

# Early Detection of Ransomware Exploits Using Snort

Raaghav Mathur<sup>1</sup>, Ashish Khanna<sup>2</sup>, Abhijeet Singh<sup>3</sup>

Maharaja Agrasen Institute of Technology (New Delhi)<sup>1</sup>, Maharaja Agrasen Institute of Technology (New Delhi)<sup>2</sup>, Lucideus (New Delhi)<sup>3</sup>

raaghavmathur1998@gmail.com<sup>1</sup>, ashishk746@gmail.com<sup>2</sup>, singhgleader@gmail.com<sup>3</sup>

**Abstract** - We live in a fast-evolving internet age where opportunities are limitless, but so are the risks and chances of security breaches. In 2017, 'The Shadow Brokers', a self-proclaimed group of hackers, released several hacking tools and zero day exploits from the intelligence arsenal of USA's National Security Agency (NSA). Chief amongst these exploits were - Eternal Blue, Eternal Romance, Eternal Synergy and Eternal Champion. These, along with Eternal Red, were collectively titled the 'Eternal Series Exploits'. These exploits were subsequently used to make potential cyber weapons like 'WannaCry', 'Petya' and 'NotPetya' ransomware, which affected lakhs of computers worldwide and exposed major vulnerabilities in Microsoft Windows. In this age, where hackers constantly devise new methods to infiltrate systems, it is imperative to safeguard and strengthen network security by anticipating data breaches and providing protection from potential hazards. Deploying simple firewalls can no longer ensure adequate security. The need of the hour is an intrusion detection system like Snort, which analyses network traffic to detect potentially malicious activity and issues alerts. Such a system provides an additional defence mechanism and is an indispensable part of any security framework. Through this project, we aim to configure new Snort rules to successfully detect the Eternal exploits (and other similar exploits) in real time, so that no user data is compromised.

**Index Terms** - Snort, Wireshark, Ransomware, Vulnerability, Exploit, Intrusion Detection System

## I. INTRODUCTION

Technological advancement, especially in computer networking, has totally changed the way of communication [1]. Creating, sharing, managing and storing information has become easier and faster. However, it has also made systems vulnerable to attacks by people with malicious intent. Vulnerabilities like injection flaws, sensitive data exposure, broken access control, etc have been long used by hackers for gaining limitless access to computer systems, servers and applications.

However, the threat landscape has changed drastically in the past decade. Cyber attacks are becoming more innovative day by day with attackers resorting to the use of polymorphic malware to evade state of the art detection mechanisms.

Most malwares require certain conditions to initiate actions in the target system [2]. Ransomware is one such type of malware. The WannaCry ransomware spread across 150 countries [3] and affected over 3,00,000 systems worldwide using Eternal Blue. The WannaCry and Petya ransomware attacks resulted in economic losses worth several billion

dollars [4]. The impact could have been even greater if the

attacks had targeted sensitive infrastructure such as aircraft systems, nuclear power plants, military systems, etc. Therefore, all organizations must have pre-emptive cyber defence systems to thwart such attacks.

## II. RELATED WORK AND BACKGROUND

### A. Ransomware

The primary aim of malware creators has changed from simply stealing information to hijacking systems and confidential files so as to earn financial profits [5]. Ransomware is a rapidly growing hazard in the cyber world [6]. The broad classification of ransomware includes the following types [7] -

a) *Non-encrypting* - It simply restricts access by preventing the operating system from loading. As the name suggests, no encryption techniques are used in this type of ransomware.

b) *Encrypting* - It is the most powerful type of ransomware as it encrypts all files and folders present on the target system. Data recovery is almost impossible without the decryption key. The attackers demand ransom payments in cryptocurrencies like Bitcoin, Ethereum and Ripple which makes it tough to trace the culprits. Advanced ransomware like 'WannaCry', 'CryptoLocker', 'Petya' and 'NotPetya' fall under this category.

c) *Leakware* - It threatens the victim about publishing his/her confidential information (unless a ransom is paid) rather than denying access to it.

d) *Mobile Ransomware* - It targets the mobile operating system, typically Android. It usually uses different forms of clickjacking to cause the user to give it administrator privileges.

UNVEIL [8] is an automated system for ransomware detection. It detects ransomware interaction with user data by tracking any changes in the system's desktop. It successfully identifies evasive ransomware, which goes undetected by most antivirus software. Our work identifies generic malware features for discovering exploits used to create ransomware like 'WannaCry'.

### B. Intrusion Detection Systems

The term 'intrusion detection' refers to the process of identifying unauthorized or illegitimate access to systems, by insiders or external offenders [9]. In the last few years, intrusion detection systems have become the first line of defence for preventing security violations [10]. Intrusion detection systems can be classified on the basis of analysed activity (network based and host based) and detection

technique (signature based and anomaly based) [11]. Using signature based systems ensures high detection rates for known attacks [12].

Network intrusion detection systems can detect and distinguish between exploits using different string or pattern matching techniques [13]. Snort is a popular intrusion detection and prevention system [14], used for safeguarding networks. It was first introduced in 1998 by Martin Roesch. It performs real traffic analysis to detect a variety of probes and attacks, though there may be some false positives [15]. Snort can work efficiently on a wide variety of systems and hardware [16].

Snort can be configured to run in the 3 modes - sniffing (read and display network packets), packet logging (create a log for the network packets) and network intrusion detection (perform real time network traffic analysis against user-defined rules).

Apart from Snort, the other popular intrusion detection tools available freely are Suricata and Bro [17].

### III. PROPOSED MODEL FOR EXPLOIT DETECTION

#### A. Assumptions

- a) There is no loss of the exploit's data packets.
- b) There is no manipulation of the exploit's data packets by Wireshark.

#### B. Flow Diagram

The approach used for configuration of the new Snort rules is represented in the diagram shown in Fig 1.

#### C. Explanation of the Flow Diagram

- 1) The attacker runs exploits and sends payloads through the Metasploit framework.
- 2) Wireshark (a network protocol analyser) intercepts the network traffic and captures all the data packets.
- 3) The data packets are filtered on the basis of the source address, destination address and protocol.
- 4) The filtered data packets are used for exploit and payload inspection. We try to identify data (hex values, keywords, patterns, byte sequences, etc) which can be used for exploit identification and detection. The raw exploit code and actual working of the exploit is also analysed for the same purpose.
- 5) The extracted data and the inbuilt Snort rule options are used for configuration of new Snort rules.
- 6) The configured Snort rules are added to the rule path in the Snort configuration file.
- 7) Snort is run to analyse network traffic and detect the exploits in real time against the defined rule set.

### IV. ENVIRONMENT VARIABLES

#### A. VMware Setup

In this project, we used VMware workstation, which is a platform virtualization software. It allows users to set-up and run multiple virtual machines simultaneously, along with the host machine. For the list of virtual machines used in the research project, refer to table 1.

#### B. Snort Installation and Configuration

Snort can be installed in 2 ways -

- a) By using the command 'sudo apt-get install snort'.

b) By manually compiling the packages from the source. The following steps briefly explain the manual configuration -

- a) Install Snort dependencies and packages - Libnet, Libpcap, PCRE, Libmysqlclient, Iptables, Libnetfilter-queue and Libdnet

b) Download and install Snort from [www.snort.org](http://www.snort.org)

c) Edit the Snort configuration file - Specify the network addresses to protect and add, modify or delete the rule paths

d) Check if Snort installation was successful by using the command 'snort -v'. You will see the version of Snort installed if there are no errors in the installation process.

### V. BRIEF ANALYSIS OF THE ETERNAL EXPLOITS

Table 2 contains exploit details such as Common Vulnerabilities and Exposures (CVE) number, vulnerabilities exploited and systems affected by the exploits. Table 3 contains a list of ransomware and malware associated with each exploit.

### VI. SNORT RULES CONFIGURATION

#### A. Snort Inbuilt Rule Options

a) *alert* - This rule option informs the user about any malicious network activity, based on the configured Snort rules.

b) *any* - Users must specify the source IP address and port number as well as the destination IP address and port number, to help Snort in attack detection. By source and destination, we mean the origin and target of the exploit respectively. However, users who are unsure about these, can use this rule option as an alternative.

c) *msg* - The 'msg' rule option prints the message specified in the Snort rule on the console to inform the user about malicious activity, upon successful detection of the attack. The message is usually a simple text string.

d) *content* - This rule option allows users to search for only specific content in the data packets such as a simple string or complex binary data. Hexadecimal representation of binary data must be enclosed within pipe (|) characters. The matching is case sensitive by default. Multiple 'content' options in a single Snort rule helps in ensuring lesser number of false positives.

e) *classtype* - The 'classtype' rule option is used to classify each Snort rule as detecting a more general type of attack. Snort comprises of several inbuilt attack classes (shellcode-detect, suspicious-login, attempted-admin, attempted-dos, etc) to categorize known attacks.

f) *priority* - Users can assign a severity level to each Snort rule with the help of this rule option. Snort rules configured to detect critical attacks usually have a higher priority rating as compared to other rules. The 'priority' rule option is generally used with the 'classtype' rule option.

g) *reference* - This rule option is not a part of Snort's attack detection and prevention machinery. It only provides extra information about the exploit such as the CVE number, a web link explaining the exploit working, etc. This information can be linked to the Snort rule designed for detection of that exploit.

h) *sid* - This rule option specifies a unique number for

recognizing each rule.

i) *rev* - The ‘rev’ rule option denotes how many times a particular rule has been revised. Revision is necessary to update and refine the signatures used for attack detection. This rule option is generally used with the ‘sid’ rule option.

*B. Extracting data for Snort rules*

Table 4 contains the data extracted for configuration of rules and exploit identification. The data extracted is closely related to the working of the exploit. Table 5 explains the significance of data extracted for each exploit.

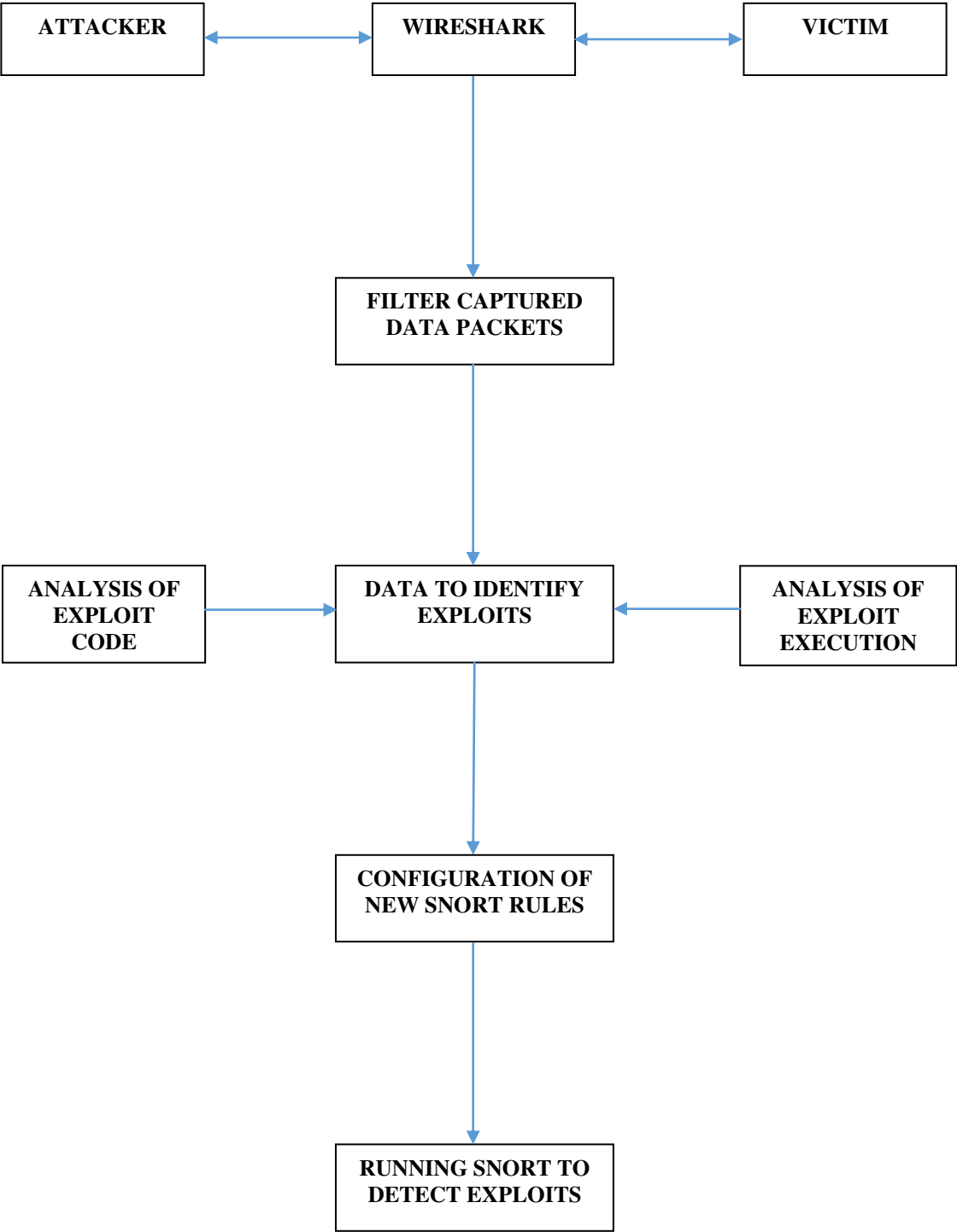


Fig 1 - Proposed Model

Table I - Virtual machines used and their purpose

Virtual Machines Used	Purpose
Ubuntu 16.04.6 LTS	This machine was used for running Snort.
Kali Linux 2018 (Version 1 and 4)	Kali Linux supports the Metasploit framework, which is a widely known vulnerability assessment and penetration testing framework. This framework was used for exploit execution on the target systems and for sending payloads.
Windows XP (32 bit) Windows Vista Business (64 bit) Windows 7 (32 bit and 64 bit) Windows 8.1 Pro (64 bit) Windows 10 (64 bit)	These machines were used as target systems for three Eternal exploits - Eternal Blue, Romance and Champion.
Kali Linux 2016 (Version 1)	We installed a vulnerable version of the Samba server on this machine and used it as a target system for Eternal Red.

Table II - Exploit Details

Exploits	CVE Number	Vulnerability	Systems Affected
Eternal Blue	CVE-2017-0144	The implementation of the Server Message Block (SMB) protocol in Microsoft Windows can be exploited through the processing of File Extended Attributes (FEA).	<u>Operating Systems</u> - Windows (XP, Vista, 7, 8, 8.1, 8.1 Pro and 10) <u>Servers</u> - Windows Servers (2003, 2008, 2012, 2016)
Eternal Champion	CVE-2017-0146	It takes advantage of SMB's improper handling of transactions. The vulnerability may be exploited through an information leak or a Remote Code Execution (RCE) attack.	<u>Operating Systems</u> - Windows (XP, Vista, 7, 8, 8.1) <u>Servers</u> - Windows Servers (2008 and 2012)
Eternal Romance	CVE-2017-0143	It is a SMB Version 1 exploit which works quite similarly to Eternal Blue. It tries to access different named pipes in the target system to gain administrative privileges.	<u>Operating Systems</u> - Windows (XP, Vista, 7, 8, 8.1, and 10) <u>Servers</u> - Windows Servers (2003, 2008, 2012 and 2016)
Eternal Red	CVE-2017-7494	The implementation of the Microsoft Remote Procedure Call (MSRPC) protocol inside the Samba service allows authenticated users to upload a malicious library to a shared directory having write permissions. This is followed by RCE attacks through named pipes.	Following versions of Samba were affected - version 3 (after 3.5.0), version 4 (before 4.4.14), version 4.5 (before 4.5.10) and version 4.6 (before 4.6.4)

Table III - Ransomware and malware associated with the exploits

Exploit	Associated Ransomware and Malware
Eternal Blue	WannaCry, Petya, NotPetya, Nyetya, Retefe, EternalPetya
Eternal Champion	Eternal Rocks
Eternal Romance	Bad Rabbit, Nyetya, EternalRocks, Eternal Petya
Eternal Red	None

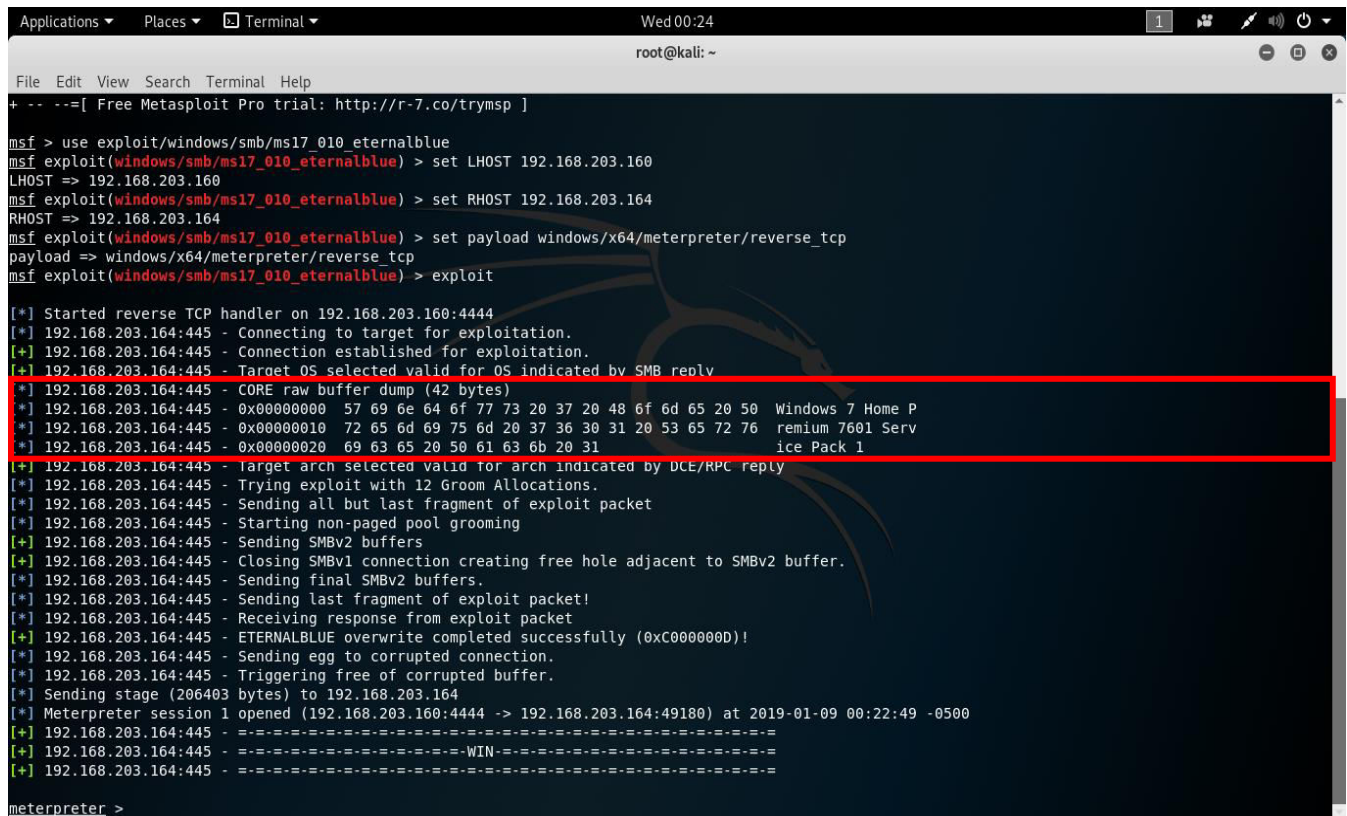
Table IV - Data extraction for configuration of Snort rules

Exploit	Type of data used for exploit identification - Keywords, Patterns, Hex Characters, etc	Data used for exploit identification
Eternal Blue	Hex Characters, Keywords	[57 69 6e 64 6f 77 73 20 37 20 48 6f 6d 65 20 50], IPC\$, NTLM, NTLMSSP
Eternal Champion	Keywords	WindowsPowerShell, ADMIN\$, IPC\$
Eternal Romance	Keywords	lsarpc, samr, browser, atsvc, epmapper, eventlog, keysvc, netsvcs, router, plugplay, protected_storage, scerpc, tapsrv, trkwks, wkssvc, SapiServerPipeS, PIPE_EVENTROOT, db2remotecomd, W32TIME_ALT
Eternal Red	Keywords	/root/smbshare, \PIPE, smbshare, srvsvc

Table V - Significance for data extracted for configuration of Snort rules

Exploit	Significance of data used for exploit identification
Eternal Blue	The hex values represent the core raw buffer dump. A core dump specifies the working memory of a computer program (in this case the exploit) at a particular point of time. NTLM is a security protocol used by Microsoft for providing authentication and privacy to users. NTLMSSP is a protocol for facilitating NTLM authentication.
Eternal Champion	The exploit tries to access the Windows PowerShell, which is a command line shell used for system administration. ADMIN\$ and IPC\$ are hidden administrative shares in Windows which are often used by system administrators to access computers in a network and manage services. ADMIN\$ - Used for system root folder or operating system directory IPC\$ - Used for interprocess communication
Eternal Romance	Each keyword represents a named pipe. Named pipes are commonly used for privilege escalation in Windows.
Eternal Red	A malicious authenticated Samba client can perform remote code execution attacks as a 'root' user if he/she has write access to a samba share.





```

Applications ▾ Places ▾ Terminal ▾ Wed 00:24
root@kali: ~

File Edit View Search Terminal Help
+ -- --[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

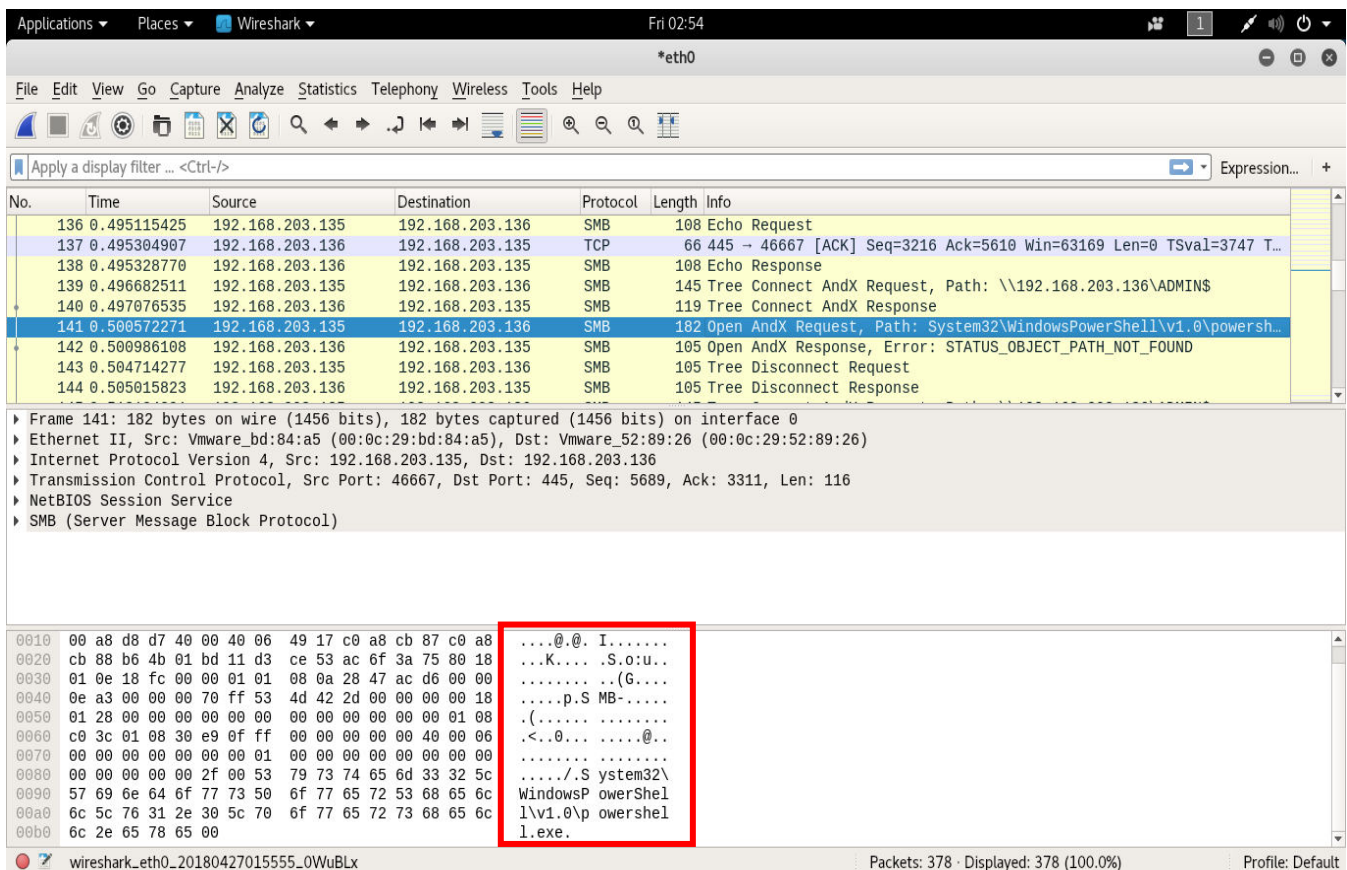
msf > use exploit/windows/smb/ms17_010_eternalblue
msf exploit(windows/smb/ms17_010_eternalblue) > set LHOST 192.168.203.160
LHOST => 192.168.203.160
msf exploit(windows/smb/ms17_010_eternalblue) > set RHOST 192.168.203.164
RHOST => 192.168.203.164
msf exploit(windows/smb/ms17_010_eternalblue) > set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf exploit(windows/smb/ms17_010_eternalblue) > exploit

[*] Started reverse TCP handler on 192.168.203.160:4444
[*] 192.168.203.164:445 - Connecting to target for exploitation.
[+] 192.168.203.164:445 - Connection established for exploitation.
[+] 192.168.203.164:445 - Target OS selected valid for OS indicated by SMB reply
[*] 192.168.203.164:445 - CORE raw buffer dump (42 bytes)
[*] 192.168.203.164:445 - 0x00000000 57 69 6e 64 6f 77 73 20 37 20 48 6f 6d 65 20 50 Windows 7 Home P
[*] 192.168.203.164:445 - 0x00000010 72 65 6d 69 75 6d 20 37 36 30 31 20 53 65 72 76 remium 7601 Serv
[*] 192.168.203.164:445 - 0x00000020 69 63 65 20 50 61 63 6b 20 31 ice Pack 1
[+] 192.168.203.164:445 - Target arch selected valid for arch indicated by DCE/RPC reply
[*] 192.168.203.164:445 - Trying exploit with 12 Groom Allocations.
[*] 192.168.203.164:445 - Sending all but last fragment of exploit packet
[*] 192.168.203.164:445 - Starting non-paged pool grooming
[+] 192.168.203.164:445 - Sending SMBv2 buffers
[*] 192.168.203.164:445 - Closing SMBv1 connection creating free hole adjacent to SMBv2 buffer.
[*] 192.168.203.164:445 - Sending final SMBv2 buffers.
[*] 192.168.203.164:445 - Sending last fragment of exploit packet!
[*] 192.168.203.164:445 - Receiving response from exploit packet
[+] 192.168.203.164:445 - ETERNALBLUE overwrite completed successfully (0xc0000000)!
[*] 192.168.203.164:445 - Sending egg to corrupted connection.
[*] 192.168.203.164:445 - Triggering free of corrupted buffer.
[*] Sending stage (206403 bytes) to 192.168.203.164
[*] Meterpreter session 1 opened (192.168.203.160:4444 -> 192.168.203.164:49180) at 2019-01-09 00:22:49 -0500
[+] 192.168.203.164:445 - =====
[*] 192.168.203.164:445 - =====WIN=====
[+] 192.168.203.164:445 - =====

meterpreter >

```

Fig 2 - Data extraction for Eternal Blue exploit



```

Applications ▾ Places ▾ Wireshark ▾ Fri 02:54
*eth0

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/> Expression... +

No. Time Source Destination Protocol Length Info
136 0.495115425 192.168.203.135 192.168.203.136 SMB 108 Echo Request
137 0.495304907 192.168.203.136 192.168.203.135 TCP 66 445 -> 46667 [ACK] Seq=3216 Ack=5610 Win=63169 Len=0 TSval=3747 T...
138 0.495328770 192.168.203.136 192.168.203.135 SMB 108 Echo Response
139 0.496682511 192.168.203.135 192.168.203.136 SMB 145 Tree Connect AndX Request, Path: \\192.168.203.136\ADMIN$
140 0.497076535 192.168.203.136 192.168.203.135 SMB 119 Tree Connect AndX Response
141 0.500572271 192.168.203.135 192.168.203.136 SMB 182 Open AndX Request, Path: System32\WindowsPowerShell\v1.0\powershell.exe
142 0.500986108 192.168.203.136 192.168.203.135 SMB 105 Open AndX Response, Error: STATUS_OBJECT_PATH_NOT_FOUND
143 0.504714277 192.168.203.135 192.168.203.136 SMB 105 Tree Disconnect Request
144 0.505015823 192.168.203.136 192.168.203.135 SMB 105 Tree Disconnect Response

▶ Frame 141: 182 bytes on wire (1456 bits), 182 bytes captured (1456 bits) on interface 0
▶ Ethernet II, Src: Vmware_bd:84:a5 (00:0c:29:bd:84:a5), Dst: Vmware_52:89:26 (00:0c:29:52:89:26)
▶ Internet Protocol Version 4, Src: 192.168.203.135, Dst: 192.168.203.136
▶ Transmission Control Protocol, Src Port: 46667, Dst Port: 445, Seq: 5689, Ack: 3311, Len: 116
▶ NetBIOS Session Service
▶ SMB (Server Message Block Protocol)

0010 00 a8 d8 d7 40 00 40 06 49 17 c0 a8 cb 87 c0 a8 .....@. I.....
0020 cb 88 b6 4b 01 bd 11 d3 ce 53 ac 6f 3a 75 80 18 ...K....S.o:u..
0030 01 0e 18 fc 00 00 01 01 08 0a 28 47 ac d6 00 00 .....(G....
0040 0e a3 00 00 00 70 ff 53 4d 42 2d 00 00 00 00 18 .....p.S MB-....
0050 01 28 00 00 00 00 00 00 00 00 00 00 00 01 08 .....
0060 c0 3c 01 08 30 e9 0f ff 00 00 00 00 00 40 00 06 ...<..@.....
0070 00 00 00 00 00 00 00 01 00 00 00 00 00 00 00 00 .....
0080 00 00 00 00 2f 00 53 79 73 74 65 6d 33 32 5c ...../S ystem32\
0090 57 69 6e 64 6f 77 73 50 6f 77 65 72 53 68 65 6c WindowsP owerShel
00a0 6c 5c 76 31 2e 30 5c 70 6f 77 65 72 73 68 65 6c l\v1.0\p ower shel
00b0 6c 2e 65 78 65 00 .....l.exe.

wireshark_eth0_20180427015555_0WuBLx Packets: 378 · Displayed: 378 (100.0%) Profile: Default

```

Fig 3 - Data extraction for Eternal Champion exploit

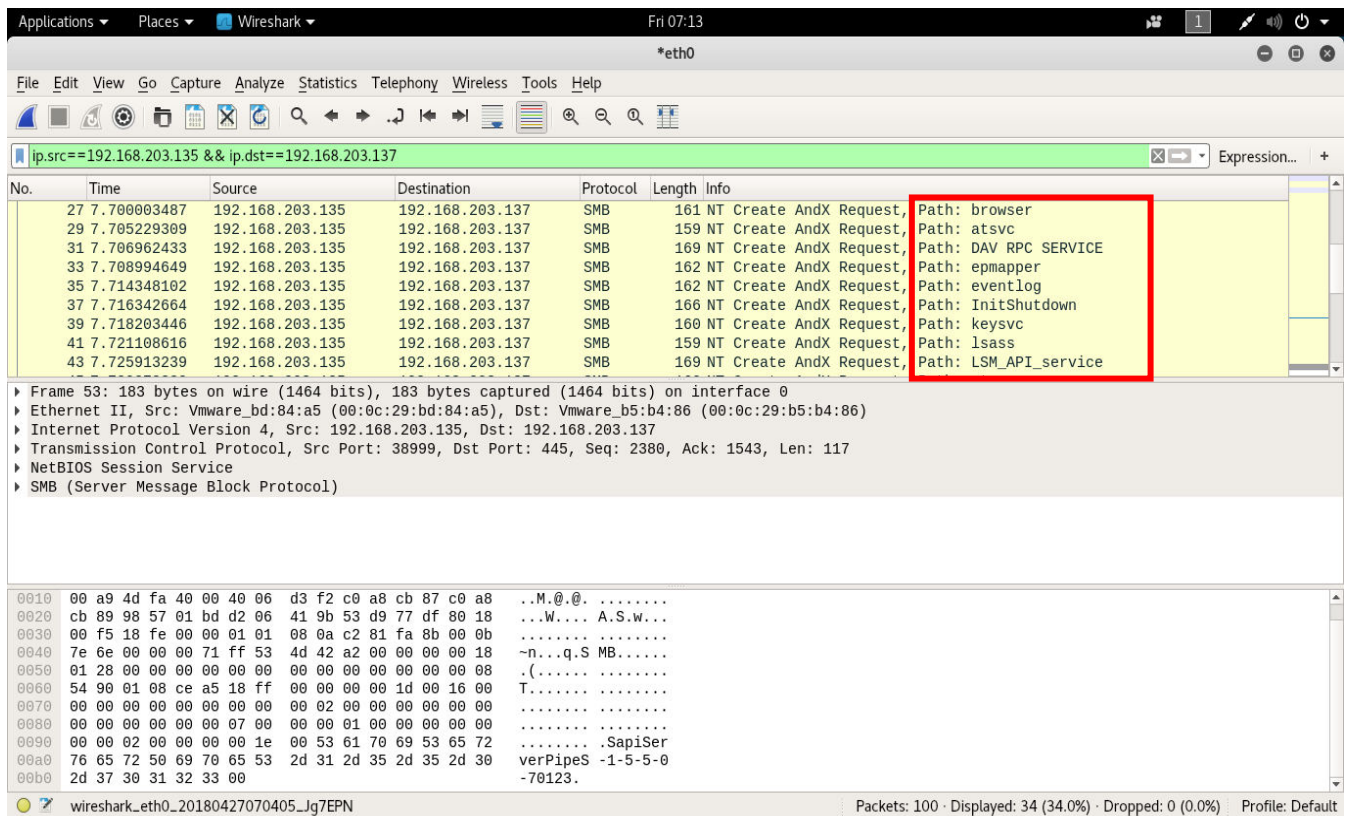


Fig 4 - Data extraction for Eternal Romance exploit

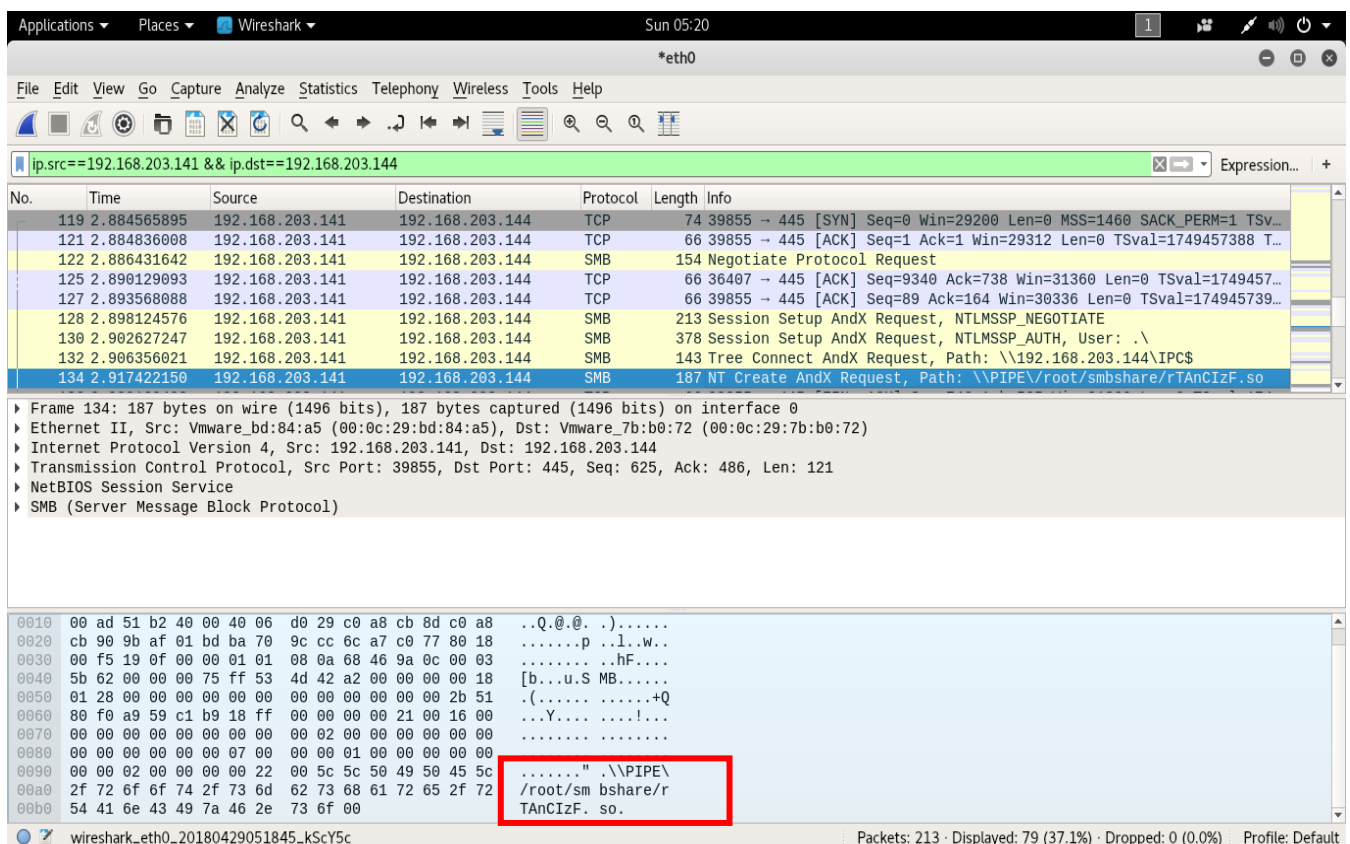


Fig 5 - Data extraction for Eternal Red exploit

### C. Samples for new Snort rules

#### Eternal Blue

- alert tcp \$EXTERNAL\_NET any -> \$HOME\_NET 445 (msg: "Exploit Detected!"; flow: to\_server, established; pcre: "/[57 69 6e 64 6f 77 73 20 37 20 48 6f 6d 65 20 50]/"; pcre: "/[72 65 6d 69 75 6d 20 37 36 30 31 20 53 65 72 76]/"; pcre: "/[69 63 65 20 50 61 63 6b 20 31]/"; reference: Exploit Database (ID's - 42030, 42031, 42315); classtype: attempted-admin; priority: 10; sid: 2094284; rev: 2;)
- alert tcp \$EXTERNAL\_NET any -> \$HOME\_NET 445 (msg: "Exploit Detected!"; flow: to\_server, established; content: "IPC\$"; reference: Exploit Database (ID's - 42030, 42031, 42315); classtype: attempted-admin; priority: 10; sid: 2094285; rev: 3;)
- alert tcp \$EXTERNAL\_NET any -> \$HOME\_NET 445 (msg: "Exploit Detected!"; flow: to\_server, established; content: "NTLMSSP"; reference: Exploit Database (ID's - 42030, 42031, 42315); classtype: attempted-admin; priority: 10; sid: 2094286; rev: 2;)

#### Eternal Champion

- alert tcp \$EXTERNAL\_NET any -> \$HOME\_NET any (msg: "Exploit Detected!"; flow: to\_server, established; content: "WindowsPowerShell"; reference: Exploit Database (ID - 43970); classtype: attempted-admin; priority: 10; sid: 20244223; rev: 3;)
- alert tcp \$EXTERNAL\_NET any -> \$HOME\_NET any (msg: "Exploit Detected!"; flow: to\_server, established; content: "ADMIN\$"; reference: Exploit Database (ID - 43970); classtype: attempted-admin; priority: 10; sid: 20244224; rev: 2;)
- alert tcp \$EXTERNAL\_NET any -> \$HOME\_NET 445 (msg: "Exploit Detected!"; flow: to\_server, established; content: "IPC\$"; reference: Exploit Database (ID - 43970); classtype: attempted-admin; priority: 10; sid: 20244225; rev: 3;)
- alert tcp \$EXTERNAL\_NET any -> \$HOME\_NET any (msg: "Exploit Detected!"; flow: to\_server, established; content: "lsarpc"; reference: Exploit Database (ID - 43970); classtype: attempted-admin; priority: 10; sid: 20244226; rev: 2;)

#### Eternal Romance

- alert tcp \$EXTERNAL\_NET any -> \$HOME\_NET any (msg: "Exploit Detected!"; flow: to\_server, established; content: "lsarpc"; reference: Exploit Database (ID - 43970); classtype: attempted-admin; priority: 10; sid: 209462812; rev: 3;)

- alert tcp \$EXTERNAL\_NET any -> \$HOME\_NET any (msg: "Exploit Detected!"; flow: to\_server, established; content: "samr"; reference: Exploit Database (ID - 43970); classtype: attempted-admin; priority: 10; sid: 209462813; rev: 3;)

- alert tcp \$EXTERNAL\_NET any -> \$HOME\_NET any (msg: "Exploit Detected!"; flow: to\_server, established; content: "browser"; reference: Exploit Database (ID - 43970); classtype: attempted-admin; priority: 10; sid: 209462814; rev: 2;)

- alert tcp \$EXTERNAL\_NET any -> \$HOME\_NET any (msg: "Exploit Detected!"; flow: to\_server, established; content: "epmapper"; reference: Exploit Database (ID - 43970); classtype: attempted-admin; priority: 10; sid: 209462815; rev: 2;)

- alert tcp \$EXTERNAL\_NET any -> \$HOME\_NET any (msg: "Exploit Detected!"; flow: to\_server, established; content: "eventlog"; reference: Exploit Database (ID - 43970); classtype: attempted-admin; priority: 10; sid: 209462816; rev: 2;)

#### Eternal Red

- alert tcp \$EXTERNAL\_NET any -> \$HOME\_NET 445 (msg: "Exploit Detected!"; flow: to\_server, established; content: "/root/smbshare"; reference: Exploit Database (ID's - 42060, 42084); classtype: shellcode-detect; priority: 10; sid: 20242290; rev: 2;)

- alert tcp \$EXTERNAL\_NET any -> \$HOME\_NET 445 (msg: "Exploit Detected!"; flow: to\_server, established; content: "\\PIPE"; reference: Exploit Database (ID's - 42060, 42084); classtype: shellcode-detect; priority: 10; sid: 20242291; rev: 3;)

- alert tcp \$EXTERNAL\_NET any -> \$HOME\_NET 445 (msg: "Exploit Detected!"; flow: to\_server, established; content: "smbshare"; reference: Exploit Database (ID's - 42060, 42084); classtype: shellcode-detect; priority: 10; sid: 20242292; rev: 3;)

- alert tcp \$EXTERNAL\_NET any -> \$HOME\_NET 445 (msg: "Exploit Detected!"; flow: to\_server, established; content: "srvsvc"; reference: Exploit Database (ID's - 42060, 42084); classtype: shellcode-detect; priority: 10; sid: 20242293; rev: 2;)

### VII. DETECTION OF EXPLOITS USING NEW SNORT RULES

The newly rules configured must be added to the rule path in the Snort configuration file (snort.conf) before we run Snort to detect the exploits. We can also create a log file for each exploit and add it to the rule path.



```

root@ubuntu: /etc/snort/rules
| 4 byte states : 0.00
-----
[ Number of patterns truncated to 20 bytes: 1039 ]
pcap DAQ configured to passive.
Acquiring network traffic from "eth0".
Reload thread starting...
Reload thread started, thread 0x7f3793836700 (3011)
Decoding Ethernet

--== Initialization Complete ==--

-*> Snort! <*-
Version 2.9.6.0 GRE (Build 47)
By Martin Roesch & The Snort Team: http://www.snort.org/snort/snort-team
Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
Copyright (C) 1998-2013 Sourcefire, Inc., et al.
Using libpcap version 1.5.3
Using PCRE version: 8.31 2012-07-06
Using ZLIB version: 1.2.8

Rules Engine: SF_SNORT_DETECTION_ENGINE Version 2.1 <Build 1>
Preprocessor Object: SF_DNS Version 1.1 <Build 4>
Preprocessor Object: SF_SDF Version 1.1 <Build 1>
Preprocessor Object: SF_GTP Version 1.1 <Build 1>
Preprocessor Object: SF_DCERPC2 Version 1.0 <Build 3>
Preprocessor Object: SF_SMTP Version 1.1 <Build 9>
Preprocessor Object: SF_SIP Version 1.1 <Build 1>
Preprocessor Object: SF_MODBUS Version 1.1 <Build 1>
Preprocessor Object: SF_SSLPP Version 1.1 <Build 4>
Preprocessor Object: SF_REPUTATION Version 1.1 <Build 1>
Preprocessor Object: SF_DNP3 Version 1.1 <Build 1>
Preprocessor Object: SF_SSH Version 1.1 <Build 3>
Preprocessor Object: SF_IMAP Version 1.0 <Build 1>
Preprocessor Object: SF_POP Version 1.0 <Build 1>
Preprocessor Object: SF_FTPTELNET Version 1.2 <Build 13>
Commencing packet processing (pid=3006)
03/31-23:20:46.932418  [**] [1:2465:7] NETBIOS SMB-DS IPC$ share access [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP
03/31-23:20:57.136987  [**] [1:41978:3] Eternal Blue Exploit [**] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 1
92.168.203.135:34117 -> 192.168.203.132:445

```

Fig 6 - Eternal Blue detection using Snort

Table VI - Testing of configured Snort rules

Exploit	Detection rate with Snort rules configured by us
Eternal Blue	87.50%
Eternal Champion	85.71%
Eternal Romance	85.71%
Eternal Red	77.77%

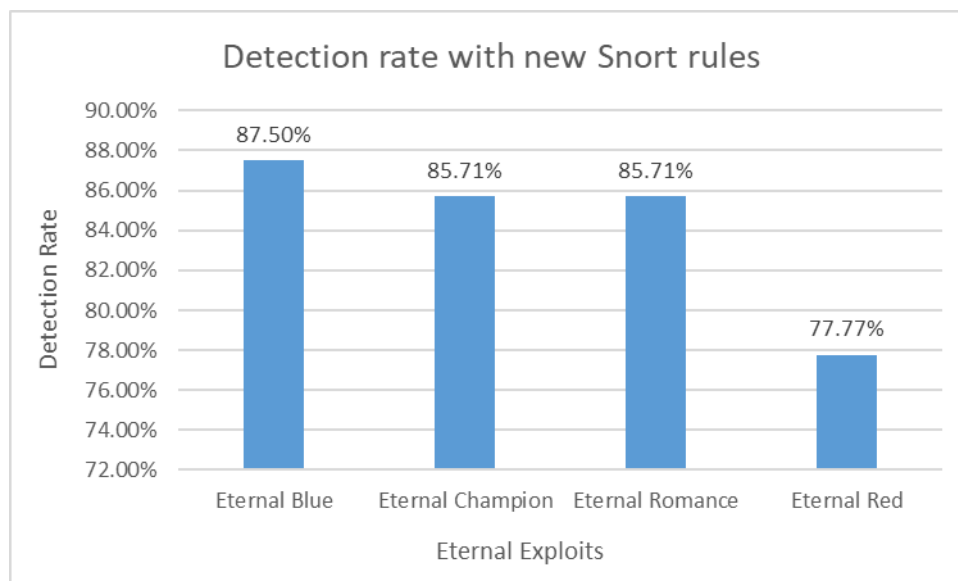


Fig 7 - Detection rates using new Snort rules

Table VII - Comparison between ‘Exploit detection using Snort and Wireshark’ and ‘Exploit detection using software and antivirus’

Parameters	Using configured Snort rules	Using Exploit detection and antivirus softwares
Exploit detection without downloading system or rules	Present	Not Present
Exploit detection without executing system or rules	Present	Not Present
Number of systems protected	Snort can be deployed over the entire network and thus the newly configured rules will protect all the systems within the network range.	Only the individual systems are protected. The software must be present on each system for detection of exploits.
Detection Speed	Snort analyses network traffic in real time against the rules defined by us.	Detection speed is slower because complete scanning of the system takes a lot of time.
Updating the configured rules or signatures	The configured rules can be revised as many times as required to detect presence of new ransomware exploits. Rules can be updated by the network administrator. No action required by the user.	Each user may have to manually download the updated version of the software. Failure to do so would make the system vulnerable to any new exploits.
Cost	Snort is a free open source system. Rules are configured after analysis of exploit using Wireshark, which is also a freely available tool.	State of the art softwares are usually paid.
Operating System dependency	The Snort rules configured by us are independent of the operating system present.	Exploit detection softwares are usually operating system specific. Exploit detection softwares for Windows may not work with Linux based systems.

After the rule files have been added to the rule path, we can run Snort in the network intrusion detection mode by the following command -

```
snort -A console -i eth0 -c /etc/snort/snort.conf -l /var/log/snort -K ascii
```

Syntax - snort [- options] <filter options>

**-A** - This option allows the user to set the type of alert mode - test, full, console, etc. The 'console' mode used here displays all generated alerts on the screen.

**-i** - Snort listens on the specified interface. Here the interface is 'eth0'.

**-c** - This option directs Snort to use the rule files for exploit detection. The user must provide the full path to the rule files. Here the rule path is '/etc/snort/snort.conf', which is the default path.

**-l** - It is used for specifying the path to the log directory. Here, the log path is '/var/log/snort', which is the default path. Log analysis can be useful in forensics and incident response in case of a security breach.

**-K** - This option specifies the logging mode. The default mode is pcap. Here, the logging mode is set to 'ascii' which means that the log file will be saved in the ASCII format.

## VIII. RESULTS AND DISCUSSION

The Snort rules configured by us were successfully able to detect and differentiate between the Eternal series exploits. We have tried to configure generic Snort rules for detection of the Eternal exploits so that the rules can be used to detect similar exploits in the future. Table 6 contains the detection rate of each exploit when tested against the Snort rules configured by us. Table 7 compares exploit detection using Snort with exploit detection using antivirus software.

## IX. CONCLUSION

Through this research paper, we proposed a technique for early detection of ransomware exploits through effective use of Wireshark and Snort, which are free open source tools. The proposed approach is cost-effective. It can help in reduction of financial losses of companies from cyber attacks. The results show the importance of an intrusion detection system in preventing potential security breaches. Improvement in the Snort rules can increase the accuracy of detection by ensuring lesser number of false positives.

## REFERENCES

- [1] Abdullah A. Al-khatib and Waleed A. Hammood, "Mobile Malware and Defending Systems: Comparison Study", *International Journal of Electronics and Information Engineering*, vol. 6, pp. 116-123, 2016.
- [2] J. M. Ceron, C. B. Margi and L. Z. Granville, "MARS: An SDN-based malware analysis solution", *2016 IEEE Symposium on Computers and Communication (ISCC)*, pp. 525-530, June 2016.
- [3] Q. Chen and R. A. Bridges, "Automated Behavioral Analysis of Malware: A Case Study of WannaCry Ransomware", *2017 16th*

- IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 454-460, December 2017.
- [4] Myroslav Komar, Vitaliy Dorosh, Grygoriy Hladiy and Anatoliy Sachenko, "Deep Neural Network for Detection of Cyber Attacks", *2018 IEEE First International Conference on System Analysis & Intelligent Computing (SAIC)*, October 2018.
- [5] D. Kao and S. Hsiao, "The dynamic analysis of WannaCry ransomware", *2018 20th International Conference on Advanced Communication Technology (ICACT)*, pp. 159-166, February 2018.
- [6] D. Kao and S. Hsiao, "The static analysis of WannaCry ransomware", *2018 20th International Conference on Advanced Communication Technology (ICACT)*, pp. 1-1, February 2018.
- [7] Sanggeun Song, Bongjoon Kim and Sangjun Lee, "The Effective Ransomware Prevention Technique Using Process Monitoring on Android Platform", *Mobile Information Systems*, vol. 2016, 9 pages, 2016.
- [8] Amin Kharaz, Sajjad Arshad, Collin Mulliner, William Robertson and Engin Kirda, "UNVEIL: A Large-Scale, Automated Approach to Detecting Ransomware", *25th USENIX Security Symposium*, pp. 757-772, August 2016.
- [9] Steven R. Snapp, James Brentano, Gihan V. Dias, Terrance L. Goan, L. Todd Heberlein, Che-Lin Ho, Karl N. Levitt, Biswanath Mukherjee, Stephen E. Smaha, Tim Grance, Daniel M. Teal, and Doug Mansur, "DIDS (Distributed Intrusion Detection System) - Motivation, Architecture, and An Early Prototype", *14th National Computer Security Conference*, October 1991.
- [10] J. Gómez, C. Gil, N. Padilla, R. Baños, C. Jiménez, "Design of a Snort-Based Hybrid Intrusion Detection System", *Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living (IWANN 2009)*, Lecture notes in Computer Science, vol. 5518, pp. 515-522, June 2009.
- [11] Ahmad Javaid, Quamar Niyaz, Weiqing Sun and Mansoor Alam, "A Deep Learning Approach for Network Intrusion Detection System", *9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*, pp. 21-26, December 2015.
- [12] Yakuta Tayyebi and D.S. Bhilare, "A Comparative Study of Open Source Network Based Intrusion Detection Systems", *International Journal of Computer Science and Information Technologies*, vol. 9 (2), pp. 23-26, 2018.
- [13] C.J. Coit, S. Staniford and J. McAlerney, "Towards faster string matching for intrusion detection or exceeding the speed of Snort", *DARPA Information Survivability Conference and Exposition II. DISCEX'01*, vol. 1, pp. 367-373, Anaheim, CA, USA, 12-14 June 2001.
- [14] Martin Roesch, "Snort - Lightweight Intrusion Detection for Networks", *Proceedings of LISA 1999: 13th System's Administration Conference*, pp. 229-238, November 1999.
- [15] Samuel Patton, David Doss, William Yurcik, "An Achilles' Heel in Signature-Based IDS: Squealing False Positives in SNORT", *4th International Symposium on Recent Advances in Intrusion Detection (RAID)*, October 2001.
- [16] Zhimin Zhou, Chen Zhongwen, Zhou Tiecheng and Guan Xiaohui, "The study on network intrusion detection system of Snort", *2010 International Conference on Networking and Digital Society*, pp. 194-196, May 2010.
- [17] K. Wong, C. Dillabaugh, N. Seddigh and B. Nandy, "Enhancing Suricata intrusion detection system for cyber security in SCADA networks", *2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE)*, pp. 1-5, April-May 2017.