used Spam campaigns as primary weapon and distributed CryptoLocker ransomware and malware to steal personal information such as banking credentials.

2014 – Attackers distributed malware through Microsoft Word documents containing malicious macros.

2015 – Attackers targeted Higher Officials in the business by compromising the emails. Notable Scams Business Email Compromise (BEC) and CEO fraud scams.

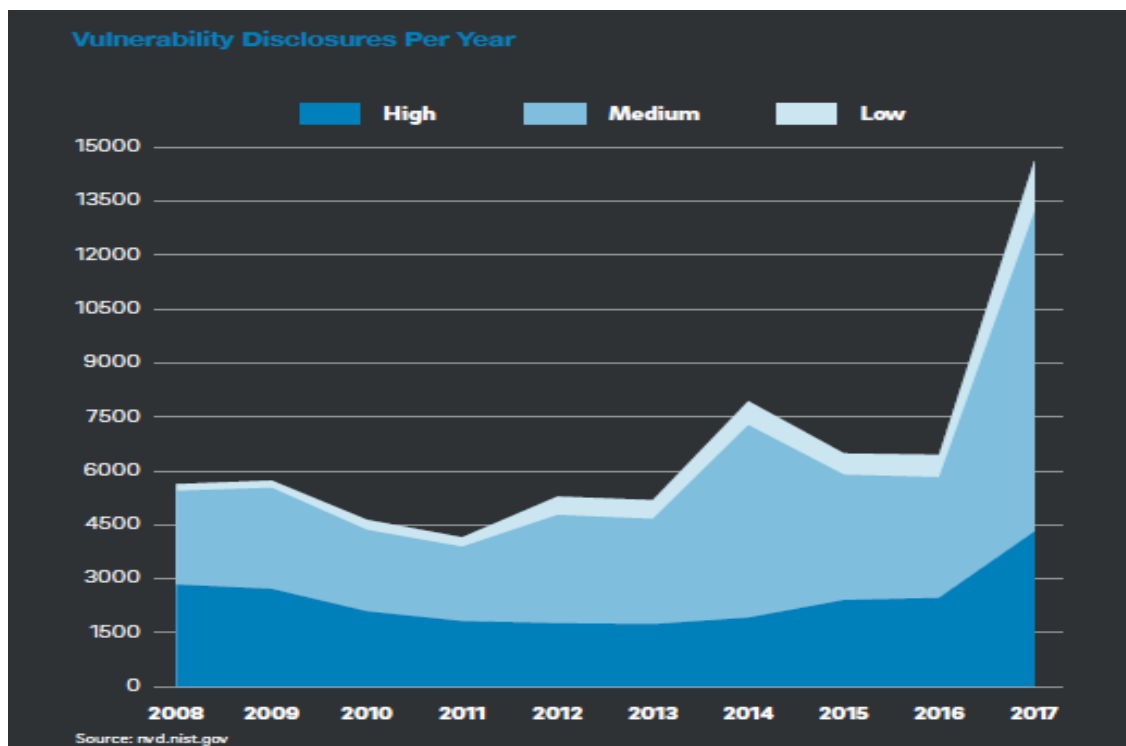


FIGURE 1: VULNERABILITIES IN LAST 10 YEARS. [1]

2.2.Data Compromise

Security compromises and Data breaches are biggest threats in today's internet connected world and affected organisations environments by putting the infrastructure at risk.

Companies of all sizes must focus on security and incident response to protect their network.

“Based on Trustwave Global Security Report, Attackers targeted all the sectors in Industries and surveys show that retail industry share 17 % percent of the total incidents, followed by Finance and insurance at 13 percent. The below graph shows the data compromised attacks in the last 2 years.[1]“. Attackers mainly focus their target on the service provider industries which in turn opens opportunities to target other businesses who are clients to the service providers. Earlier there were large volumes of smaller breaches, but now the attackers are targeting to compromise service providers and the impact is seen on multiple business

sectors. Attackers target payment card data with card-track (magnetic stripe) data incidents of nearly 23 percent of the total incidents and card-not-present (CNP) data such as online or e-commerce transactions compromised incidents are nearly 20 percent.

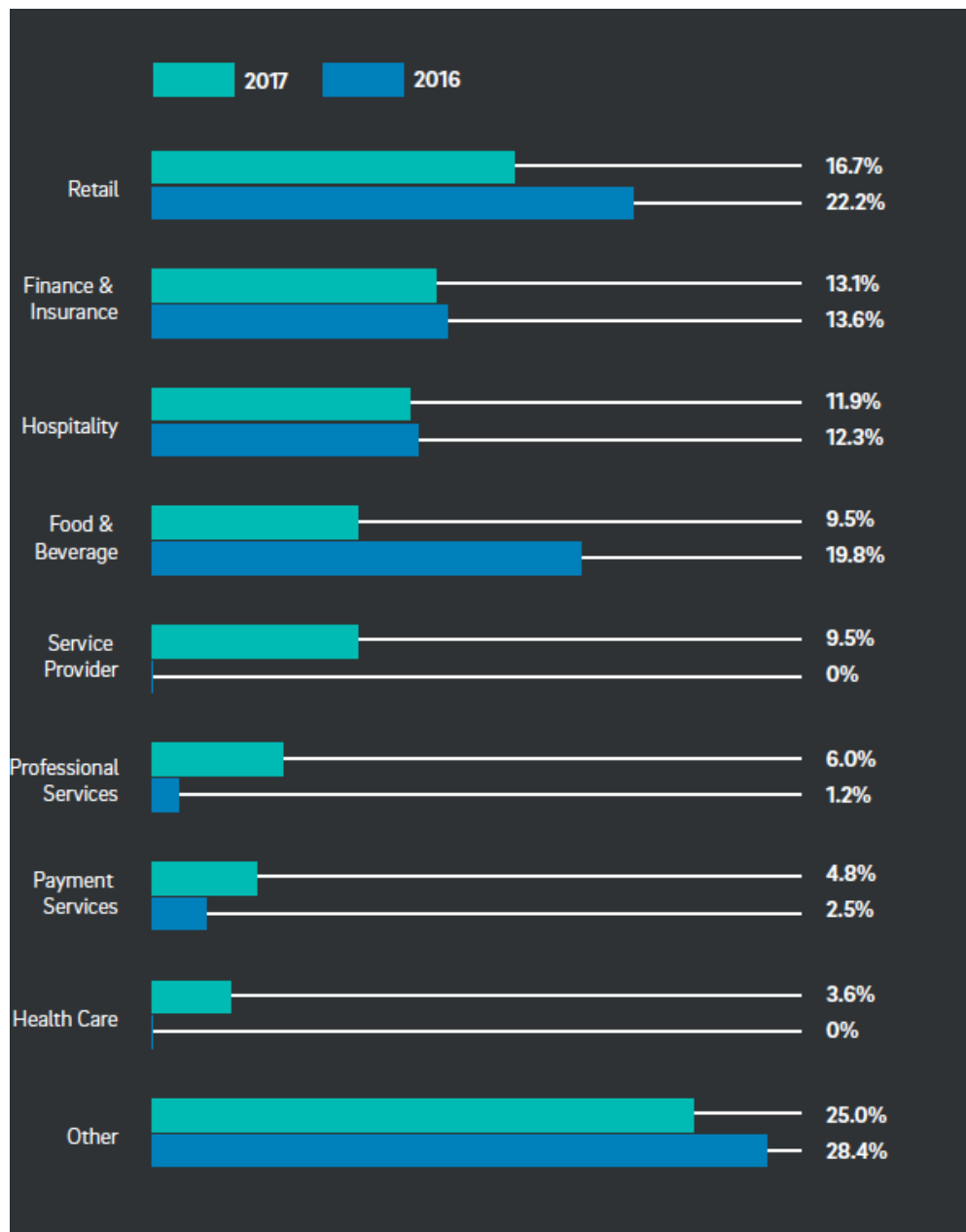


FIGURE 2: DATA COMPROMISES BY INDUSTRY [1]

The vulnerable account management systems enabled fraudulent ATM transaction breaches targeting cash transactions. There is a significant rise in the breaches targeting personally identifiable information (PII) and proprietary data. The noticeable strategy of the attackers is attackers will focus on reaching their goals even if there are other possible breaches. For example: Attackers who gained administrative level access of large infrastructures has only installed ransomware even though they have an opportunity to steal the valuable data of the company.

Based on the sources like the National Vulnerability Database (NVD), CVE Details and Exploit-DB.com, security incidents and individual vulnerabilities have taken a sharp upturn over the past 10 years. But, it doesn't mean that software developed are getting less safe, but the attackers are finding innovative ways to attack with vulnerabilities. Higher vulnerability disclosure rates are inevitable even when Software developers are security conscious while developing software. Higher the vulnerability, greater is the potential for exploitation. Cyber criminals have sufficient patience, time and resources to reconnaissance before they actually perform an attack. The attacker groups use zero-day exploits in limited attacks to prevent or delay the widespread awareness of the exploits. White hat researchers contribute to this Exploit – DB with proof of concept exploit code which is useful for the attackers though their intention is to use the work for the remediation and patch development.

2.3.Understanding the Impact of An Attack

Cyber Security experts need to understand how long it takes to detect a breach in the organisation and how long it takes to perform a remediation action. This can be done by identifying the below details during an attack.

Intrusion: This refers to unauthorised access of the attacker to victim's machine. The attackers often maintain access for very long periods of time and have ample amount of time to collect sensitive data. With successful access to the victim's machine, attackers can install multiple backdoors which increases the difficulty of removing them from the network. Attackers can use various techniques ranging from Phishing and other Social Engineering techniques to perform the attacks. Remote based attacks are possible when the service providers details are compromised. Code injection and file upload are the most common techniques used to compromise websites and these techniques are easy to implement by using Exploit kits. The main motivation of the malicious activity is to take control of the victim's infrastructure and install malicious software in their network and they can target any device in the network. Attackers attack individually or form cooperative attack groups to perform attacks in large scale and share their resources for a successful attack. The common entry point in most of the attacks was email containing malicious attachment. As the target opens these attachments to view, malicious software or code starts to run download the create malicious macros or to download multiple files from the internet. The attackers can even use Social Engineering techniques and trick the victims to open these malicious attachments. The next step is to escalate the privileges by using pass-the-hash technique to have the admin or root level access. Once they have the access, the attackers use cloud services to keep track of malicious activities in the infected systems and perform additional

malicious activities. The businesses cannot block the cloud services as they are dependent on these for business as usual. Intrusion is relatively easy with customisable exploit toolkits, such as Metasploit, NMap, John the Ripper, Air Crak-ng, Nikto and Veil Framework. Attackers purchase and fake identities to bypass the additional security controls and they even sign some of the executables with valid certificates from a trusted security authority. Attackers can bypass the network security controls by using Command and Control Servers and they initiate the connections from unknown systems located in multiple locations.

Detection: The longer the attacker has access to the infrastructure, the more harm the attacker can do. So, the Organisations must have security infrastructure such as anti-malware services or suspicious windows event loggers to detect the suspicious activities in the network traffic. The alerts of such discovered activities must be sent to the Security administrators in the Company to respond and perform a cleaning task. Earlier in many organisations, the detection is delayed, or the breach is detected after the attacker left the network. But the recent advancements in detections tools and mechanisms such as endpoint detection and response (EDR) solutions has made a significant reduction in the time needed to detect and respond to an incident. Now, businesses detected majority of the breaches on the same day the breach had happened. Organisations should try and maintain the logs for months in at least archive mode to help in proper investigation. One of the interesting facts is that businesses can detect the internal compromises quickly than external compromises and malicious insiders can cause more damage to the network. There is an increase in the attacker notification in the last two years because of the large breakouts of ransomware attacks. Detecting the attacks is an added advantaged layer of Security. If the internal detected systems are working as expected, the companies don't need to invest in external

investigators. Despite these advantages, many small organisations are not willing to use the detection tools and techniques to defend or detect a breach.

Remediation: The administrators must clean the compromise and the attackers should no longer have access to the attacked infrastructure. As a part of the remediation practices, companies can take proactive countermeasures to defend the attacks. When there are indicators of compromise, the organisations should take necessary steps to remediate or mitigate the compromise impact on the organisation. Setting up traps can quarantine the malicious executables on the endpoint. Traps notifies the user about the quarantined files and alerts administrator about the suspicious activity. Traps will even move the quarantined files from local drive to local quarantine folder. Scanning all the machines in the network on weekly or daily basis will assist in finding the vulnerabilities of the machines and patching them appropriately. Real time data monitoring and alerting systems will help in investigating the malicious activities and mitigating the impact of the threat. Threat Intelligence teams must focus on the methods and mechanisms the attackers used to compromise the infrastructure.

In this section, we will first discuss web applications attacks with focus on the strategies and mechanisms attackers implemented. Next, we discuss various types of email threats and the strategies involved in successful exploitation. In the third part of this section, we discuss significant exploit kits that are developed during last year and examine the state of the exploit kit landscape now and we will also focus on several major exploit kits that are disappeared. We will look at cryptocurrencies and its involvement in malicious motivations. In the last section, we will discuss about the malware which caused data compromise incidents and explore known malware families.

2.3.1. Attacks on Web Application

The attack landscape has changed a lot in the last decade with changes in motives, tool sets, attack strategies and targeted vulnerabilities and exploits. Attacks have become more sophisticated and attackers show signs of significant preplanning in identifying weak or vulnerable packages and tools on the targeted infrastructure, but the investigations and surveys show that the basic attack techniques attackers used are traditional techniques of attacking such as cross-site scripting (XSS), Brute Force, SQL injection (SQLi), DoS and DDoS. We will understand these attacking techniques in detail below.

Cross-site Scripting (XSS) refers to client-side code injection attack wherein an attacker can execute malicious scripts or payloads into a trusted website or web application. Last year, 40% of the attacks are based on Cross Site Scripting. In this attack, an untrusted source will have access to inject its own code into a web application, and that malicious code is included with dynamic content delivered to a victim's browser. This attack is possible because browsers are not advanced, and the attack is mostly preferred by inexperienced attackers as there are available scripts and tools to perform these kinds of attacks.

SQLi is the second most common attack with 24 percent of attacks in the last year. In this attack, attacker enters a malicious SQL code in the input field of the web page and the server-side code submits the malicious SQL code to the database without validating the input. A successful SQL injection attack can give complete access of the database to the attacker. Sanitization of the input and using Web Application Firewall filters can help to reduce these kinds of attacks.

“Path or Directory Traversal attacks are another common attack which attempt to access unauthorized files or directories outside the web root folder. Attackers inject patterns such as

“../” sequences or its variations to manipulate and move up in the server directory hierarchy.” A successful attack can allow attackers to have root or administrator level access. The attacks can be mitigated by using the latest web server software and patching all the vulnerabilities as required.

DDoS is a type of DOS attack where multiple compromised systems are used to target a single system causing a Denial of Service (DoS) attack. These compromised systems bombard a targeted web server with requests, the server will be overloaded its resources and the services will be inaccessible to legitimate users. DDoS may not provide access to any resources, but it helps to distract automated security defence appliances from not responding to a more serious attack.

Common Vulnerabilities and Exposures are released every year which covers multiple vulnerabilities in several popular security appliances. Attackers can exploit these vulnerabilities and can gain administrative level access to the security devices. Organisations must update and patch with latest firmware to avoid these vulnerabilities in the network. To limit the exposure, it is recommended to limit the access to the affected devices by using access control lists and network segmentations. Firmware updates are available to patch the vulnerabilities and it is the responsibility of the organisation to be aware of the latest patches released. Security administrators must be aware of the known risks and frequently update core and install the latest security patches and should attend to security incidents or warnings.

Web Application Firewalls protect the critical data and applications as they are designed to recognise and restrict suspicious activity. These Firewalls operate on application layer, top

level in OSI Layer and have access to all the below layer protocols in OSI Layers. So, they can monitor and protect websites from a variety of attacks.

In the last ten years, there has been a sharp rise in number of attacks on internet-connected devices. These devices range from traditional network infrastructure hardware in the organisations to smart connected devices called the internet of things (IoT), which include home appliances, vehicles, wearable devices and many other technologies. These devices run on unique embedded systems not hardened through years of responding to attacks.

OWAS listed the below as top 10 critical web applications in the last four years.

RANK	2013	2017
A1	Injection	Injection
A2	Broken Authentication and Session Management	Broken Authentication
A3	Cross-Site Scripting (XSS)	Sensitive Data Exposure
A4	Insecure Direct Object References	External Entities (XXE)
A5	Security Misconfiguration	Broken Access Control
A6	Sensitive Data Exposure	Security Misconfiguration
A7	Missing Function Level Access Control	Cross-Site Scripting
A8	Cross-Site Request Forgery (CSRF)	Insecure Deserialization
A9	Using Components with Known Vulnerabilities	Using Components with Known Vulnerabilities
A10	Unvalidated Redirects and Forwards	Insufficient Logging & Monitoring

FIGURE 3 : TOP 10 CRITICAL WEB APPLICATIONS [1]

2.3.2. Email Threats

Email threats are constantly evolving and will continue to be the biggest threats as email has become central to business communications. Attackers have continued to find new ways of using email to breach the security of Organisation. Email threats are developed from traditional attacks using viruses and spam to advanced targeted attacks like phishing, spear-

phishing and wire transfer phishing. Botnets such as Necurs botnet continued to pump spam to email inboxes. The good news is that overall spam volumes fell again in 2017 to the lowest level in the last 10 years but is still high.

The spam-borne malware is increased in 2016 and 2017 and the spike is mostly because of Necurs botnet which is currently the world's largest botnet to send spam traffic. Necurs operates in short bursts of intense spamming activity followed by periods of dormancy. At its peak, the botnet is capable of sending spam from between 200,000 and 400,000 unique IP addresses daily. Necurs botnet activity decreased in 2017 compared to last two years, but the start-and-stop nature of the botnet makes it difficult to conclude. Necurs begin to exploit a newly discovered technique for exploiting Word's Dynamic Data Exchange (DDE) protocol to execute malicious code. Attackers can craft malicious documents using the macro-less .docx file type. All Word users should apply the latest available security patches to secure their machine from this exploit.

“According to Trustwave Global Security Survey Attackers deliver more than 90 percent of spam-borne malware inside archive files as attachments labelled as business files [1].” These malicious files are typically a small script file or a document that contains code when clicked runs malicious code. The small script/code is highly obfuscated, and it will download and execute additional malware. The script is also capable of fetching another download script, often of a different file type, which then downloads the payload to attack. Below are the common payloads.

“Banking Trojans, such as Ursnif, Emotet and Trickbot, steal information such as online credentials for banking websites from windows systems [2]”. “Kovter is an ad fraud and is very difficult to detect as they are a fileless malware family that performs multiple malicious

actions [3] ”. “Ransomware families including Cerber, FakeGlobe and Locky are spread in the attachments of emails [4]”. Most of the spam-delivered malware came in the form of attachments with .vbs extension files containing embedded PowerShell code. JavaScript .js files, PDF files and Microsoft Word documents are common file types in these kinds of attacks. Phishing is another common technique that attackers use to trick the customers. “Phishing is a cybercrime in which a target or targets are contacted by email, telephone or text message by someone posing as a legitimate institution to lure individuals into providing sensitive data such as personally identifiable information, banking and credit card details, and passwords [5]. The information is then used to access important accounts and can result in identity theft and financial loss.”

“CEO fraud” (or “CEO wire transfer fraud”) is a technique that attackers use to steal money from companies by spoofing the emails. In this scam, the target is usually a financial officer with the authority to send money on behalf of a company. The scammer sends the target an email message that looks like to originate from the company’s CEO, asking them to send a payment to a vendor or other party and the Reply-To message header is set to attacker’s account to ensure that replies or follow-up messages from the target reroute to the attacker and not the CEO [6] .”

2.3.3. Exploits

“An exploit is a piece of software, a chunk of data, or a sequence of commands that takes advantage of a bug or vulnerability to cause unintended or unanticipated behaviour to occur on computer software, hardware, or something electronic (usually computerized). Such

behaviour frequently includes things like gaining control of a computer system, allowing privilege escalation, or a denial-of-service (DoS or related DDoS) attack [7].”

The Shadow Brokers hacking group leaked High-profile zero-day exploits which enabled the widespread WannaCry and Petya ransomware attacks and disclosed a number of tools and exploits, including ETERNALBLUE which was widely used in WannaCry and Petya ransomware attacks at the same time, the exploit kit landscape has targeted narrow attacks instead of widespread exploitation. Most of the exploits must be delivered through malicious documents or applications and so, they are not suitable to exploit landing pages. The magnitude of exploit kits is not a straight-line graph. The attackers prefer to attack as soon as the exploit kits are launched and soon the exploit kits disappear from the market space.

2.3.4. Crypto Currency and Crime

“Cryptocurrency (or crypto currency) is a digital asset designed to work as a medium of exchange that uses strong cryptography to secure financial transactions, control the creation of additional units, and verify the transfer of assets. Cryptocurrencies are a kind of alternative currency and digital currency (of which virtual currency is a subset). Cryptocurrencies use decentralized control as opposed to centralized digital currency and central banking systems [8].”

There is a sharp rise in crypto-jacking attacks, exploiting the power of victims' computers to mine crypto-currency. The other kind of crime includes crypto currencies as a choice of payment method in ransomware attacks. Bitcoin is the first crypto-currency born in 2009 [9]. The market for crypto currency is highly volatile and hackers have taken note of this

payment method because of its anonymous nature as the payments are untraceable as they are not linked to any bank accounts or addresses.

In Crypto-currencies, the history of transactions is logged on a distributed ledger and the process is called mining. It requires masses of computing power which pays crypto-currency in return to the miners. Hackers make use of botnets, a network of infected computers which can be controlled by hacker to log the transactions to distribution ledger. Through phishing emails attackers share a compromised link and convince the victims to click the link which directs users to a website domain that allows hackers to run a short script designed to begin the mining. Last year's WannaCry attack was the largest ransomware attack which affected Windows systems all over the world started the trend to demand the ransom to be paid in crypto currencies and even offered the choice to negotiate the ransom value. Attackers started attacking more profitable online exchanges, the cryptocurrency equivalent of banks and financial markets instead of targeting individuals. The most common obfuscation techniques were packing and crypting as they render the signature-based detection techniques difficult and ineffective [10]. Cybercriminals quickly noticed the value and popularity and developed several mechanisms that enable people to parcel out mining work to thousands of victim's machines. There are sites that runs Monero mining software in a browser instead of advertisements. When a user visits the website, the user's browser executes the Coinhive JavaScript code embedded in the website page, causing the visitor's CPU to mine the Monero cryptocurrency and deliver coins discovered to the site owner's wallet and to Coinhive [1]. Recently, CheckPoint Software mentioned that the level of exploitation done by Coinhive miners made them the 'most wanted' malware, with nearly 55% of their customers exposed to one or more crypto-currency mining malware families.

Providing Security to Crypto currency is difficult because anything that is connected to the internet is vulnerable and can become a target for compromise.

2.3.5. Malware

“Malware (a portmanteau for malicious software) is any software intentionally designed to cause damage to a computer, server or computer network. Malware does the damage after it is implanted or introduced in some way into a target’s computer and can take the form of executable code, scripts, active content, and other software. The code is described as computer viruses, worms, Trojan horses, ransomware, spyware, adware, and scareware, among other terms. Malware has a malicious intent, acting against the interest of the computer user and so does not include software that causes unintentional harm due to some deficiency, which is typically described as a software bug [11]. “

The purpose of malware is to steal information from the victim’s machine. The infected machines can be used for various illicit purposes such as sending email spams, to host contraband data or to engage in distributed denial-of-service attacks.

Recent Types of Malware Encounters The largest category consisted of memory scrapers and dumpers that POS malware often used to grab payment card numbers. Those gathered details are transmitted to an attacker or store them for later exfiltration. The second largest category are the Downloaders that fetch other malware from malicious servers. The next largest category is remote access Trojans (RATs) which open a backdoor for attackers to access the system. Injectors attempt to hide malware within existing processes running on the attacked systems and keystroke loggers to capture as much data as possible. One of the most significant means of initiating cyber-attacks is by doing initial penetration test.

Penetration testing tools are often used by attackers to understand the vulnerabilities on the target machine. In many Organisations, Penetration testing is a standard procedure that security professionals use to test the security of machines in the network. Pen testing tools are designed to discover vulnerabilities and other weaknesses in the network. White-hat security experts use Pen testing tools to harden systems and unfortunately, the same tools are used by black-hat attackers frequently to identify the vulnerabilities and compromise them. The top pen testing tools we saw attackers use in 2017 include Cobalt Strike Beacon, Metasploit Meterpreter, Empire PowerShell, Mimikatz. Security Experts find it difficult to detect Malware as Malware often use obfuscation techniques to hide the true nature of their code's functionality to avoid detection. As the Security Tools are advancing, Malware is advanced as well. Malware is using advanced techniques such as packing and crypting. A packer is a utility that bundles program files and resources into a single archive file. Packing algorithms are designed to avoid anti-malware systems and this technique is used by legitimate software developers as well [12] . Most Crypters encrypt the file and the Crypter software also offers the user with options to generate the hidden executables which are hard to be detect by Security Tools. These techniques are used by some packers as well. The ultimate goal for malware authors is to make it Fully undetectable. Being able to mislead security tools is difficult for malware authors as well. So, Malware authors uses simple functions and escape sequences to go undetected for a while and once they are detected, they easily change their files again. Attackers usually write scripts which ensures that their malware executes every time the computer reboots by manipulating the registry entries or by creating services which starts the Malware execution automatically. There is an increase in file less malware which tries to avoid detection by security software that scans files on disk.

The Autorun key contains a PowerShell script that loads script in another key. The second script injects a small shellcode that decompresses and executes the malware and DLL in the memory of the third key which opens a backdoor for an attacker. Attacker can use this backdoor to download and execute arbitrary programs.

After stealing the data, attackers need to implement exfiltration techniques on the malware samples to utilise the data. But, exfiltration can create a trail which may identify the malware's source. So, attacker has to connect to another remote computer to exfiltrate the data or they should use another malware component to handle the exfiltration. The attackers have to maintain connections to transfer the exfiltrated data and this is mostly done by using the raw TCP socket connections to communicate between the victim and the attacker. Remote Access Trojan and bots are frequently used to transfer data.

3. AUTOMATED MALWARE INFRASTRUCTURE

3.1.Introduction to Cuckoo Sandbox Environment

As defined by Wikipedia, “In computer security, a sandbox is a security mechanism for separating running programs. It is often used to execute untested code, or untrusted programs from unverified third-parties, suppliers, untrusted users and untrusted websites.”.

A sandbox provides a tightly controlled set of resources such as disk and memory for guest programs to run in. A Sandbox environment is often used to reduce the risk of harming the host machine or operating system while giving enough access to allow proper testing [13] .

To cope with the malware attacks, a strong automated malware analysis system is required.

Cuckoo Sandbox is an open source automated malware analysis system to run an unknown and untrusted application or file inside an isolated environment and observe its behaviour. It can automatically run and analyse files and collect comprehensive analysis results with details of what the malware does while running inside an isolated operating system.

Cuckoo is designed to analyse the following kinds of files:

- Generic Windows executables
- DLL files
- PDF documents
- Microsoft Office documents
- URLs
- PHP scripts
- Almost everything else

Cuckoo Sandbox can retrieve the following type of results:

- Traces Cuckoo of calls performed by all processes spawned by the malware.
- Files being created, deleted and downloaded by the malware during its execution.
- Memory dumps of the malware processes.
- Network traffic trace in PCAP format.
- Screenshots taken during the execution of the malware.
- Full memory dumps of the machines [\[14\]](#) .

3.2.History of Cuckoo Sandbox

Cuckoo Sandbox was started as a Google Summer of Code project in 2010 within The HoneyNet Project. It was originally designed and developed by Claudio “nex” Guarnieri, who is still the project leader and core developer. After initial work during the summer 2010, Cuckoo was publicly announced and distributed for the first time with the first beta release on 5 February 2010. Cuckoo was continuously part of supporting projects of Google Summer of Code for many years. Cuckoo Foundation was established in March 2014 as non-profit organization to work for the growth of Cuckoo Sandbox and the surrounding projects and initiatives. With continuous effort, Cuckoo released version 2.0 Release Candidate 1 on 21, February 2016 as an open source for daily usage. Cuckoo Sandbox can be used without licensing requirements [\[14\]](#) .

3.3.Use Cases

Cuckoo’s modular design and powerful scripting capabilities allows us to use Cuckoo as a standalone application or can be integrated in larger frameworks of distributed environment. Cuckoo can be customised to any aspect depending on the requirement to analyse, process and report the behaviours. Cuckoo can analyse any file and provide a detailed report

outlining the behaviour of the file when executed inside a realistic but isolated environment.

This open source sandbox can automate the analysing task of any suspicious file under Windows, MAC, Linux, and Android Operating Systems.

By default, Cuckoo can be used to analyse:

“Analyse many different malicious files (executables, office documents, pdf files, emails, etc) as well as malicious websites under Windows, Linux, Mac OS X, and Android virtualized environments.

Trace API calls and general behaviour of the file and distil this into high level information and signatures comprehensible by anyone.

Dump and analyse network traffic, even when encrypted with SSL/TLS. With native network routing support to drop all traffic or route it through InetSIM, a network interface, or a VPN.

Perform advanced memory analysis of the infected virtualized system through Volatility as well as on a process memory granularity using YARA [14].”

3.4.Architecture

The Cuckoo Sandbox consists of a central management software and isolated virtual or physical machine. Central management software handles sample execution and analysis. Each analysis is launched in a new isolated virtual or physical machine.

The main components of Cuckoo’s infrastructure are a Host machine which hosts the Central Management software and several Guest machines which are either virtual or physical machines. The host runs the core sandbox component to manage the entire analysis process, while the guests are isolated environments where the malware samples are executed and analysed safely.

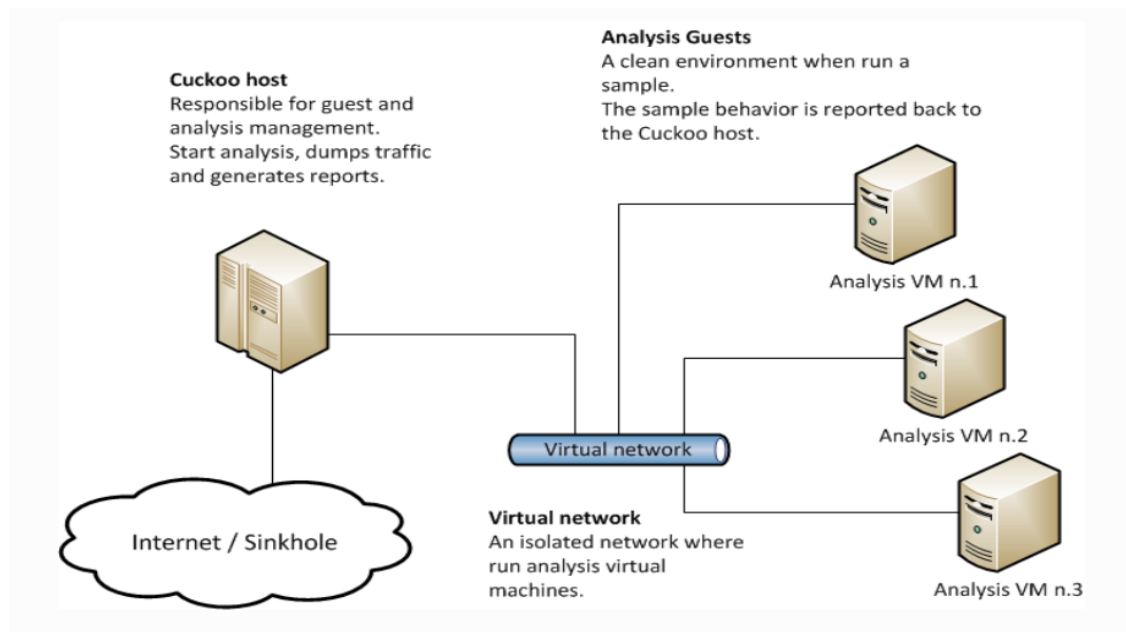


FIGURE 4 : CUCKOO ARCHITECTURE [15]

3.5.Malware analysis

As defined by Wikipedia, “Malware analysis is the study or process of determining the functionality, origin and potential impact of a given malware sample such as a virus, worm, trojan horse, rootkit, or backdoor. Malware or malicious software is any computer software intended to harm the host operating system or to steal sensitive data from users, organizations or companies. Malware may include software that gathers user information without permission.”

Malware analysis involves a complex process to identify malware behaviour, what they do, what they want and what their main objectives are. Forensics, reverse engineering, disassembly, debugging, these activities are taking a long time. Malware analysis must be performed to be more proactive to new and modern threats that can suddenly attack organisation's network. It is also an advanced detection form before antivirus vendors found

a new malware sample. The main aim of malware analysis is to understand how malware works to protect any network by preventing malware attacks.

Methodologies of Malware Analysis: Malware Analysis techniques enable Security Analysts to quickly understand the risks and intentions of a given sample malware. There are two common malware analysis methodologies commonly used by malware analysts: static analysis (or code analysis) and dynamic analysis (or behavioural analysis).

1. Static Analysis (or Code Analysis): In static analysis process, the malware is not executed, and the source code of malware samples is not readily available. Malware is first disassembled and decompiled, and after successfully performing Reverse Engineering, the low-level assembly code is analysed. Most malware analysts prefer static analysis at an earlier stage in the malware analysis process as it is safer than dynamic analysis. The challenge in static analysis is the complexity in sophisticated malware as these kinds of malware implement anti-debugging systems to prevent from analysing the pieces of code. Complex configuration is not required for the test environment, because there is no actual execution of the malware itself.

2. Dynamic Analysis (or Behaviour Analysis): Dynamic analysis (behaviour analysis) is a malware analysis process that performs malware execution and observes malware activity. It also observes changes when malware is executed. Infecting a machine with malware may be very dangerous. Malware infection on machine can damage to the machine, such as modification or deletion of the files, registry change, steal confidential data / information, and so on. So, there is a need of safe environment when performing malware analysis and the network should not be connected to production networks. The changes made to the filesystem, registry, processes and

network communication can be monitored with dynamic analysis. Dynamic Analysis provided the advantage of complete understanding how malware behaves in the real environment. A more complex testing setup is expected, as the malware is being executed.

3.6.Malware Analysis Setup

There are two options in building a malware analysis setup, a physical environment and a virtualization environment. Building your physical laboratory will require a lot of money and time to build the environment where as building a virtualization malware laboratory saves money and time. Virtualization software allows to save a virtual machine's state as it runs and can revert to that state at any time using snapshot. Using snapshots feature, you can have a virtual machine environment that contains an operating system with dynamic and static analysis tools to analyse the suspicious file and can save the session to analyse the behaviour. The snapshot feature adds the advantage as we don't have to worry about malware that infects Guest OS, as it is easy to restore it to the previous state. Analyses of malware in virtual environment helps to shorten the time in analysing malware samples. The disadvantage in implementing automated malware analysis is that it can be easily detected by malware writers as it often uses evasion techniques such as anti- debugging, packers, encryption, obfuscating code etc.

4. IMPLEMENTATION

For this project, I used virtualization environment. I installed the Central Management Software in Linux machine and used Windows 7 OS as the Guest Machine. The execution of the suspicious files is done in the Guest OS and the results are exported to Host Machine in JSON, HTML and PDF formats.

4.1.Specifications

Hardware Requirements: There is no specific hardware requirement for this project and I'm using the below specifications.

The Host

- Ubuntu 16.04
- 4Gb RAM
- 64-bit processor
- VirtualBox 5.2.8 r121009
- Cuckoo Sandbox

The Guest

- Windows 7 Home Basic
- 4 GB RAM
- 64-bit processor

4.2.Architecture

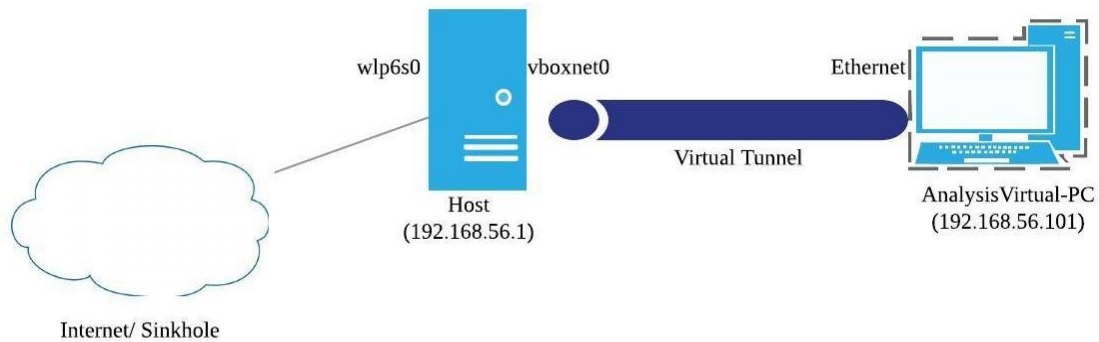


FIGURE 5: IMPLEMENTED CUCKOO ARCHITECTURE

As shown in the figure above, Guest Machine is running Windows 7 OS in virtual environment and is assigned with IP address: 192.168.56.101. The Guest machine is connected to Host machine via a virtual tunnel on vboxnet0 interface. The host machine interface: wlp6s0 forwards all the traffic from Virtual Machine to the internet.

4.3.Network Configuration

I installed Windows 7 Operating System as the Guest Machine and configured the necessary settings such as Network Settings and shared folder settings to exchange files between Host and Guest machines.

As mentioned earlier, the traffic is forwarded to Host OS and filtering rules are applied as the Host OS acts as Gateway to the traffic originating from VM. To setup the network environment as mentioned above, use the following commands.

- i. Create a host-only interface on host and configure vboxnet0 with IP: 192.168.56.1

Commands:

```
$ vboxmanage hostonlyif create
```

```
$ vboxmanage hostonlyif ipconfig vboxnet0 --ip 192.168.56.1
```

- ii. Configure to set the NIC on the guest Virtual Machine as a host only interface.

Commands:

```
$ vboxmanage modifyvm windows --hostonlyadapter1 vboxnet0
```

```
$ vboxmanage modifyvm windows --nic1 hostonly
```

- iii. Configure Virtual Machine with static IP, default Gateway and DNS as below.

Static IP - 192.168.56.101

DNS - any DNS server (8.8.8.8)

Default Gateway - 192.168.56.1

- iv. Configure the IP table and filtering rules to enable internet access on the Virtual Machine

```
# Internal traffic.
```

```
$ iptables -A FORWARD -o eth0 -i vboxnet0 -s 192.168.56.0/24 -m  
conntrack --ctstate NEW -j ACCEPT
```

```
# Existing connections.
```

```
$ iptables -A FORWARD -m conntrack --ctstate ESTABLISHED,RELATED -j  
ACCEPT
```

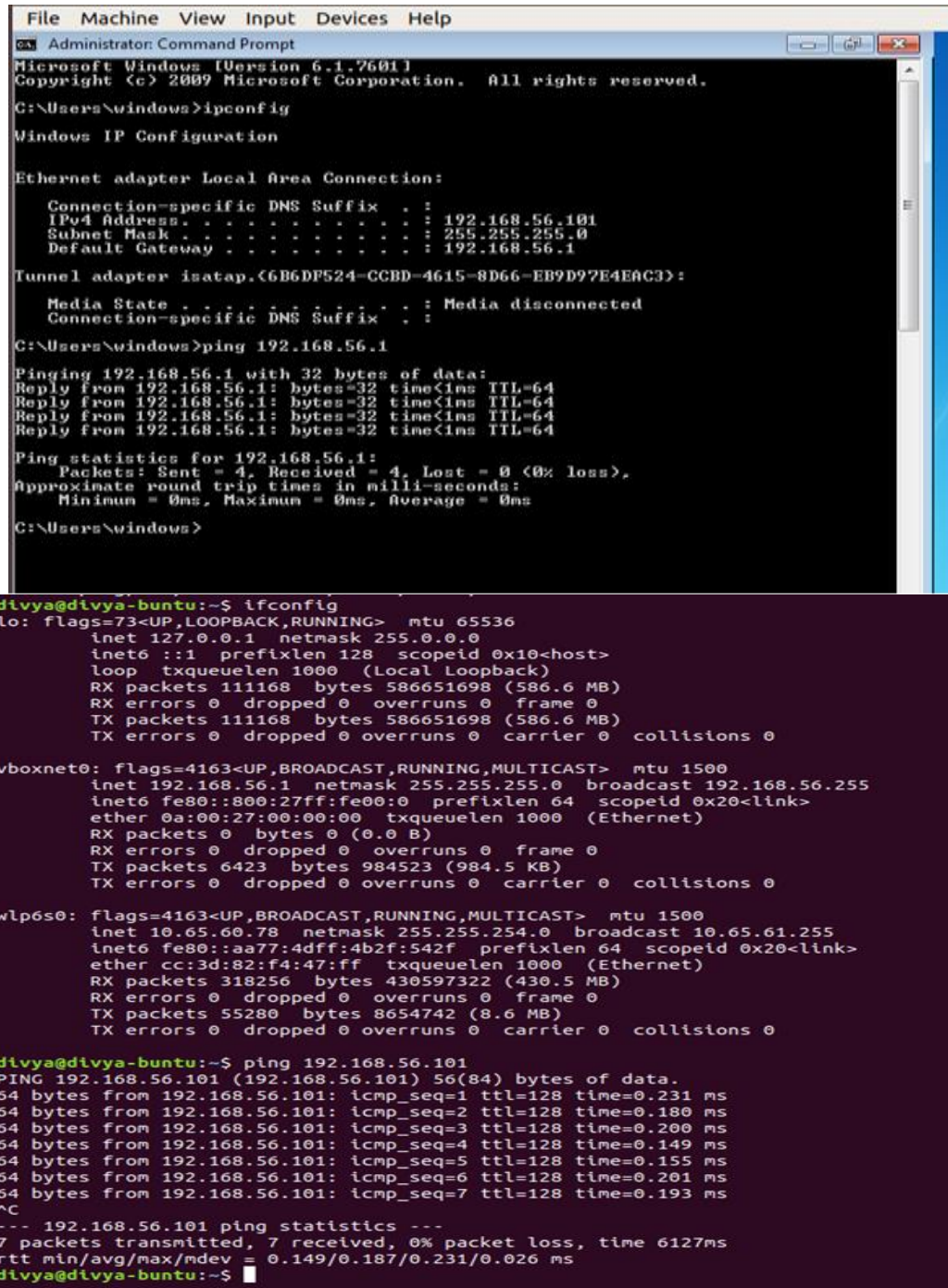
```
#NATting the traffic.
```

```
$ iptables -A POSTROUTING -t nat -j MASQUERADE
```

- v. IP forwarding can be enabled with the below command.

```
$ echo 1 > /proc/sys/net/ipv4/ip_forward
```

After the above configuration, Internet access from the guest to the Internet should be enabled and ping back and forth between the guest and the host should be successful.



```

File Machine View Input Devices Help
Administrator: Command Prompt
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\windows>ipconfig

Windows IP Configuration

Ethernet adapter Local Area Connection:

    Connection-specific DNS Suffix  . : 
    IPv4 Address. . . . . : 192.168.56.101
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 192.168.56.1

Tunnel adapter isatap.{6B6DF524-CCBD-4615-8D66-EB9D97E4EAC3}:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . : 

C:\Users\windows>ping 192.168.56.1

Pinging 192.168.56.1 with 32 bytes of data:
Reply from 192.168.56.1: bytes=32 time<1ms TTL=64
Reply from 192.168.56.1: bytes=32 time<1ms TTL=64
Reply from 192.168.56.1: bytes=32 time<1ms TTL=64
Reply from 192.168.56.1: bytes=32 time<1ms TTL=64

Ping statistics for 192.168.56.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\Users\windows>

divya@divya-buntu:~$ ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 111168 bytes 586651698 (586.6 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 111168 bytes 586651698 (586.6 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

vboxnet0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.56.1 netmask 255.255.255.0 broadcast 192.168.56.255
    inet6 fe80::800:27ff:fe00:0 prefixlen 64 scopeid 0x20<link>
    ether 0a:00:27:00:00:00 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 6423 bytes 984523 (984.5 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlp6s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.65.60.78 netmask 255.255.254.0 broadcast 10.65.61.255
    inet6 fe80::aa77:4dff:4b2f:542f prefixlen 64 scopeid 0x20<link>
    ether cc:3d:82:f4:47:ff txqueuelen 1000 (Ethernet)
    RX packets 318256 bytes 430597322 (430.5 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 55280 bytes 8654742 (8.6 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

divya@divya-buntu:~$ ping 192.168.56.101
PING 192.168.56.101 (192.168.56.101) 56(84) bytes of data.
64 bytes from 192.168.56.101: icmp_seq=1 ttl=128 time=0.231 ms
64 bytes from 192.168.56.101: icmp_seq=2 ttl=128 time=0.180 ms
64 bytes from 192.168.56.101: icmp_seq=3 ttl=128 time=0.200 ms
64 bytes from 192.168.56.101: icmp_seq=4 ttl=128 time=0.149 ms
64 bytes from 192.168.56.101: icmp_seq=5 ttl=128 time=0.155 ms
64 bytes from 192.168.56.101: icmp_seq=6 ttl=128 time=0.201 ms
64 bytes from 192.168.56.101: icmp_seq=7 ttl=128 time=0.193 ms
^C
--- 192.168.56.101 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6127ms
rtt min/avg/max/mdev = 0.149/0.187/0.231/0.026 ms
divya@divya-buntu:~$
  
```

FIGURE 6: NETWORK CONNECTION BETWEEN HOST AND GUEST

4.4.Preparing the Host Machine

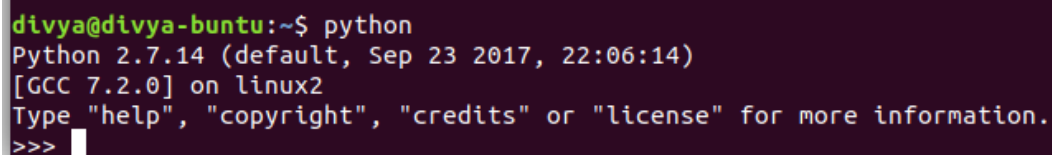
4.4.1. Installing Support Software in Host Machine

The host and guest machines must be prepared to be installed with few applications and libraries before implementing the Cuckoo Sandbox.

The supporting software are as follows.

Python: Cuckoo uses Python extensively to perform analysis.

Command: `$ sudo apt-get install python`



```
divya@divya-buntu:~$ python
Python 2.7.14 (default, Sep 23 2017, 22:06:14)
[GCC 7.2.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> █
```

FIGURE 7: PYTHON VERSION

Other Python Libraries: Install python and other required libraries so that the cuckoo will be able to communicate from host to guest system.

Command: `$ sudo apt-get install git mongodb libffi-dev build-essential python-django python python-dev python-pip python-pil python-sqlalchemy python-bson python-dpkt python-jinja2 python-magic python-pymongo python-gridfs python-libvirt python-bottle python-pefile python-chardet tcpdump -y`

Yara: Yara is a multi-platform program running on Windows, Linux and Mac OS X. This library is optional and is used for matching Yara signatures

Commands

`$ sudo apt-get install autoconf libtool libjansson-dev libmagic-dev libssl-dev -y`

```
$ wget https://github.com/plusvic/yara/archive/v3.4.0.tar.gz -O yara-
3.4.0.tar.gz

$ sudo tar -zxf yara-3.4.0.tar.gz

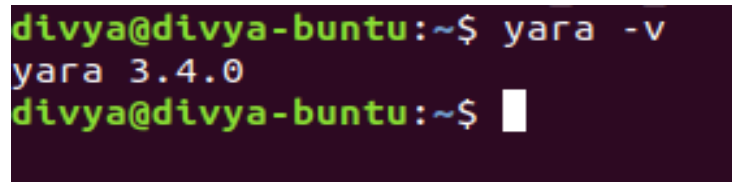
$ cd yara-3.4.0

$ sudo ./bootstrap.sh

$ sudo ./configure --with-crypto --enable-cuckoo --enable-magic

$ sudo make

$ sudo make install
```



```
divya@divya-buntu:~$ yara -v
yara 3.4.0
divya@divya-buntu:~$
```

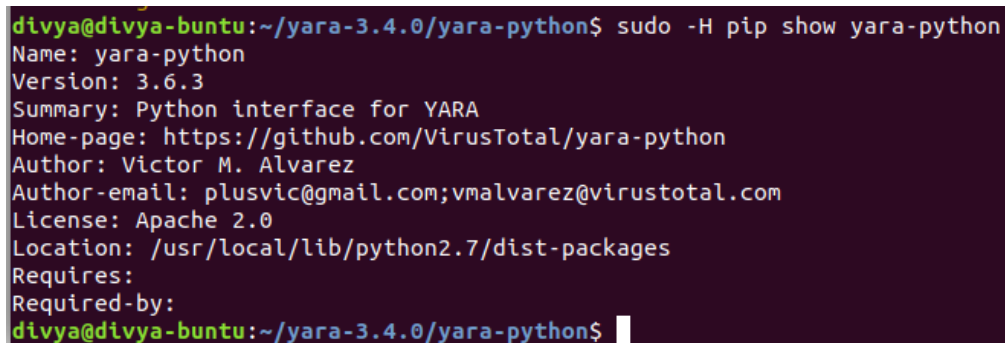
FIGURE 8: INSTALLED YARA VERSION

Yara- Python: Yara – Python is an extension in Yara Module.

```
$ cd yara-python

$ sudo python setup.py build

$ sudo python setup.py install
```

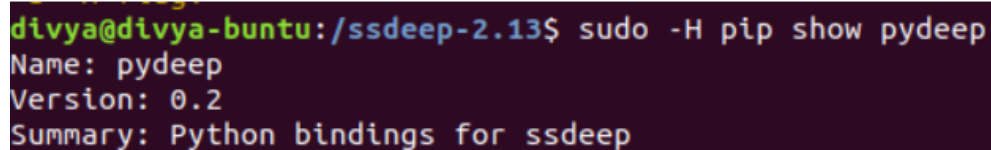


```
divya@divya-buntu:~/yara-3.4.0/yara-python$ sudo -H pip show yara-python
Name: yara-python
Version: 3.6.3
Summary: Python interface for YARA
Home-page: https://github.com/VirusTotal/yara-python
Author: Victor M. Alvarez
Author-email: plusvic@gmail.com;vmalvarez@virustotal.com
License: Apache 2.0
Location: /usr/local/lib/python2.7/dist-packages
Requires:
Required-by:
divya@divya-buntu:~/yara-3.4.0/yara-python$
```

FIGURE 9: YARA-PYTHON VERSION

PyDeep: PyDeep is a machine learning / deep learning library with focus on unsupervised learning [16]. The library has a modular design, is well documented and purely written in Python/Numpy.

Command: `$ pip install pydeep`



```
divya@divya-buntu:/ssdeep-2.13$ sudo -H pip show pydeep
Name: pydeep
Version: 0.2
Summary: Python bindings for ssdeep
```

FIGURE 10: PYDEEP VERSION

Volatility: A single, cohesive framework analyzes RAM dumps from 32- and 64-bit windows, linux, mac, and android systems [17] .

Command:

`$ pip install openpyxl`

`$ pip install ujson`

`$ pip install pycrypto`

`$ pip install distorm3`

`$ pip install pytz`

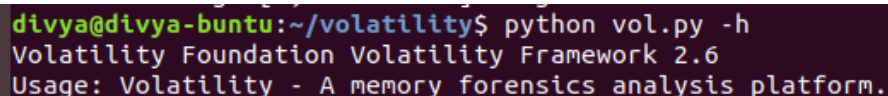
To continue install of the volatility type below commands

`git clone https://github.com/volatilityfoundation/volatility.git`

`$ cd volatility`

`$ python setup.py build`

`$ python setup.py install`



```
divya@divya-buntu:~/volatility$ python vol.py -h
Volatility Foundation Volatility Framework 2.6
Usage: Volatility - A memory forensics analysis platform.
```

FIGURE 11: VOLATILITY VERSION

4.4.2. Installing Cuckoo in Host Machine

Command: `$ sudo pip install -U cuckoo`

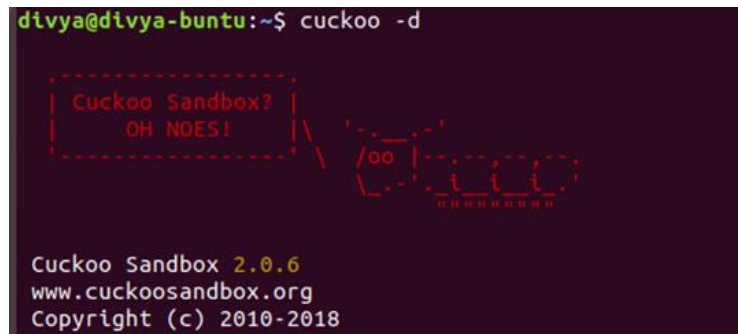


FIGURE 12: INSTALLED CUCKOO VERSION

4.5. Creating a user in cuckoo

Create a new user to create and run the virtual machines. Add the new user to the vboxusers group. Cuckoo uses this user to identify and launch these Virtual Machines. Cuckoo can also be launched from default user.

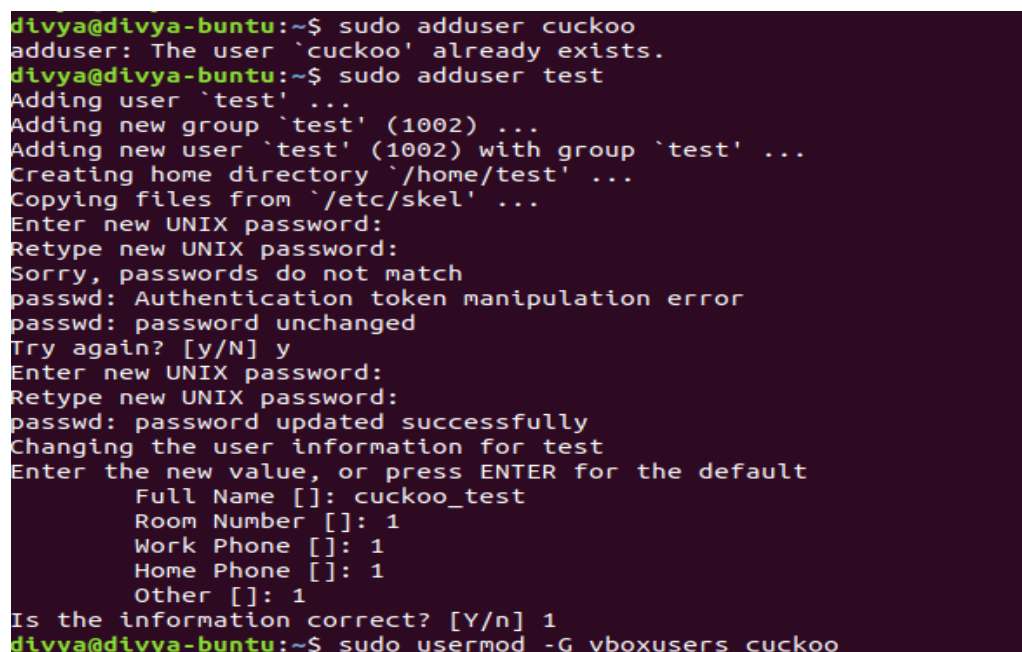


FIGURE 13: ADDING USER IN CUCKOO

4.6.Installing the Guest Machine

4.6.1. Installing Virtual Box

As mentioned earlier, I'm using Virtual environment for the Guest Machine. To install the Guest Machine, we need to create a virtual environment and install the Guest Machine in this Virtual Box. VirtualBox 5.2.8 r121009 is installed in the Linux Machine.

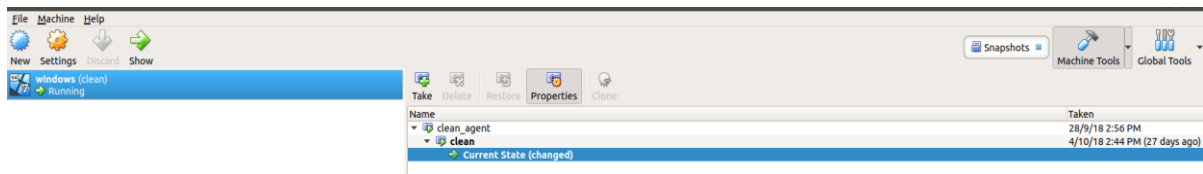


FIGURE 14: INSTALLED VIRTUAL BOX

4.6.2. Preparing Guest Machine

The Guest machine is installed in a virtual box with Windows 7 OS, 64 bit as recommended. To prepare the Guest Machine, we need to do the below.

i. Disable User Access Control

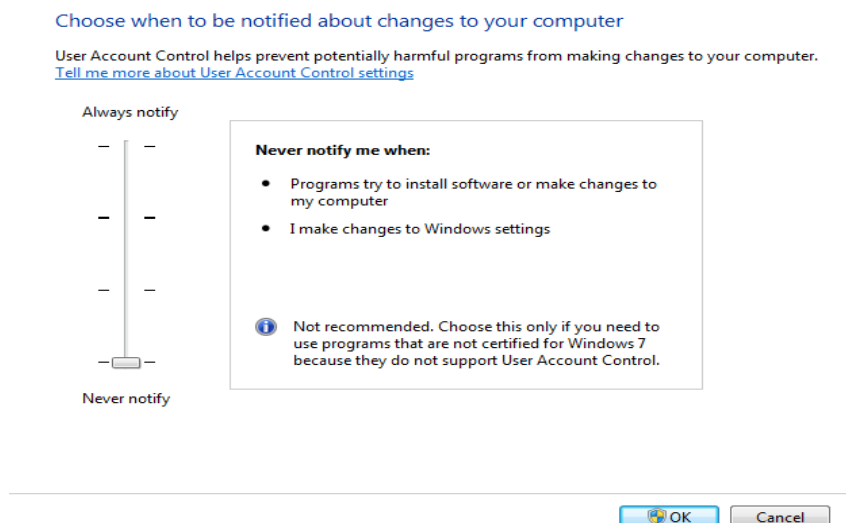


FIGURE 15: USER ACCESS CONTROL PERMISSIONS

ii. Disable Firewall and Windows Updates.

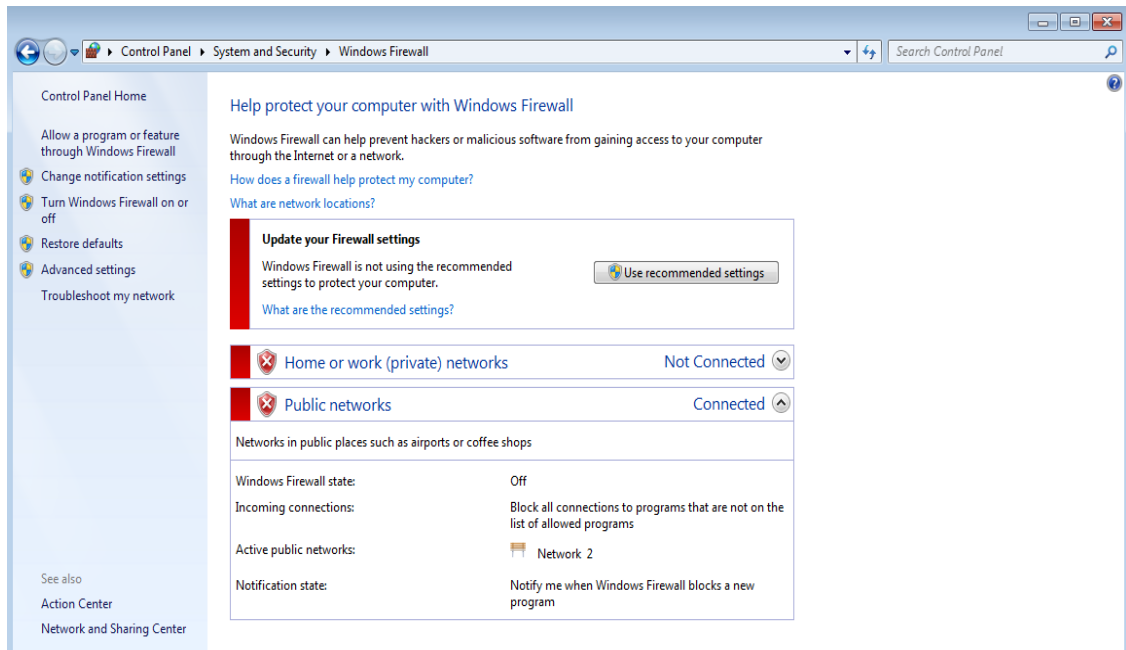


FIGURE 16: FIREWALL SETTINGS

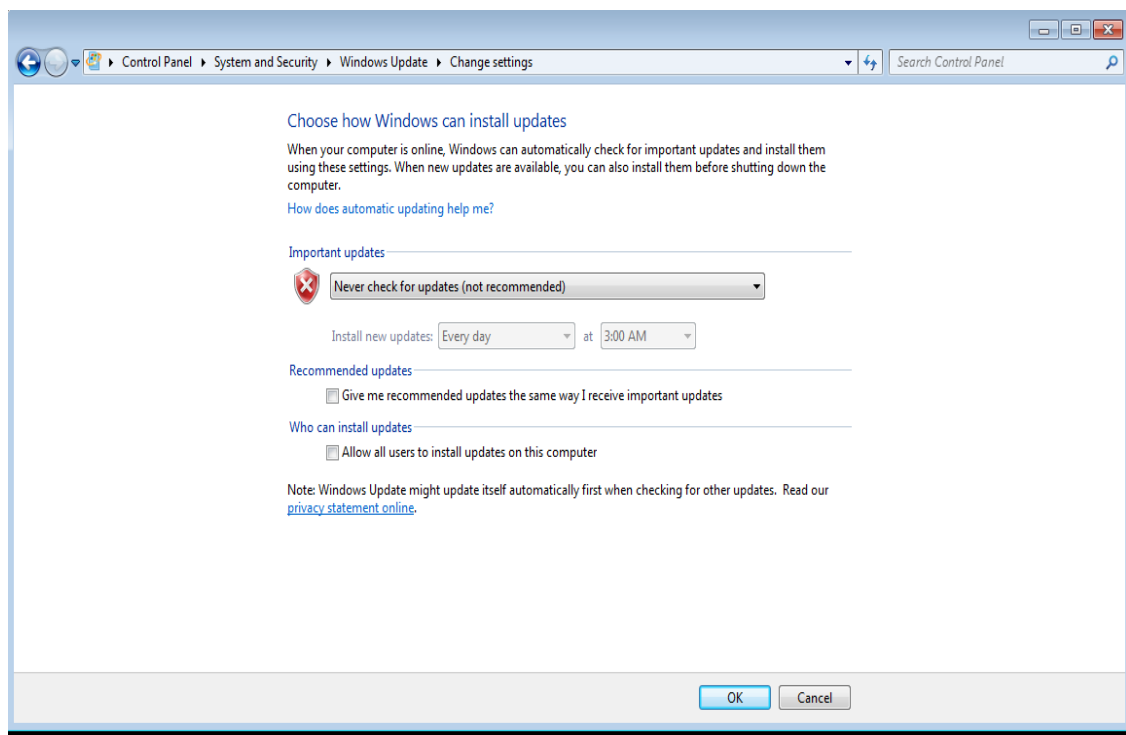


FIGURE 17: WINDOWS UPDATE SETTINGS

iii. Install agent in Guest Machine:

Agent enables the communication and exchange of data between Guest Machine and Host Machine. Run Command line in the administrator mode and run the agent. The Agent will launch a small API server that the host will be able to talk to.

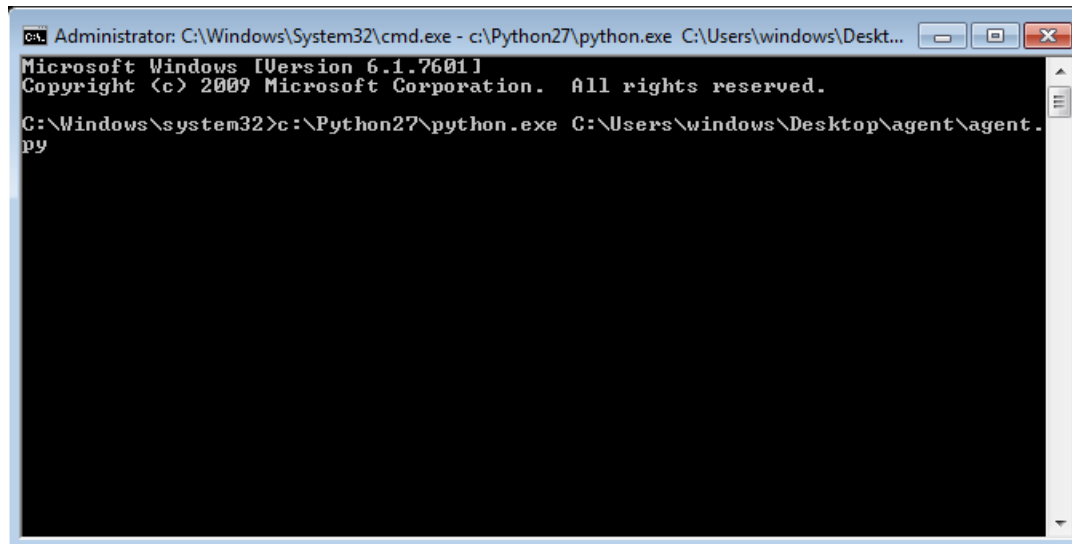


FIGURE 18: AGENT INSTALLATION

iv. Agent Verification

Agent can be launched on port 8000 by default. Verify the running status of the agent in the browser by entering the URL as Guest IP on port 8000. In my installation, Guest IP is 192.168.56.101 and agent verification can be done on <http://192.168.56.101:8000> as shown below. Kindly note that the version of the installed agent is “0.8” .

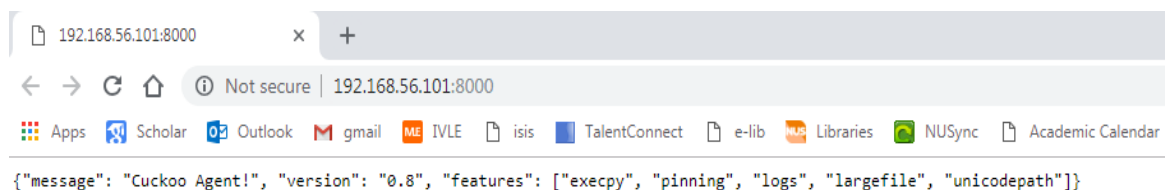


FIGURE 19: INSTALLED AGENT VERSION

v. Snapshot of Guest Machine

Verify the communication between Host and Guest Machines. If everything is working as expected, save the snapshot of the Guest Machine. We will load the same state of the machine every time we need to run analysis of new suspicious sample.

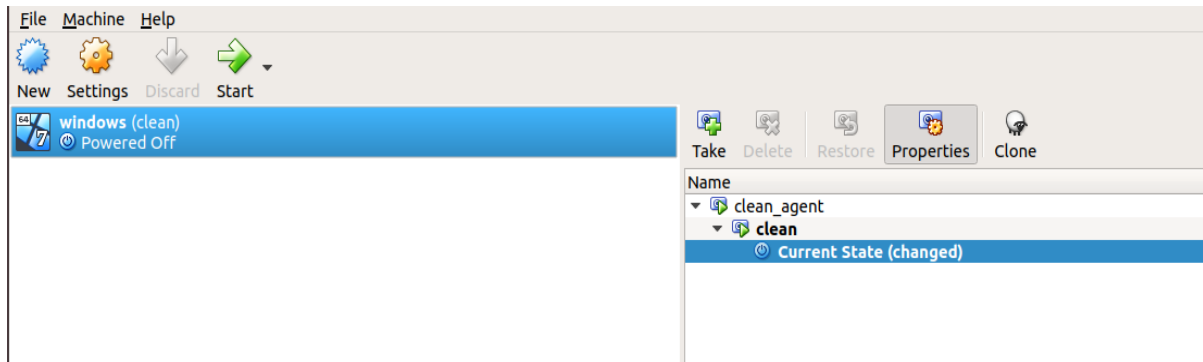


FIGURE 20: SNAPSHOT OF THE MACHINE

5. SUBMITTING MALWARE SAMPLES FOR ANALYSIS

5.1.Starting Cuckoo

To analyse the sample, we need to initialise the cuckoo daemon with the command as below.

Command: `$ cuckoo -d`



FIGURE 21: INITIALISING CUCKOO

Cuckoo starts and wait for the samples to analyse. Samples can be submitted using multiple utilities.

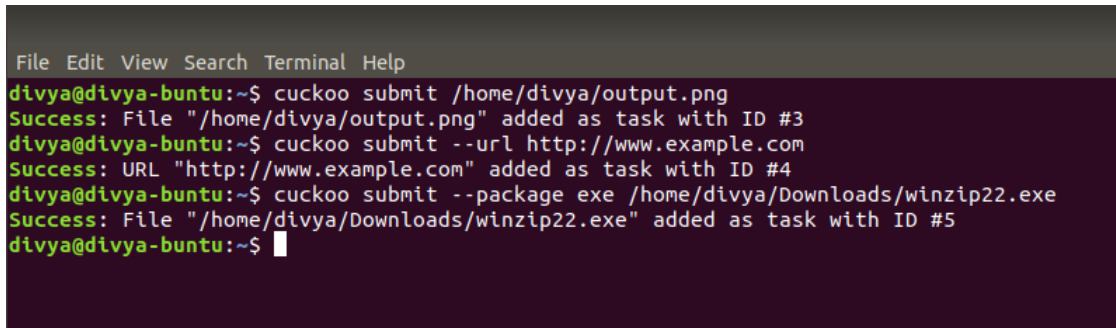
5.2.Submitting Samples in Command line Utility

Cuckoo provides submit utility as shown below to submit the sample files through command line. We may specify multiple files or directories at once and cuckoo will enumerate all its files and submit them one by one. After successful submission, cuckoo will return the task id of submitted files.

There are multiple options available to customize and submit the samples as shown below.

Cuckoo can analyse the URLs/ Hashes and almost all the available packages. Cuckoo also

provides options to define the package of the sample suspicious file. If the package is not known, it can be left with default package for Cuckoo to analyse.

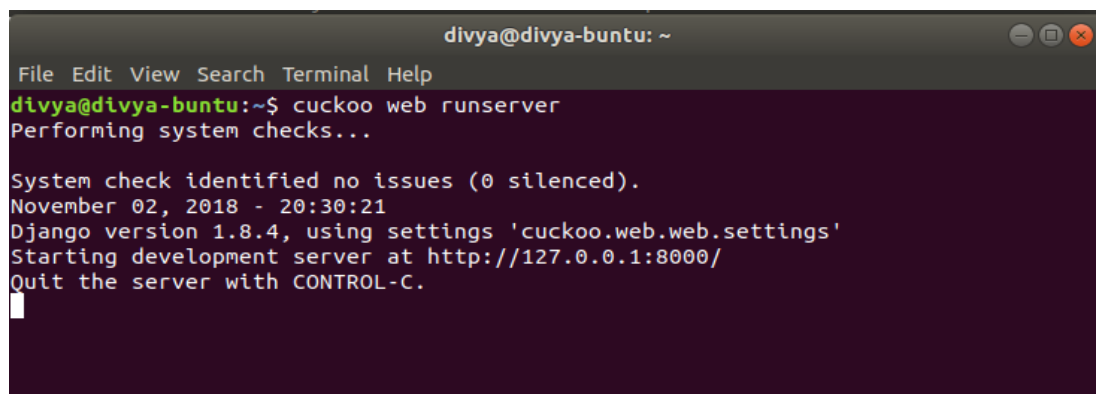


```
File Edit View Search Terminal Help
divya@divya-buntu:~$ cuckoo submit /home/divya/output.png
Success: File "/home/divya/output.png" added as task with ID #3
divya@divya-buntu:~$ cuckoo submit --url http://www.example.com
Success: URL "http://www.example.com" added as task with ID #4
divya@divya-buntu:~$ cuckoo submit --package exe /home/divya/Downloads/winzip22.exe
Success: File "/home/divya/Downloads/winzip22.exe" added as task with ID #5
divya@divya-buntu:~$
```

FIGURE 22 : SUBMISSION OF SAMPLES IN CLI

5.3. Submitting Sample in Web Interface Utility

To use Cuckoo Web service, we need to launch the web service utility with the below command. The web service runs on local host with a default port number 8080.



```
divya@divya-buntu: ~
File Edit View Search Terminal Help
divya@divya-buntu:~$ cuckoo web runserver
Performing system checks...

System check identified no issues (0 silenced).
November 02, 2018 - 20:30:21
Django version 1.8.4, using settings 'cuckoo.web.web.settings'
Starting development server at http://127.0.0.1:8080/
Quit the server with CONTROL-C.
```

FIGURE 23: INITIALISING WEB SERVER

Open the link in the browser and click on submit a file for analysis option to upload a file for analysis. Click Submit URL/Hashes option to submit URL/Hash.

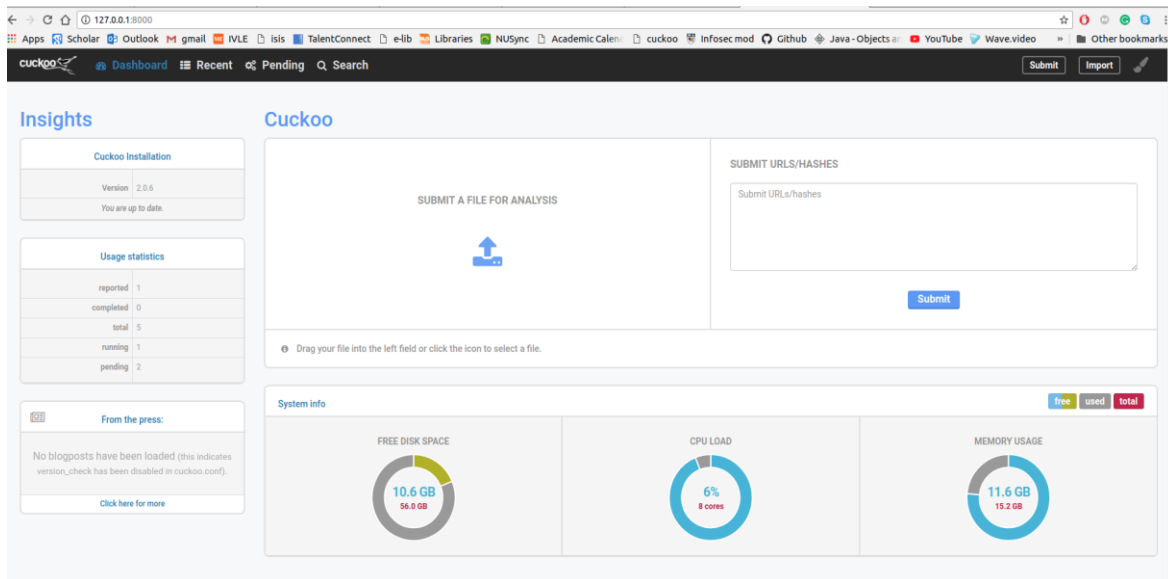


FIGURE 24: WEB INTERFACE TO SUBMIT SAMPLES

The destination port number can be customized as shown in the command below.

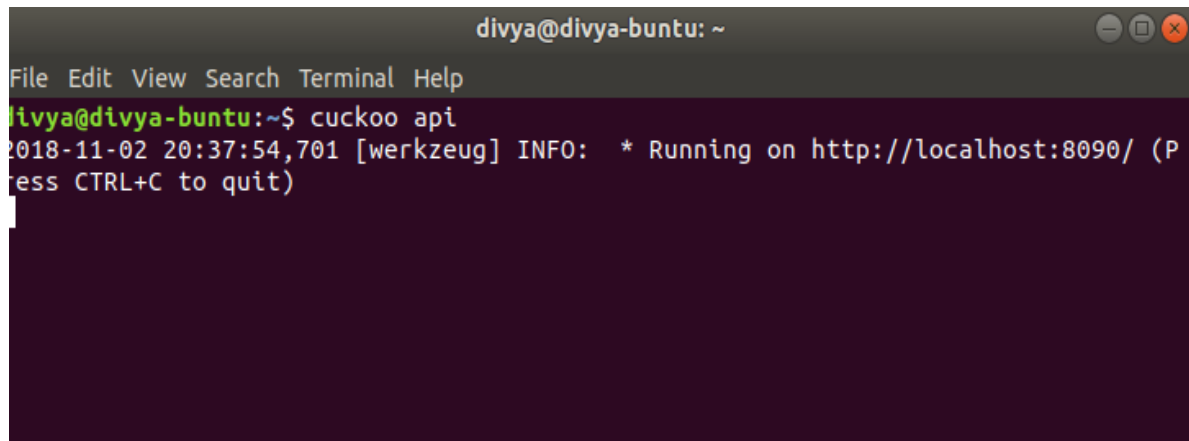
```
divya@divya-buntu: ~
File Edit View Search Terminal Help
divya@divya-buntu:~$ cuckoo web runserver 9001
Performing system checks...

System check identified no issues (0 silenced).
November 02, 2018 - 20:33:20
Django version 1.8.4, using settings 'cuckoo.web.web.settings'
Starting development server at http://127.0.0.1:9001/
Quit the server with CONTROL-C.
```

FIGURE 25: CUSTOMIZING WEB SERVER

5.4. Rest API

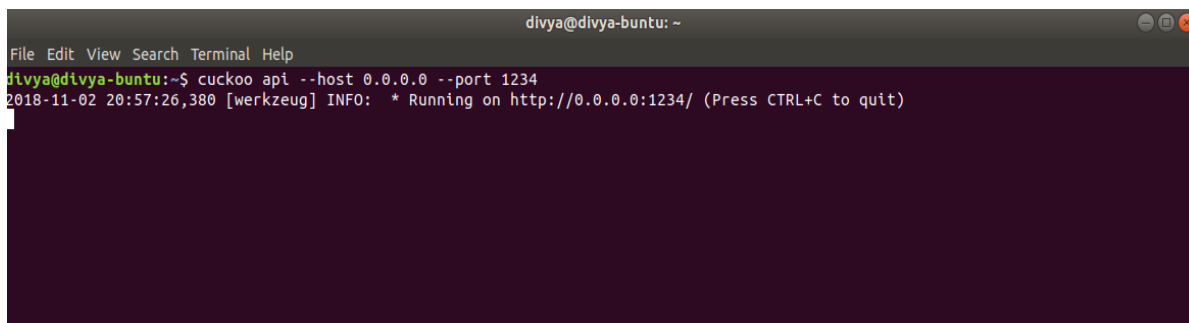
Samples can be submitted through REST API utility which is a simple and light weighted API. To submit samples in REST API, we need to start the API as shown below. The API runs on local host and binds the service on port 8090 by default.

A terminal window titled 'divya@divya-buntu: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The command 'cuckoo api' has been executed. The output shows a timestamp '2018-11-02 20:37:54,701 [werkzeug]' followed by an INFO message: '* Running on http://localhost:8090/ (Press CTRL+C to quit)'. The terminal background is dark purple.

```
divya@divya-buntu: ~  
File Edit View Search Terminal Help  
divya@divya-buntu:~$ cuckoo api  
2018-11-02 20:37:54,701 [werkzeug] INFO: * Running on http://localhost:8090/ (Press CTRL+C to quit)
```

FIGURE 26: INITIALIZING REST API INTERFACE

Alternatively, we can start the API on any customized host and service as shown below.

A terminal window titled 'divya@divya-buntu: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The command 'cuckoo api --host 0.0.0.0 --port 1234' has been executed. The output shows a timestamp '2018-11-02 20:57:26,380 [werkzeug]' followed by an INFO message: '* Running on http://0.0.0.0:1234/ (Press CTRL+C to quit)'. The terminal background is dark purple.

```
divya@divya-buntu: ~  
File Edit View Search Terminal Help  
divya@divya-buntu:~$ cuckoo api --host 0.0.0.0 --port 1234  
2018-11-02 20:57:26,380 [werkzeug] INFO: * Running on http://0.0.0.0:1234/ (Press CTRL+C to quit)
```

FIGURE 27: CUSTOMIZED REST API

We can submit a sample with simple python files and it will submit the suspicious samples for analysis. Here I saved a sample code in Rest.py file and the file is submitted through REST API. As soon as the file is submitted, cuckoo starts analysing the files.


```

divya@divya-buntu:~$ python /home/divya/files/python/rest.py
divya@divya-buntu:~$

divya@divya-buntu:~$ cuckoo api
2018-11-02 20:55:00,892 [werkzeug] INFO: * Running on http://localhost:8090/ (p
ress CTRL+C to quit)
2018-11-02 20:55:23,819 [werkzeug] INFO: 192.168.1.106 - - [02/Nov/2018 20:55:23
] "POST /tasks/create/file HTTP/1.1" 200 -

2018-11-02 20:54:52,601 [cuckoo.core.scheduler] INFO: Loaded 1 machine/s
2018-11-02 20:54:52,616 [cuckoo.core.scheduler] INFO: Waiting for analysis tasks.
2018-11-02 20:55:24,340 [cuckoo.core.scheduler] DEBUG: Processing task #1
2018-11-02 20:55:24,357 [cuckoo.core.scheduler] INFO: Starting analysis of FILE "temp_file_name" (task #1, options
)
2018-11-02 20:55:24,392 [cuckoo.core.scheduler] INFO: Task #1: acquired machine windows (label=windows)
2018-11-02 20:55:24,399 [cuckoo.auxiliary.mitm] INFO: Started mitm interception with PID 18510 (ip=192.168.56.1, p
=s00000).
2018-11-02 20:55:24,399 [cuckoo.core.plugins] DEBUG: Started auxiliary module: MITM
2018-11-02 20:55:24,405 [cuckoo.auxiliary.sniffer] INFO: Started sniffer with PID 18511 (interface=vboxnet0, hosts
.168.56.101)
2018-11-02 20:55:24,406 [cuckoo.core.plugins] DEBUG: Started auxiliary module: Sniffer
2018-11-02 20:55:24,442 [cuckoo.machinery.virtualbox] DEBUG: Starting vm windows
2018-11-02 20:55:24,541 [cuckoo.machinery.virtualbox] DEBUG: Restoring virtual machine windows to its current snap
t
2018-11-02 20:55:31,524 [cuckoo.core.guest] INFO: Starting analysis on guest (id=windows, ip=192.168.56.101)
2018-11-02 20:55:32,528 [cuckoo.core.guest] DEBUG: windows: not ready yet
2018-11-02 20:55:33,233 [cuckoo.core.guest] DEBUG: windows: not ready yet
2018-11-02 20:55:35,240 [cuckoo.core.guest] DEBUG: windows: not ready yet
2018-11-02 20:55:35,264 [cuckoo.core.guest] INFO: Guest is running Cuckoo Agent 0.8 (id=windows, ip=192.168.56.10
2018-11-02 20:55:35,288 [cuckoo.core.guest] DEBUG: Uploading analyzer to guest (id=windows, ip=192.168.56.101, no
relatest, size=3835175)
2018-11-02 20:55:35,597 [cuckoo.core.guest] DEBUG: windows: analysis still processing
2018-11-02 20:55:35,712 [cuckoo.core.resultserver] DEBUG: LogHandler for live analysis.log initialized.
2018-11-02 20:55:36,616 [cuckoo.core.guest] DEBUG: windows: analysis still processing
2018-11-02 20:55:37,641 [cuckoo.core.guest] DEBUG: windows: analysis still processing
2018-11-02 20:55:38,777 [cuckoo.core.guest] DEBUG: windows: analysis still processing
2018-11-02 20:55:39,784 [cuckoo.core.guest] DEBUG: windows: analysis still processing
2018-11-02 20:55:40,856 [cuckoo.core.guest] DEBUG: windows: analysis still processing
2018-11-02 20:55:41,999 [cuckoo.core.guest] DEBUG: windows: analysis still processing
2018-11-02 20:55:43,089 [cuckoo.core.guest] DEBUG: windows: analysis still processing
2018-11-02 20:55:44,170 [cuckoo.core.guest] DEBUG: windows: analysis still processing
2018-11-02 20:55:45,236 [cuckoo.core.guest] DEBUG: windows: analysis still processing
2018-11-02 20:55:46,306 [cuckoo.core.guest] DEBUG: windows: analysis still processing
2018-11-02 20:55:47,405 [cuckoo.core.guest] DEBUG: windows: analysis still processing
2018-11-02 20:55:48,475 [cuckoo.core.guest] DEBUG: windows: analysis still processing
2018-11-02 20:55:49,479 [cuckoo.core.guest] INFO: windows: analysis completed successfully

```

FIGURE 28: SUBMITTING SAMPLE USING REST API

The tasks submitted can be viewed in Web Interface as shown below.

```

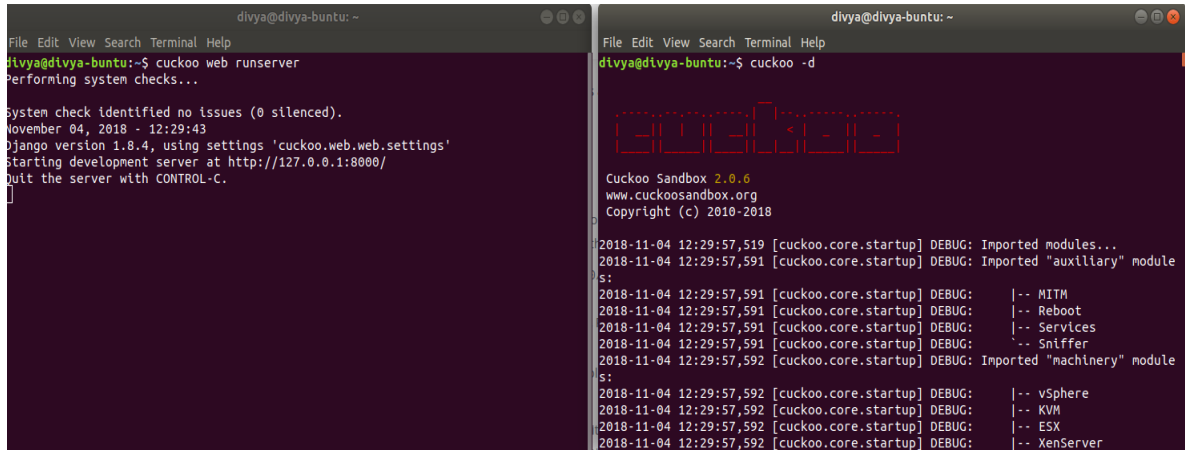
{
  "tasks": [
    {
      "added_on": "2018-11-02 20:21:44",
      "category": "file",
      "clock": "2018-11-02 20:21:44",
      "completed_on": null,
      "custom": null,
      "duration": -1,
      "enforce_timeout": false,
      "errors": [],
      "guest": {
        "id": 2,
        "label": "windows",
        "manager": "VirtualBox",
        "name": "Windows",
        "shutdown_on": null,
        "started_on": "2018-11-02 20:21:44",
        "status": "starting",
        "task_id": 2
      },
      "id": 2,
      "machine": null,
      "memory": false,
      "options": {},
      "owner": null,
      "package": "",
      "platform": null,
      "priority": 1,
      "processing": null,
      "route": "none",
      "sample": {
        "crc32": "30E872EE",
        "file_size": 23822,
        "file_type": "PNG image data, 1853 x 991, 8-bit/color RGBA, non-interlaced",
        "id": 2,
        "md5": "d213b210023ce76de451ccd46299a0bb",
        "sha1": "4ba879710b859b193c4b7c5146d01dd6cbd3582e",
        "sha256": "4517f64745c24070071cd13d3f26c94860fa0a20c5a37322cb1394e27a4c5bea",
        "sha512": "90ecb3adfd6804da1598f62b576c90b09776dfec4a66580e07523fee20127fb60e31921bd2322d6a0a8529f6e9551950b7914850bb2111606dc981c4926c97d",
        "ssdeep": "384:TKLBDC9yMp0s9BPDABh0L78t+hevXQu6Sbl7:BBUWYzS980gB48wwJQs7"
      },
      "sample_id": 2,
      "started_on": "2018-11-02 20:21:44",
      "status": "failed_analysis",
      "submit_id": null,
      "tags": [],
      "target": "/home/divya/output.png",
      "timeout": 0
    },
    {
      "added_on": "2018-11-02 20:22:43",
      "category": "file",
      "clock": "2018-11-02 20:22:43",
      "completed_on": null,
      "custom": null,
      "duration": -1,
      "enforce_timeout": false,
      "errors": [],
      "guest": {
        "id": 3,
        "label": "windows",
        "manager": "VirtualBox",
        "name": "Windows"
      },
      "id": 3,
      "machine": null,
      "memory": false,
      "options": {},
      "owner": null,
      "package": "",
      "platform": null,
      "priority": 1,
      "processing": null,
      "route": "none",
      "sample": {
        "crc32": "30E872EE",
        "file_size": 23822,
        "file_type": "PNG image data, 1853 x 991, 8-bit/color RGBA, non-interlaced",
        "id": 3,
        "md5": "d213b210023ce76de451ccd46299a0bb",
        "sha1": "4ba879710b859b193c4b7c5146d01dd6cbd3582e",
        "sha256": "4517f64745c24070071cd13d3f26c94860fa0a20c5a37322cb1394e27a4c5bea",
        "sha512": "90ecb3adfd6804da1598f62b576c90b09776dfec4a66580e07523fee20127fb60e31921bd2322d6a0a8529f6e9551950b7914850bb2111606dc981c4926c97d",
        "ssdeep": "384:TKLBDC9yMp0s9BPDABh0L78t+hevXQu6Sbl7:BBUWYzS980gB48wwJQs7"
      },
      "sample_id": 3,
      "started_on": "2018-11-02 20:22:43",
      "status": "starting",
      "submit_id": null,
      "tags": [],
      "target": "/home/divya/output.png",
      "timeout": 0
    }
  ]
}

```

FIGURE 29: TASKS SUBMITTED IN REST API

5.5. Analysis of Sample File in Cuckoo Sandbox.

- i. Start Cuckoo Daemon and Web interface to submit a file using Web interface



```
divya@divya-buntu: ~  
File Edit View Search Terminal Help  
divya@divya-buntu:~$ cuckoo web runserver  
Performing system checks...  
  
system check identified no issues (0 silenced).  
November 04, 2018 - 12:29:43  
Django version 1.8.4, using settings 'cuckoo.web.settings'  
Starting development server at http://127.0.0.1:8080/  
Quit the server with CONTROL-C.  
[1]
```

```
divya@divya-buntu: ~  
File Edit View Search Terminal Help  
divya@divya-buntu:~$ cuckoo -d  
  
Cuckoo Sandbox 2.0.6  
www.cuckoosandbox.org  
Copyright (c) 2010-2018  
  
2018-11-04 12:29:57,519 [cuckoo.core.startup] DEBUG: Imported modules...  
2018-11-04 12:29:57,591 [cuckoo.core.startup] DEBUG: Imported "auxiliary" module  
s:  
2018-11-04 12:29:57,591 [cuckoo.core.startup] DEBUG: |-- MITM  
2018-11-04 12:29:57,591 [cuckoo.core.startup] DEBUG: |-- Reboot  
2018-11-04 12:29:57,591 [cuckoo.core.startup] DEBUG: |-- Services  
2018-11-04 12:29:57,591 [cuckoo.core.startup] DEBUG: |-- Sniffer  
2018-11-04 12:29:57,592 [cuckoo.core.startup] DEBUG: Imported "machinery" module  
s:  
2018-11-04 12:29:57,592 [cuckoo.core.startup] DEBUG: |-- vSphere  
2018-11-04 12:29:57,592 [cuckoo.core.startup] DEBUG: |-- KVM  
2018-11-04 12:29:57,592 [cuckoo.core.startup] DEBUG: |-- ESX  
2018-11-04 12:29:57,592 [cuckoo.core.startup] DEBUG: |-- XenServer
```

FIGURE 30: INITIALISING CUCKOO

- ii. Submit a Test File that needs to be analysed in Cuckoo Sandbox.

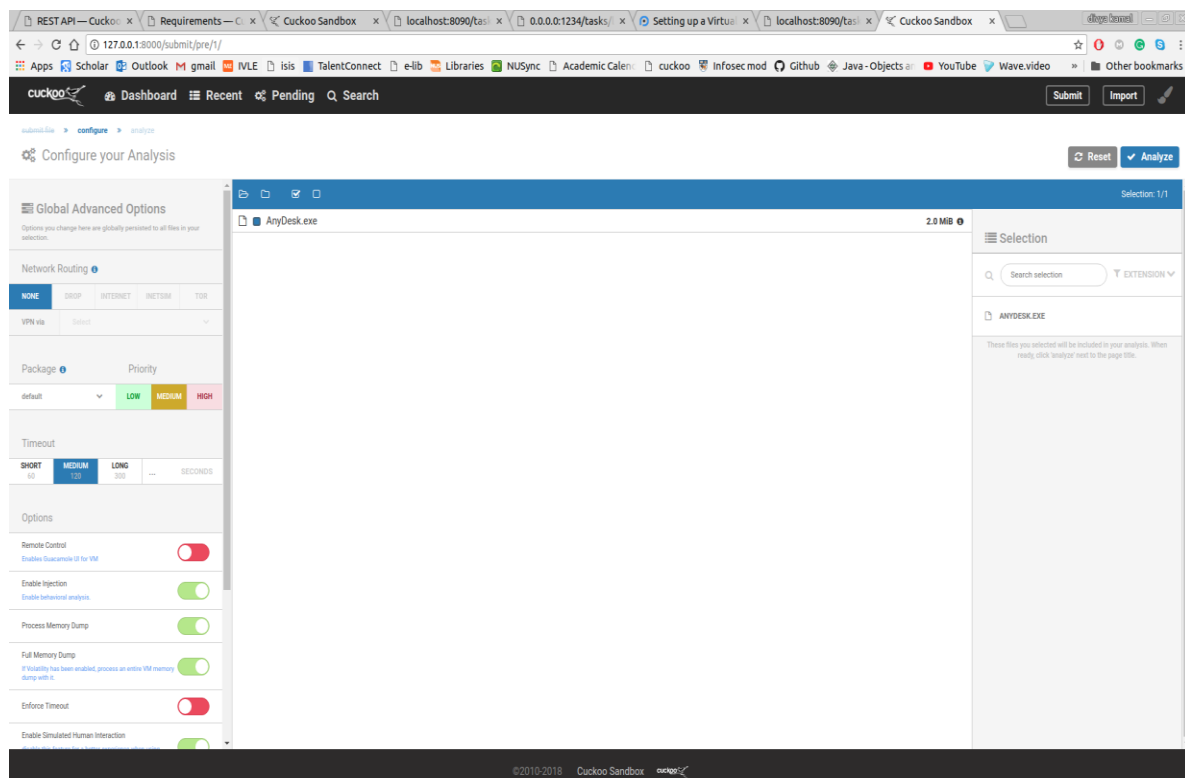


FIGURE 31: SUBMITTING SAMPLE FILE FOR ANALYSIS

- iii. As soon as we submit a sample file, Guest Machine snapshot state is restored. All the suspicious samples will be executed in the Guest Machine and the analysis will be stored in cuckoo database.

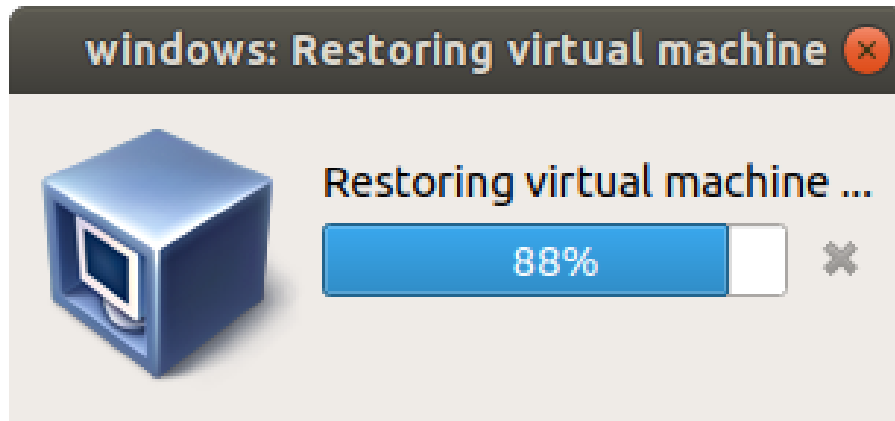


FIGURE 32: RESTORING VIRTUAL MACHINE

- iv. Agent is automatically started in the Guest Machine to allow communication between Host and Guest Machines.

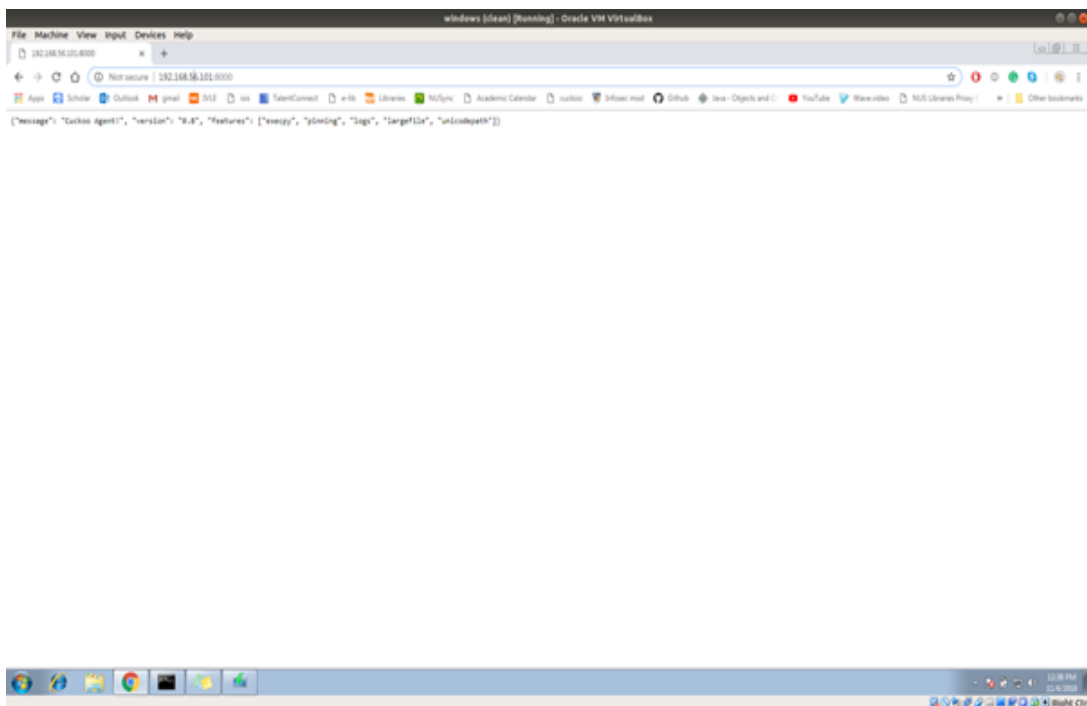


FIGURE 33: AGENT IS INITIALIZED

- v. Observe the execution of suspicious test file in Guest Machine. Sometimes, we may not be able to see the execution of the file as it may be done as a background job.

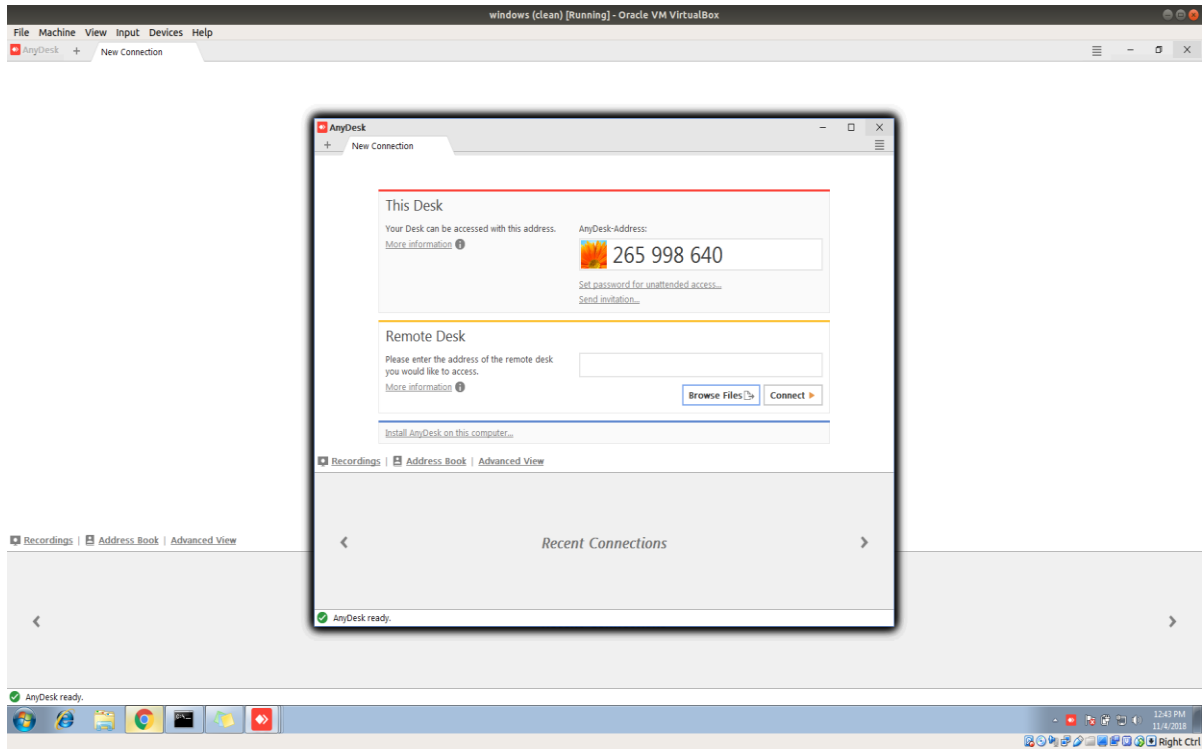


FIGURE 34: EXECUTION OF SAMPLE FILE IN GUEST MACHINE

6. ANALYSIS RESULTS IN CUCKOO SANDBOX

The analysis results output is store in the

\$CWD/.cuckoo/storage/analysis/task_id/reports/report.json.

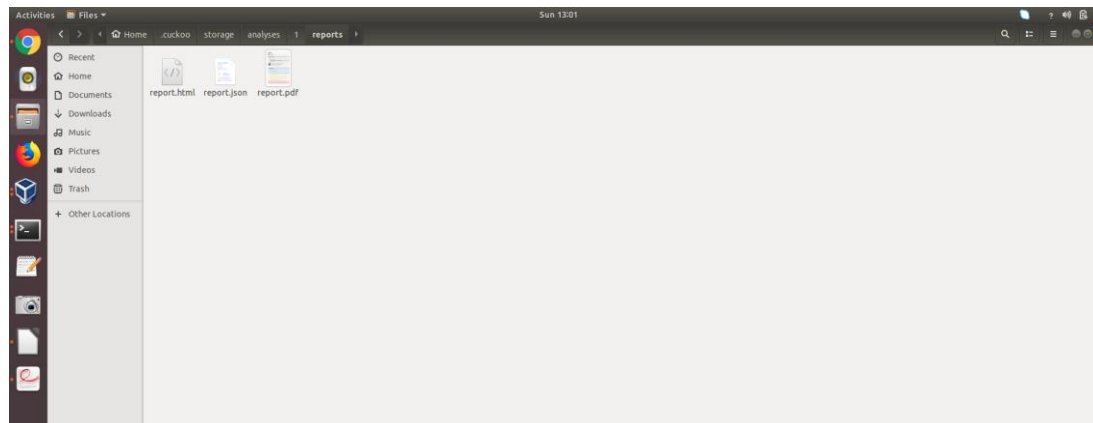


FIGURE 35: ANALYSIS RESULTS STORED IN HOST MACHINE

The results output can also be viewed in the Web interface.

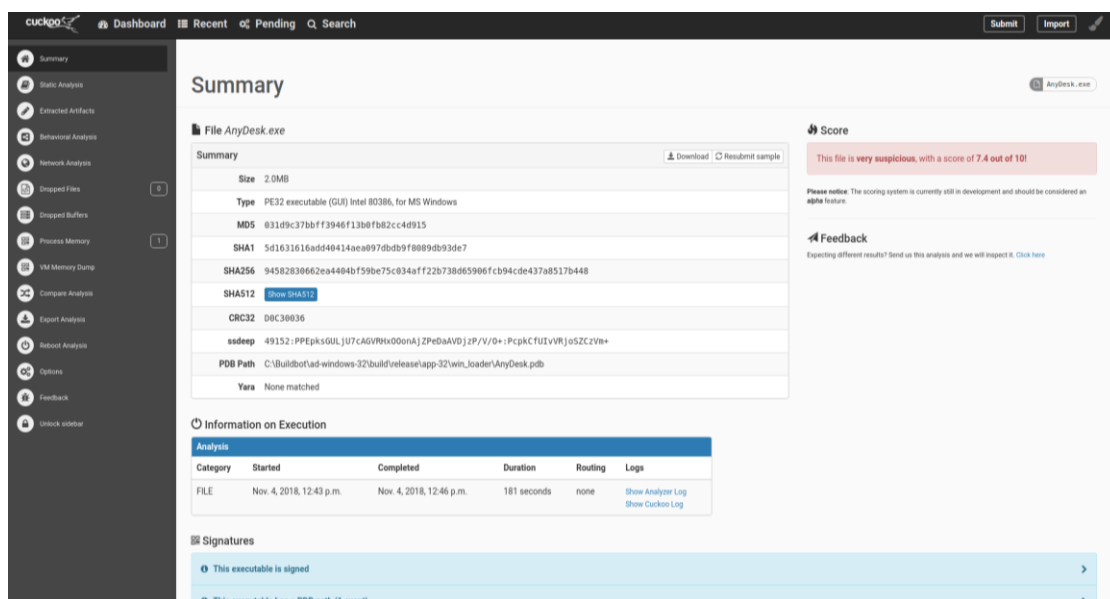


FIGURE 36: ANALYSIS RESULTS IN WEB UI

As Cuckoo Sandbox performed dynamic analysis (actual execution of the suspicious sample file), we can observe the test results from different modules along with the static analysis.

Static Analysis provides information such as signatures, certificate details, version and strings in the sample files.

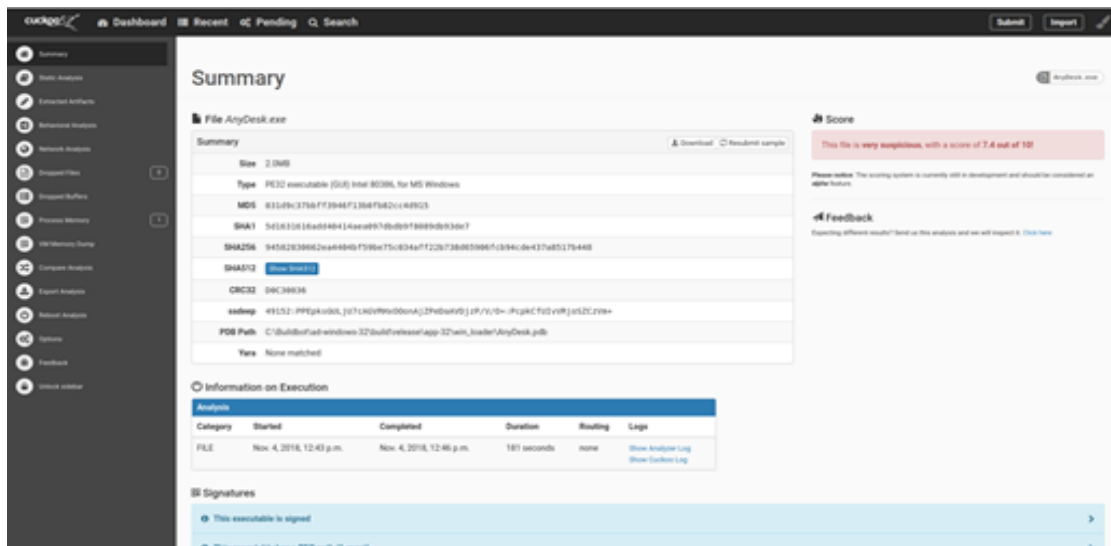


FIGURE 37: STATIC ANALYSIS IN WEB UI

Network Analysis provides information on all the Network addresses it contacted while executing the sample.

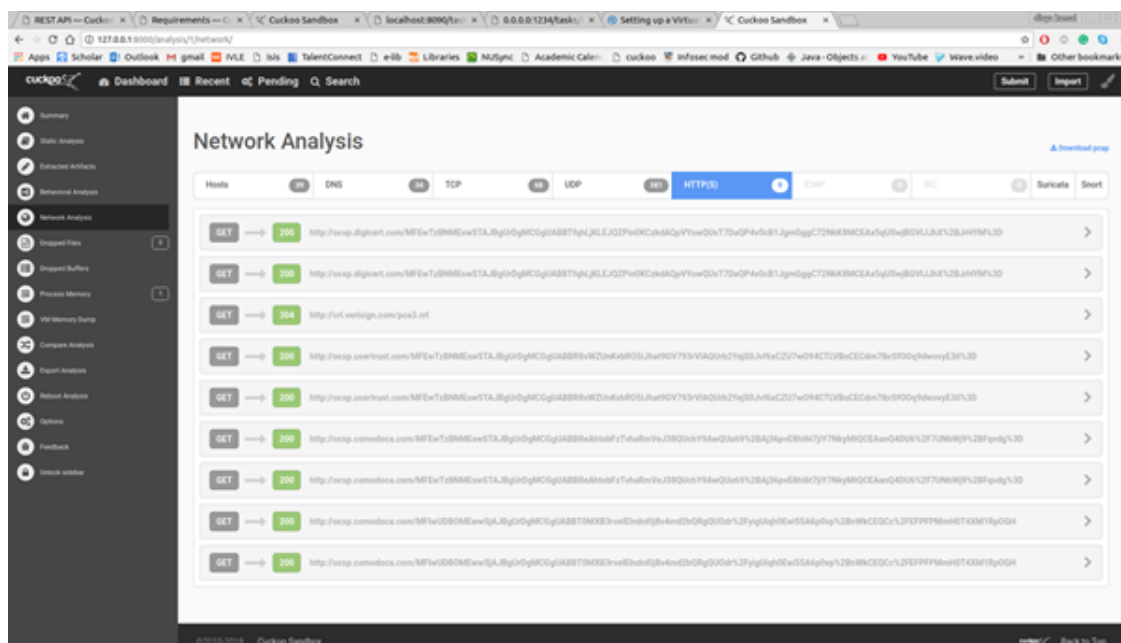


FIGURE 38: NETWORK ANALYSIS IN WEB UI

Process Memory provides information on the malware activities during the execution

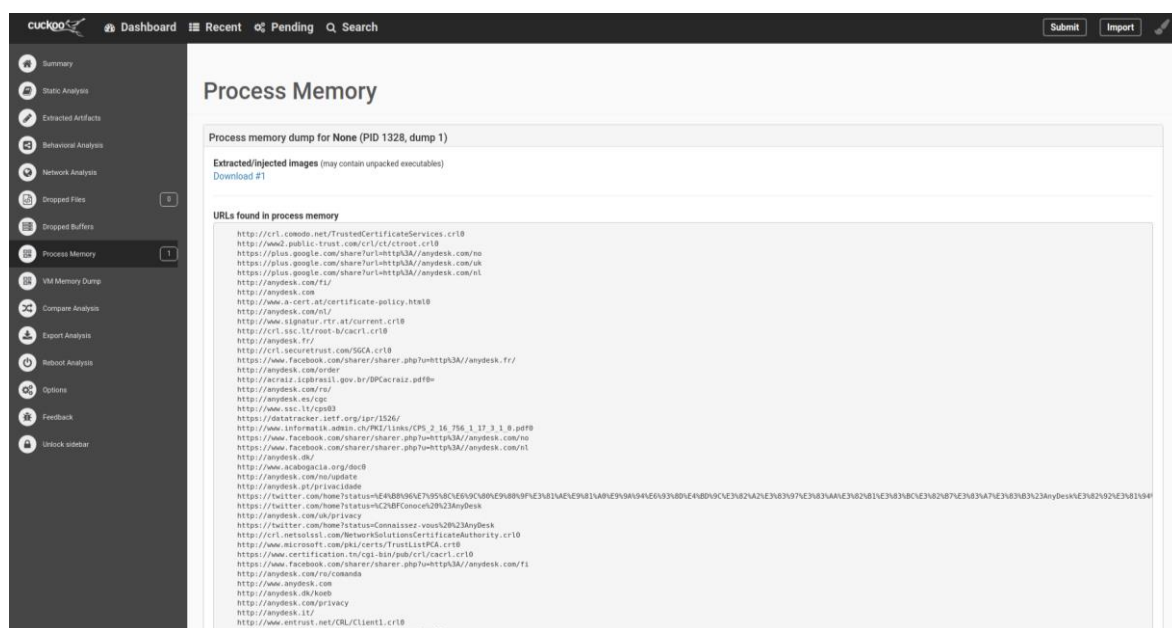


FIGURE 39: PROCESS MEMORY IN WEB UI

VM Memory Dump option can be enabled in Cuckoo to analyse the memory dump later.

The Memory dump gives information of all the modules, the file contacted while executing.

The screenshot shows the 'VM Memory Dump' section of the Cuckoo Sandbox web UI. It features a table with columns: Process List, Services, Kernel Modules, Device Tree, Code Injection, Timers, and Pixview. The 'Process List' column is expanded, showing a table with columns: Parent PID, PID, Name, Create Time, Exit Time, # Threads, # Handles, and Session ID. The table lists several processes, including System, smss.exe, csrss.exe, wininit.exe, winlogon.exe, services.exe, lsass.exe, and lsm.exe, along with their creation and exit times, thread counts, handle counts, and session IDs.

Parent PID	PID	Name	Create Time	Exit Time	# Threads	# Handles	Session ID
0	4	System	2018-10-04 06:39:30 UTC+0000	-	74	561	None
4	232	smss.exe	2018-10-04 06:39:30 UTC+0000	-	2	29	None
296	304	csrss.exe	2018-10-04 06:39:31 UTC+0000	-	9	440	0
296	340	wininit.exe	2018-10-04 06:39:31 UTC+0000	-	3	75	0
332	352	csrss.exe	2018-10-04 06:39:31 UTC+0000	-	9	884	1
332	392	winlogon.exe	2018-10-04 06:39:31 UTC+0000	-	3	108	1
340	436	services.exe	2018-10-04 06:39:31 UTC+0000	-	9	204	0
340	448	lsass.exe	2018-10-04 06:39:31 UTC+0000	-	6	776	0
340	456	lsm.exe	2018-10-04 06:39:31 UTC+0000	-	10	147	0

FIGURE 40: VOLATILITY ANALYSIS IN WEB UI

Compare Analysis Cuckoo provides option to compare two different analysis and identify the differences.

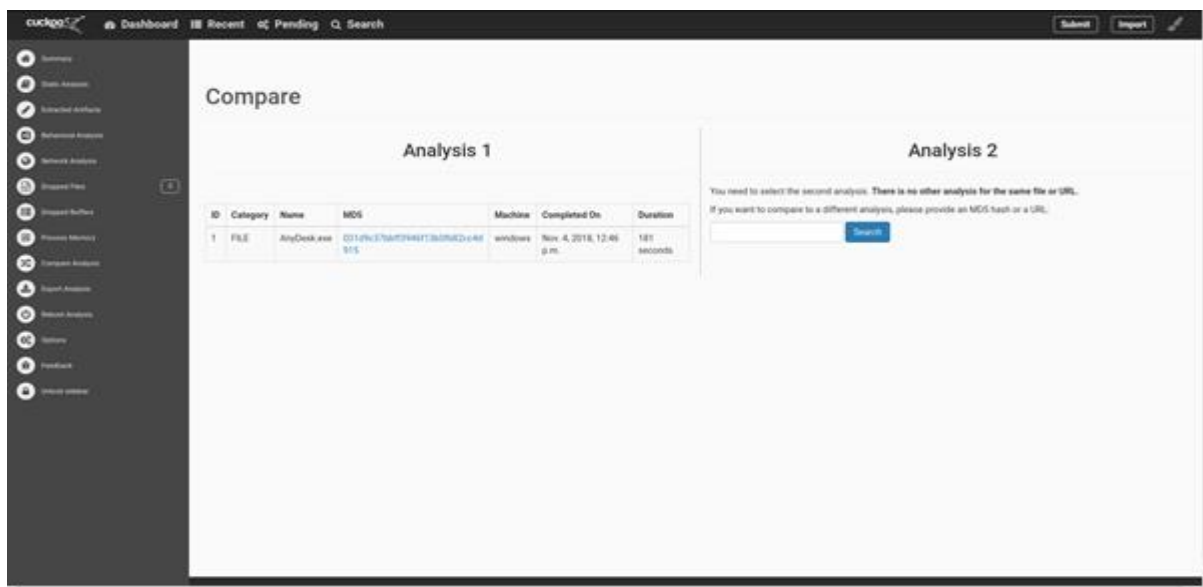


FIGURE 41: COMPARE ANALYSIS OPTION IN WEB UI

Export Analysis Cuckoo provides option to export the analysis and perform various other functionalities on the analysis.

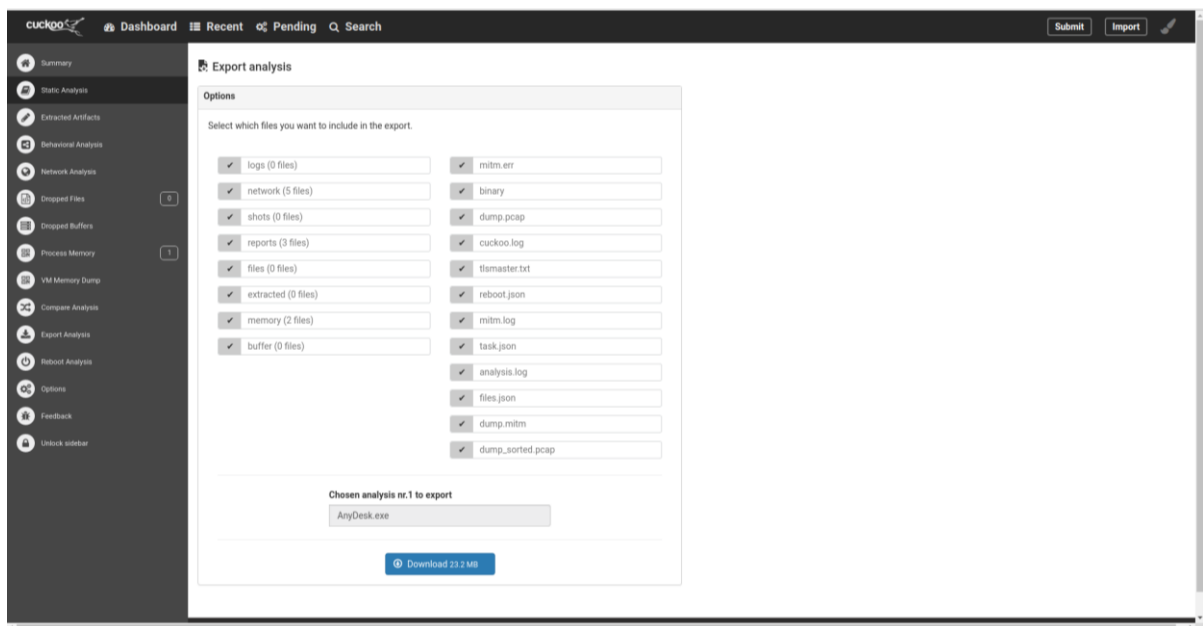


FIGURE 42: EXPORT ANALYSIS IN WEB UI

Reboot Analysis: This option provides a feature to perform analysis again. The complete analysis process will be rebooted if this option is clicked.

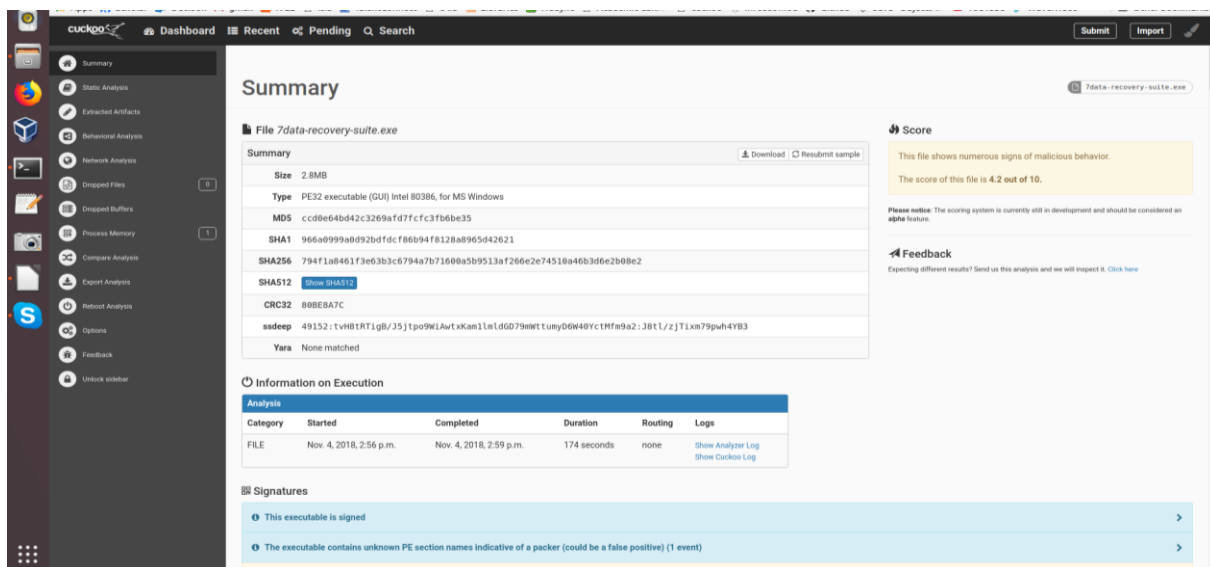


FIGURE 43:REBOOT ANALYSIS IN WEB UI

7. CLUSTERING THE ANALYSIS RESULTS INTO MALWARE FAMILIES

After analysing the samples in Cuckoo Sandbox, we want to cluster the resulting analyses into malware families. We are clustering based on the Network address the sample has contacted while executing the sample in the Guest Machine. We will first extract the interesting information and convert into Pandas Data Frame.

- i. Consider the output in JSON Format. Please find the output from `$CWD/.cuckoo/storage/analysis/task id/reports/report.json`

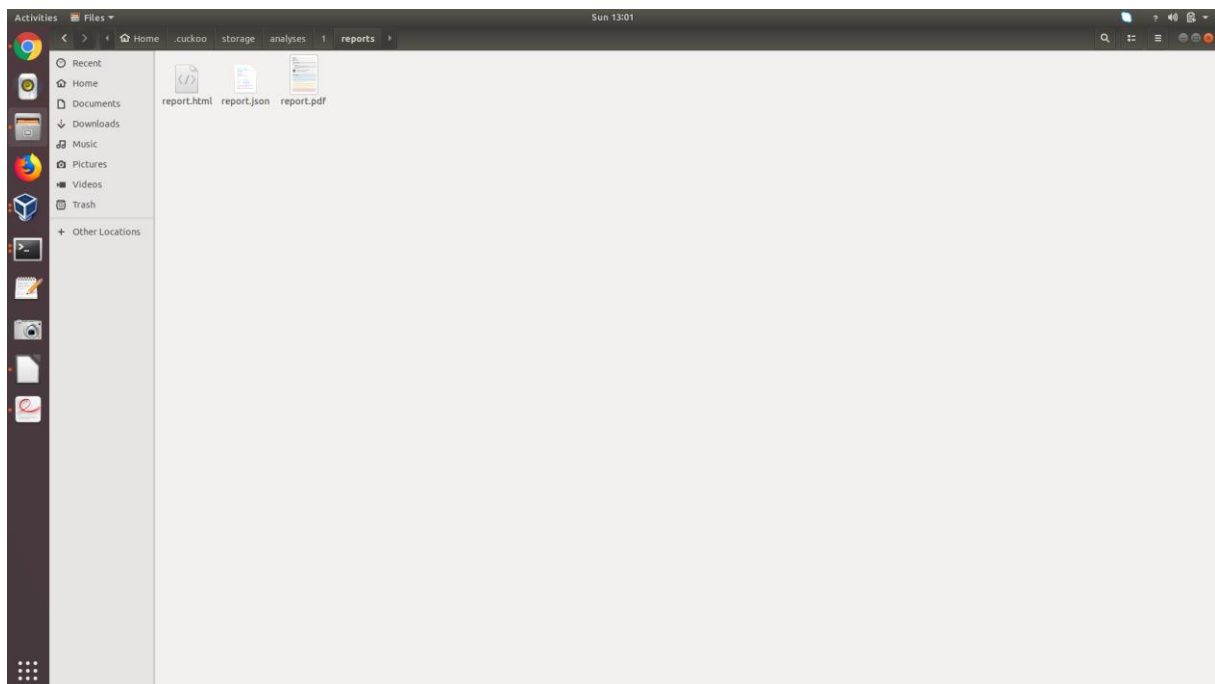


FIGURE 44: MALWARE ANALYSIS OUTPUT

- ii. As working with Large Unstructured data is difficult, explore the json and identify the interesting information. We want to identify the destination addresses it is contacting and cluster them based on it.

- iv. Read all the available data frames into a single csv file. Here, I analysed 32 suspicious files and all the 32 data frames of these suspicious files is copied into a single csv file.

The screenshot shows a LibreOffice Calc spreadsheet with a table of data. The table has 10 columns labeled A through J and 20 rows numbered 1 through 20. The data is organized into four groups of five rows each, each starting with a row number in the first column. The columns contain numerical values and text labels. The text labels 'north' and 'south' appear in columns D, E, F, and G. The numerical values are formatted with commas as thousands separators. The spreadsheet interface includes a menu bar at the top with options like File, Edit, View, Insert, Format, Share, Data, Tools, Windows, and Help. Below the menu bar is a toolbar with various icons for editing and formatting. The status bar at the bottom shows the current row and column as 20 and J.

	A	B	C	D	E	F	G	H	I	J
1	3	31042	308	301	31042	308	301	31042	308	301
2	3	31042	308	302	31042	308	302	31042	308	302
3	3	31042	308	303	31042	308	303	31042	308	303
4	3	31042	308	304	31042	308	304	31042	308	304
5	3	31042	308	305	31042	308	305	31042	308	305
6	3	31042	308	306	31042	308	306	31042	308	306
7	3	31042	308	307	31042	308	307	31042	308	307
8	3	31042	308	308	31042	308	308	31042	308	308
9	3	31042	308	309	31042	308	309	31042	308	309
10	3	31042	308	310	31042	308	310	31042	308	310
11	3	31042	308	311	31042	308	311	31042	308	311
12	3	31042	308	312	31042	308	312	31042	308	312
13	3	31042	308	313	31042	308	313	31042	308	313
14	3	31042	308	314	31042	308	314	31042	308	314
15	3	31042	308	315	31042	308	315	31042	308	315
16	3	31042	308	316	31042	308	316	31042	308	316
17	3	31042	308	317	31042	308	317	31042	308	317
18	3	31042	308	318	31042	308	318	31042	308	318
19	3	31042	308	319	31042	308	319	31042	308	319
20	3	31042	308	320	31042	308	320	31042	308	320

FIGURE 47: DATA FRAME IN CSV FILE

v. Convert to TFIDF matrix

To convert to tfidf matrix, we count the destination addresses cocurrence in the Data frame and convert it to Document term matrix (dtm) which is also called term frequency matrix. The shape of the matrix shows the number of rows and columns in the matrix.

```

File Edit View Search Terminal Help
@vaya@vaya-host1:~/flux/python$ python readcsv.py
(0, 213) 0.4673967382000043
(0, 177) 0.4673967382000043
(0, 433) 0.4673967382000043
(0, 31) 0.4673967382000043
(0, 440) 0.35413728748923206
(0, 253) 0.4673967382000043
(1, 177) 0.4673967382000043
(0, 433) 0.4673967382000043
(1, 31) 0.4673967382000043
(1, 31) 0.4673967382000043
(0, 440) 0.35413728748923206
(2, 440) 0.38853564604437276
(0, 180) 0.46871645426153206
(2, 333) 0.46871647438153206
(2, 263) 0.453339174653206
(0, 403) 0.33764373996109197
(2, 180) 0.234268664669547
(0, 440) 0.2862399623889911302
(2, 337) 0.232743588996469692
(0, 433) 0.232743588996469692
(0, 253) 0.4673967382000043
(0, 177) 0.4673967382000043
(0, 433) 0.4673967382000043
(0, 31) 0.4673967382000043
(0, 440) 0.4673967382000043
(0, 253) 0.4673967382000043
(0, 213) 0.4673967382000043
(2, 213) 0.46224942411628416
(2, 380) 0.46224942411628416
(2, 67) 0.223124712081610206
(2, 130) 0.223124712081610206
(2, 123) 0.2437515508744493611
(0, 180) 0.2437515508744493611
(0, 440) 0.3232338633899610906
(0, 180) 0.3232338633899610906
(0, 333) 0.583698219988881
(0, 333) 0.35342386033437264
(0, 67) 0.14453337381137709
(0, 440) 0.21286478991085138
(0, 155) 0.327866634762277318
(0, 440) 0.327866634762277318
(0, 440) 0.381313688746831547
(31, 31) 0.52399255887706676
(0, 180) 0.3568682548555692748
(31, 243) 0.37288888881929919
(31, 213) 0.19766593471188884
(31, 123) 0.19766593471188884
(2, 180) 0.2272897891232723
(31, 43) 0.24832193828658168
(31, 180) 0.24832193828658168
(31, 223) 0.24832193828658168
(31, 223) 0.24832193828658168
(31, 343) 0.24832193828658168
[done]
[32, 47]

```

Figure 48: TFIDF Matrix

vi. K- Means Clustering

“We use the tf-idf matrix to run a slew of clustering algorithms to better understand the hidden structure of the matrix. I initialized K- Means with a pre-determined number of clusters (I chose 4 clusters of Malware Families in my Project). Each observation is assigned to a cluster and next, the mean of the clustered observations is calculated and used as the new cluster centroid. Then, the observations are reassigned to clusters and centroids are recalculated in an iterative process [18].”

```
divya@divya-buntu: ~/files/python
File Edit View Search Terminal Help
divya@divya-buntu:~/files/python$ python readcsv.py
Shape of the matrix
(32, 47)
Below are the clusters
[0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 2, 0, 1, 1, 0, 2, 1, 0, 0, 3, 1, 2]
```

FIGURE 49: K-MEANS CLUSTERING

vii. Convert the dist matrix into a 2-dimensional array.

To convert into a 2 -dimensional matrix, I used matplotlib and multidimensional scaling to convert and visualise the output.

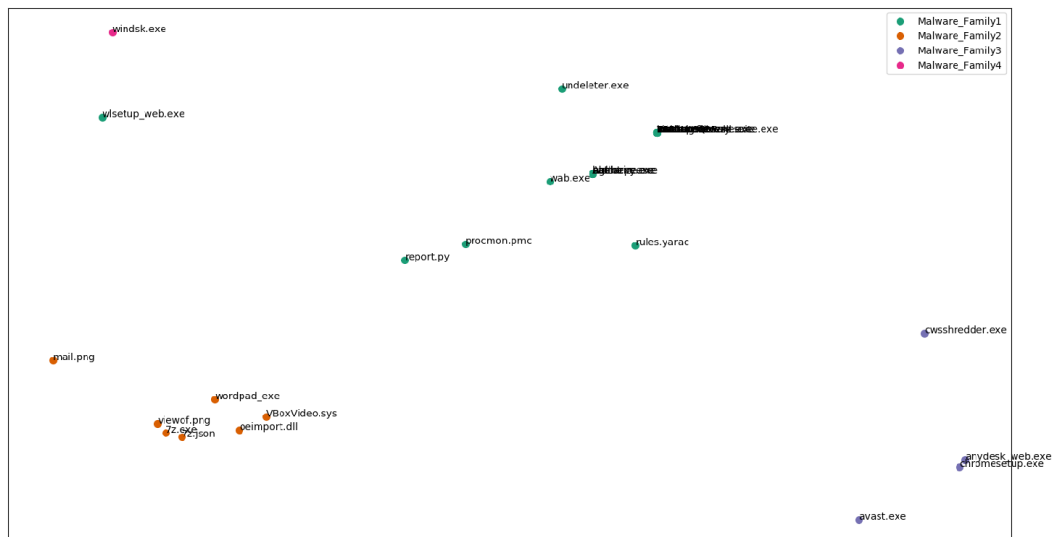


FIGURE 50: CLUSTERED MALWARE SAMPLES INTO MALWARE FAMILIES

8. CODE

a. Submitting the Sample File through REST API

```
import requests

REST_URL = "http://localhost:8090/tasks/create/file"

SAMPLE_FILE = "/home/divya/test.sh"

with open(SAMPLE_FILE, "rb") as sample:

    files = {"file": ("temp_file_name", sample)}

    r = requests.post(REST_URL, files=files)

# Add your code to error checking for r.status_code.

task_id = r.json()["task_id"]

# Add your code for error checking if task_id is None.
```

b. Extract the interesting information from the JSON output and save it in CSV format file.

```
import ijson

import json

import pandas as pd

import sys

#filepath

filename = "/home/divya/files/json/WordpadFilter_dll.json"

cols = []

#Read data from the file.

with open(filename, 'r') as f:

    objects_tcp = ijson.items(f, 'network.tcp.item')

    columns_tcp = list(objects_tcp)
```

```

with open(filename, 'r') as f1:
    objects_udp = ijson.items(f1, 'network.udp.item')
    columns_udp = list(objects_udp)
    #print columns_udp
with open(filename, 'r') as f1:
    objects_http = ijson.items(f1, 'network.http_ex.item')
    columns_http = list(objects_http)
#Merging all the columns
columns = columns_tcp + columns_udp + columns_http
#print(cols[0])
#extract the columns with destination address.
column_names = [col["dst"] for col in columns]
#Save the output
sys.stdout = open('/home/divya/files/csv/WordpadFilter_dll.csv', 'w')
print column_names

```

c. Convert the interesting information into Pandas Data Frame by reading all the interesting Information from the CSV files.

```

from sklearn.feature_extraction.text import TfidfVectorizer
import glob
from pandas import *
import os
import numpy as np
import sys
path = "/home/divya/files/csv"
files = glob.glob(os.path.join(path, "*.csv"))

```

```

print files
def reader(f):
    d = read_csv(f,header=None)
    d.columns = range(d.shape[1])
    return d
df = concat([reader(f) for f in files]) # Rev =df
sys.stdout = open('/home/divya/files/json/dataframe.csv', 'w')
#print(type(df))
print df

```

d. Convert the Dataframe into tfidf matrix. Cluster the resulting analyses from Cuckoo. Plot the observations into a 2 – dimensional visualizing plot.

```

from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.cluster import KMeans
from sklearn.externals import joblib
from sklearn.manifold import MDS
import glob
import pandas as pd
import os
import numpy as np
import sys
import matplotlib.pyplot as plt
import matplotlib as mpl
df = pd.DataFrame([])

```



```

Reviews = np.loadtxt(fname = "/home/divya/files/python/dataframe.csv",
dtype = 'str')
df['Reviews']=[" ".join(review) for review in Reviews]
dfn=df['Reviews']
#dfn = np.array([])
#define vectorizer parameters
tfidf_vectorizer = TfidfVectorizer()
tfidf_matrix = tfidf_vectorizer.fit_transform(dfn) #fit the vectorizer to
synopses
#print(tfidf_matrix)
print "Shape of the matrix"
print(tfidf_matrix.shape)
terms = tfidf_vectorizer.get_feature_names()
#print terms
dist = 1 - cosine_similarity(tfidf_matrix)
#print dist
num_clusters = 4
km = KMeans(n_clusters=num_clusters)
km.fit(tfidf_matrix)
clusters = km.labels_.tolist()
print "Below are the clusters"
print clusters
titles      =      ['ir053.msi',      'Firefox.exe',      'viewof.png',
'wabmig.exe','wab.exe',      'oeimport.dll',      '7z.json',
'rules.yarac','monitor.dll',      'undeleter.exe',      'battoexe.exe',

```

```

'samurize.exe', 'sunbelt-firewall.exe', 'VBoxWHQLFake.exe', 'agent.py',
'wlsetup_web.exe', 'abalarm.exe', 'rcsetup153.exe', 'Procmon.exe',
'avast.exe', 'chromesetup.exe', 'procmon.pmc', '7z.exe', 'mail.png',
'report.py', 'cwsshredder.exe', 'VBoxVideo.sys', '7data-recovery-
suite.exe', 'A.dll', 'windsk.exe', 'wordpad_exe', 'anydesk_web.exe']
#print title[:3]

films = { 'title': titles }

frame = pd.DataFrame(films, index = [clusters] , columns = ['title'])

# convert two components as we're plotting points in a two-dimensional
plane

# "precomputed" because we provide a distance matrix
# we will also specify `random_state` so the plot is reproducible.

MDS()

mds = MDS(n_components=2, dissimilarity="precomputed", random_state=1)
pos = mds.fit_transform(dist) # shape (n_components, n_samples)
xs, ys = pos[:, 0], pos[:, 1]

#print pos

#print ()

#print ()

cluster_colors = {0: '#1b9e77', 1: '#d95f02', 2: '#7570b3', 3: '#e7298a'}

#set up cluster names using a dict

cluster_names = {0: 'Malware_Family1',
                  1: 'Malware_Family2',
                  2: 'Malware_Family3',
                  3: 'Malware_Family4'}

```

```

#title = ['T1', 'T2', 'T3', 'T4', 'T5', 'T6', 'T7', 'T8']
#print title[:3]

#some ipython magic to show the matplotlib plots inline
#matplotlib inline

#create data frame that has the result of the MDS plus the cluster
numbers and titles

df = pd.DataFrame(dict(x=xs, y=ys, label=clusters, title=titles))

#print(df)  ##### PRINT THIS

#group by cluster

groups = df.groupby('label')

# set up plot

fig, ax = plt.subplots(figsize=(16,9)) # set size

ax.margins(0.05) # Optional, just adds 5% padding to the autoscaling

#iterate through groups to layer the plot

#note that I use the cluster_name and cluster_color dicts with the 'name'
lookup to return the appropriate color/label

for name, group in groups:

    ax.plot(group.x, group.y, marker='o', linestyle='', ms=8,

            label=cluster_names[name], color=cluster_colors[name],

            mec='none')

ax.set_aspect('auto')

ax.tick_params(\

    axis= 'x',          # changes apply to the x-axis

    which='both',      # both major and minor ticks are affected

    bottom='off',       # ticks along the bottom edge are off

```

```

        top='off',          # ticks along the top edge are off
        labelbottom='off')
ax.tick_params(\
    axis= 'y',             # changes apply to the y-axis
    which='both',          # both major and minor ticks are affected
    left='off',            # ticks along the bottom edge are off
    top='off',             # ticks along the top edge are off
    labelleft='off')

ax.legend(numpoints=1) #show legend with only 1 point
#add label in x,y position with the label as the film title
for i in range(len(df)):
    ax.text(df.ix[i]['x'], df.ix[i]['y'], df.ix[i]['title'], size=10)
plt.show() #show the plot
#uncomment the below to save the plot if need be
plt.savefig('clusters_op.png', dpi=200)

```

9. BIBLIOGRAPHY

1. 2018 Trustwave Global Security Report
2. <https://researchcenter.paloaltonetworks.com/2018/07/unit42-malware-team-malspam-pushing-emotet-trickbot/>
3. <https://blog.malwarebytes.com/detections/trojan-kovter/>

4. <https://www.trendmicro.com/vinfo/us/security/news/cybercrime-and-digital-threats/ransomware-recap-locky-returns-cerber-evolves-anew>
5. <http://www.phishing.org/what-is-phishing>
6. <https://www.trustwave.com/Resources/Trustwave-Blog/Here-is-an-Email-Thread-of-an-Actual-CEO-Fraud-Attack/>
7. [https://en.wikipedia.org/wiki/Exploit_\(computer_security\)](https://en.wikipedia.org/wiki/Exploit_(computer_security))
8. <https://en.wikipedia.org/wiki/Cryptocurrency>
9. https://ac-els-cdn-com.libproxy1.nus.edu.sg/S1361372318300642/1-s2.0-S1361372318300642-main.pdf?_tid=3e213831-faa5-4af0-9f69-b4bd6bb5a68d&acdnat=1540820853_23adc3ccb3f1b59a0010d8647c4f6514
10. <https://www.sciencedirect.com/science/article/pii/S016740481830004X>
11. <https://en.wikipedia.org/wiki/Malware>
12. http://bb2sz3ek3z.search.serialssolutions.com/?ctx_ver=Z39.88-2004&ctx_enc=info%3Aofi%2Fenc%3AUTF-8&rft_id=info%3Asid%2Fsummon.serialssolutions.com&rft_val_fmt=info%3Aofi%2Ffmt%3Akev%3Amtx%3Ajournal&rft.genre=article&rft.atitle=Packer+Detection+for+Multi-Layer+Executables+Using+Entropy+Analysis&rft.au=Munkhbayar+Bat-Erdene&rft.au=Taebeom+Kim&rft.au=Hyundo+Park&rft.au=Heejo+Lee&rft.date=2017-01-01&rft.pub=MDPI+AG&rft.eissn=1099-4300&rft.volume=19&rft.issue=3&rft.spage=125&rft_id=info:doi/10.3390%2Fe19030125¶mdict=en-US

13. [https://en.wikipedia.org/wiki/Sandbox_\(computer_security\)](https://en.wikipedia.org/wiki/Sandbox_(computer_security))
14. <https://cuckoo.sh/docs/>
15. <https://cuckoo.sh/docs/introduction/what.html>
16. <https://pydeep.readthedocs.io/en/latest/>
17. <https://github.com/volatilityfoundation/volatility/wiki>
18. <http://brandonrose.org/clustering>

