



Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permitted without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Reverse-Engineering Malware: Malware Analysis Tools and Techniques (Forensics)"
at <http://www.giac.org/registration/grem>

Reverse Engineering of WannaCry Worm and Anti Exploit Snort Rules

GIAC (GREM) Gold Certification

Author: Hirokazu Murakami, h-murakami@sig-c.co.jp

Advisor: Chris Walker, CISSP, GCED, CCISO

Accepted: March 15, 2018

Abstract

Today, a lot of malware is being created and utilized. To solve this problem, many researchers study technologies that can quickly respond automatically to detected malware. Using artificial intelligence (AI) is such an example. However, modern AI has difficulty responding to new attack methods. On the other hand, malware consists of variants, and the root (core) part often uses the same technology. Therefore, I think that if we can identify that core part of malware through analysis, we can identify many variants as well. Consider the possibility of reverse engineering to identify countermeasures from malware analysis results.

1. Introduction

Ransomware "WannaCry" spread worldwide in May 2017. Many systems were damaged due to this Ransomware. This Ransomware has file encryption and worm functions. I obtained a sample of the WannaCry and I analyzed file encryption and decryption processing, as well as worm functions.

The encryption function of the WannaCry encrypts the file with 128-bit AES encryption (CBC mode). In addition, it encrypts the AES encryption key with the RSA public key created on its own machine. Then, it combines the "WANACRY!" string, the encrypted AES key and the encrypted file data. ".WNCRY" is added to the end of the file name. The RSA secret key created on your machine is encrypted with the RSA public key embedded in the ransomware and saved as the file name "00000000.eky." The attacker tries to transmit "00000000.eky" via TOR by pressing the "Check Payment" button on the intimidation screen. When the check is successful, it receives the decrypted RSA secret key and it is trying to save the key as "00000000.dky." However, it is known that there is a problem with the function and files are not decrypted even if ransom is paid.

The function of the worm uses the exploit "EternalBlue" and the backdoor "DoublePulsar." These result in major damage. In the main part of this document, I focus on the analysis results of the worm. Then, I try to detect and shut down the worm's communication using the analysis result. Hash values of the analyzed samples are as follows:

MD5: db349b97c37d22f5ea1d1841e3c89eb4

SHA1: e889544aff85ffaf8b0d0da705105dee7c97fe26

SHA256:24d004a104d4d54034dbccff2a4b19a11f39008a575aa614ea04703480b1
022c

A patch was made to wait 20 seconds before starting the service in order to attach to the process, and a survey was conducted.

Hirokazu Murakami, h-murakami@sig-c.co.jp

The paper is based on my R&D work in SIG Corp. And it is published in SIG Corp (SIG Corp, 2017).

2. Fundamental of WannaCry's worm

2.1. Outline

WannaCry spreads using the vulnerability of SMB. Infected and executed code works with system privilege because SMB is a system privilege. After the WannaCry infection, it performs two actions. One is encryption processing as ransomware. The other is the function of the worm. This attempts to connect on port 445 which SMB uses on the network. When connection is possible, it checks whether the SMB “EternalBlue” vulnerability exists. If there is a vulnerability of “EternalBlue,” it checks whether it is infected with the “DoublePulsar” backdoor. If there is no “DoublePulsar” backdoor, it infects the backdoor using the “EternalBlue” vulnerability. When it infects the “DoublePulsar” backdoor, it sends the body of the WannaCry malware. This communication is evaded by IDS / IPS detection by encrypting with 32 bit XOR.

2.2. Worm Detail - Pre Exploit

When infected with an early version of the WannaCry, a system first connects to the URL of the famous “Kill Switch.” If the connection cannot be established, the processing is continued. Next, one would check the arguments. If it has two arguments, execute the routine which operates as a worm. In the other cases, it executes a routine to encrypt it as a ransomware. When operating as a worm, it runs as a service. This service is registered when it acts as ransomware immediately after infection. Service names and executable file names are given names that look like Microsoft's legitimate process as follow:

- Service Name: mssecsvc2.0
- Display name: Microsoft Security Center (2.0) Service
- Execution file path: C:\Windows\ mssecsvc.exe -m security
- Startup type: Auto
- Recovery: Restart service (after 1 minute)

Hirokazu Murakami, h-murakami@sig-c.co.jp

The worm function searches for machines that can be connected by SMB. The search method creates an IPv4 address with a random number. At this time, avoiding the first octet becoming 127 or above 224. For this reason, there is a possibility of infecting machines capable of SMB connection whether it is a global address or a local address. If the connection succeeds at the port 445, the first to third octets of the address are made identical, and the fourth octet is attempted to connect the port 445 in order from 1 to 254.

If it can connect at port 445, the WannaCry's service creates a thread for exploit and malware transmission. The thread sends a request with specific data at first. If there is an "EternalBlue" vulnerability, the target returns a specific return value (STATUS_INSUFF_SERVER_RESOURCES). For this reason, malware checks whether the return value is a vulnerable value (STATUS_INSUFF_SERVER_RESOURCES). If the return value is a vulnerable return value, it sends a request to check whether it is infected with the "DoublePulsar." The thread checks that the Multiplex ID of the received data from the target is 81 (0x51). If it matches, the thread judges that it is infected with the "DoublePulsar." As a reason, the thread send a MultiPlex ID of 65 (0x41) in the request to check whether it is infected with the "DoublePulsar." If the "DoublePulsar" is not infected, this value will not be changed. However, if the "DoublePulsar" is infected, it will change this value. Therefore, malware can confirm the infection of the "DoublePulsar" by checking this value. If target is infected with the "DoublePulsar," the thread runs post exploit.

Assembly code snippet:

```

:00401B0E push    400h
:00401B13 push    ecx
:00401B14 push    esi
:00401B15 call    sub_4097B6
:00401B1A cmp     eax, 0FFFFFFFh
:00401B1D jz      short loc_401B50
:00401B1F cmp     byte ptr [esp+25h], 5
:00401B24 jnz     short loc_401B50
:00401B26 cmp     byte ptr [esp+26h], 2
:00401B2B jnz     short loc_401B50
:00401B2D mov     al, [esp+27h]
:00401B31 test    al, al
:00401B33 jnz     short loc_401B50
:00401B35 cmp     byte ptr [esp+28h], 0C0h
:00401B3A jnz     short loc_401B50
:00401B3C push    esi
:00401B3D call    sub_4097B0
:00401B42 pop     edi

```

Annotations:

- Recieve function**: Points to the `call sub_4097B6` instruction.
- *Reverse order caused by little endian**: Points to the comparison instructions for the last, 3rd, 2nd, and 1st bytes.
- Compare last byte is 0x05**: Points to `cmp byte ptr [esp+25h], 5`.
- Compare 3rd byte is 0x02**: Points to `cmp byte ptr [esp+26h], 2`.
- Compare 2nd byte is 0x00**: Points to `mov al, [esp+27h]` and `test al, al`.
- Compare 1st byte is 0xC0**: Points to `cmp byte ptr [esp+28h], 0C0h`.
- Judgment of STATUS_INSUFF_SERVER_RESOURCES (=0xC0000205)**: Points to the `cmp byte ptr [esp+28h], 0C0h` instruction.
- Continue if match**: Points to the `call sub_4097B0` instruction.

Hirokazu Murakami, h-murakami@sig-c.co.jp

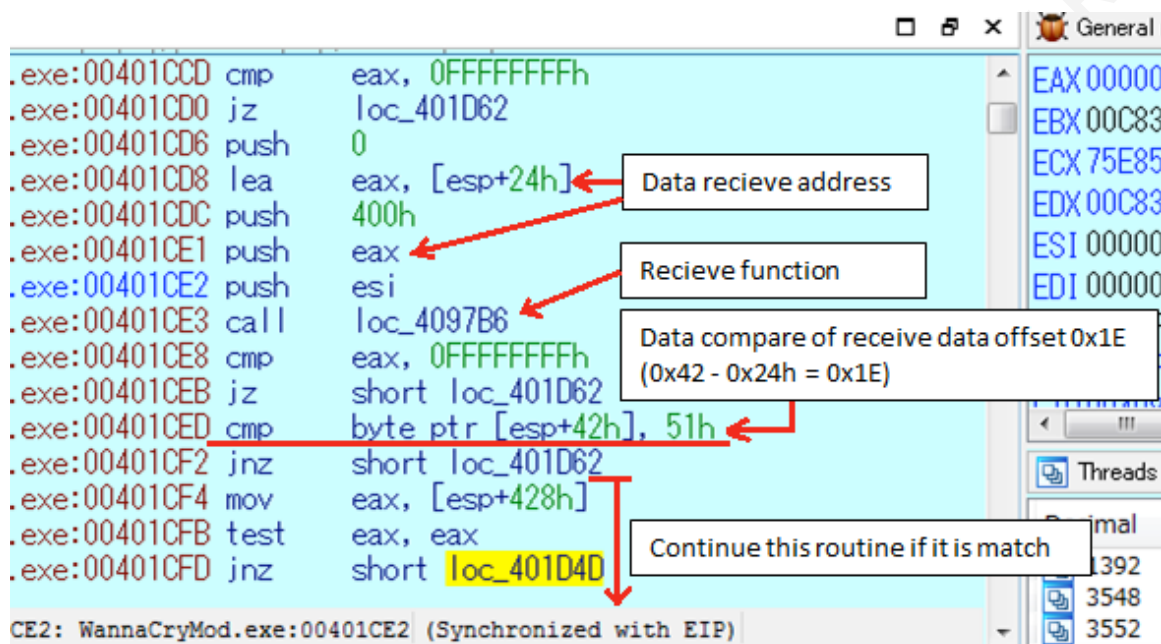


Figure 1 Determine if there is a vulnerability in EternalBlue

Figure 2 Determining whether the backdoor is infected

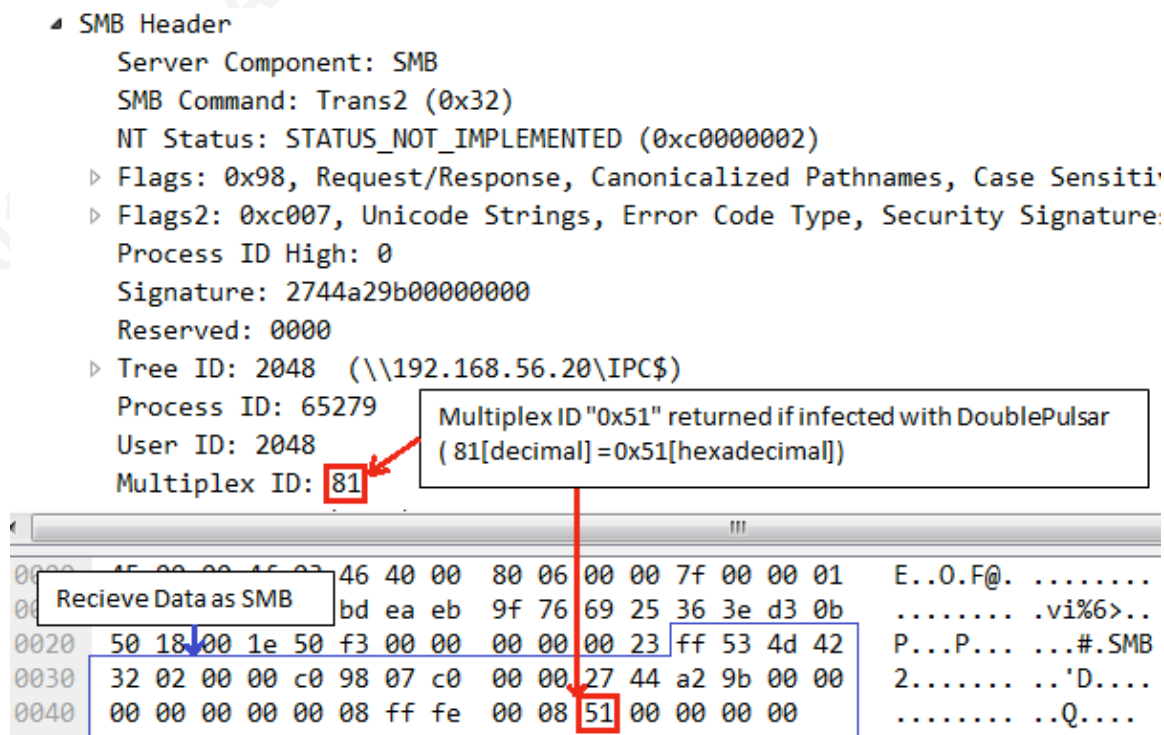


Figure 3 Response if DoublePulsar backdoor is infected

2.3. Worm Detail - Exploit

In order to infect the “DoublePulsar” to target, the WannaCry executes the “EternalBlue” exploit. During Transaction 2, Secondary Request and SMB 2 data are transmitted in exploit. This has been found to be a communication for memory destruction. (Trend Micro’s Blog MS17-010: EternalBlue’s Large Non-Paged Pool Overflow in SRV Driver: <http://blog.trendmicro.com/trendlabs-security-intelligence/ms17-010-eternalblue/>)

After this communication, the “DoublePulsar” backdoor is sent. It can be confirmed that this backdoor data includes decryption processing that performs 32 bit XOR. After communication, it checks whether the “DoublePulsar” is infected and

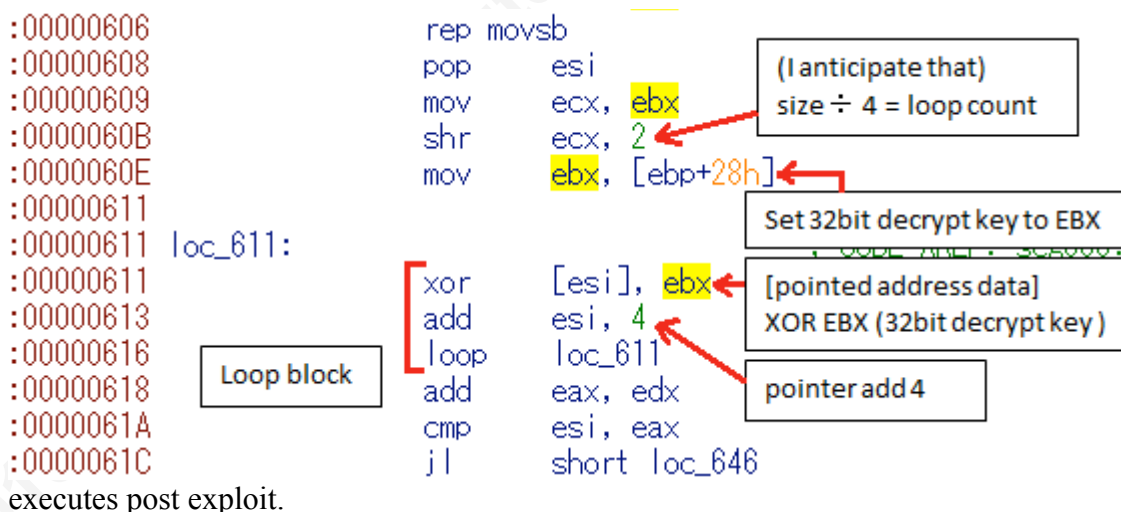


Figure 4 Processing which is regarded as decoding received data of backdoor

2.4. Worm Detail - Post Exploit

After the target has infected the “DoublePulsar” backdoor, the thread sends the WannaCry payload. For transmission, Trans2 SESSION_SETUP which is the SMB reserved command is used. The thread encrypts the WannaCry payload with 32 bit XOR and transmits it. It is confirmed that the 32 bit decryption key is transmitted in the last communication at least. When the target receives this data, it is decrypted by the “DoublePulsar” and the remote code is executed. The WannaCry executable file is created with the file name mssecsvc.exe in the Windows folder by remote code and is executed without arguments. As a result, the target is infected with WannaCry.

Hirokazu Murakami, h-murakami@sig-c.co.jp

2.5. Communication of the “EternalBlue” and the “DoublePulsar”

I will detail the communication of the “EternalBlue” and the “DoublePulsar.”

The “EternalBlue” performs communication for memory destruction and executes remote code. The WannaCry’s “EternalBlue” exploit communication send the payload for memory destruction. However, the content of this payload is dummy data. The important point is the size of the data to be transmitted. An attacker can use arbitrary data for payload for memory destruction. Therefore, when variants are created, the contents of the payload may be changed. For this reason, I think that it is not appropriate to create a signature on the “EternalBlue” payload.

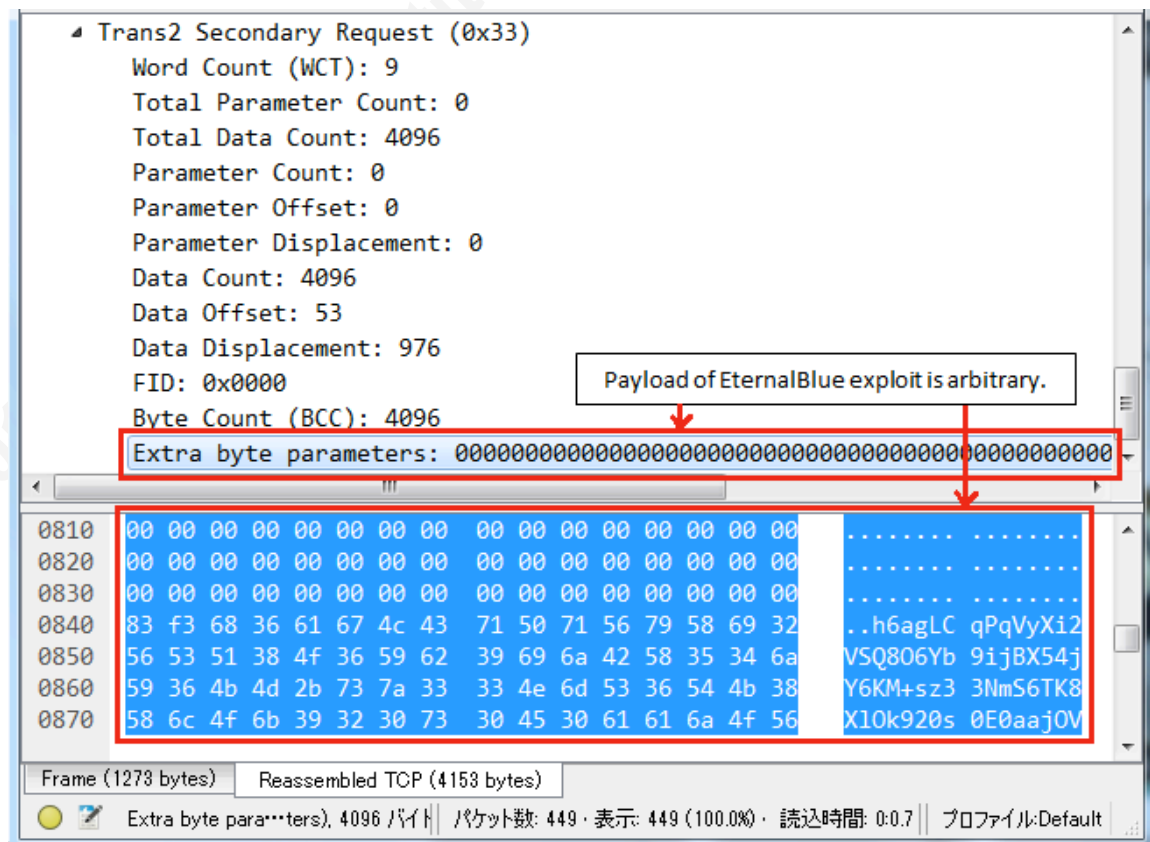


Figure 5 Payload of “EternalBlue” Exploit sent by WannaCry

The WannaCry sends the payload of the “DoublePulsar” in plaintext after memory destruction by the “EternalBlue.” We can detect attacks by signature of the DoublePulsar's payload. Remember, if the backdoor to be sent with the “EternalBlue”

exploit is changed, we will not be able to detect it. For this reason, we think that it is more useful to detect characteristic metadata of the “EternalBlue” than the transmitted payload.

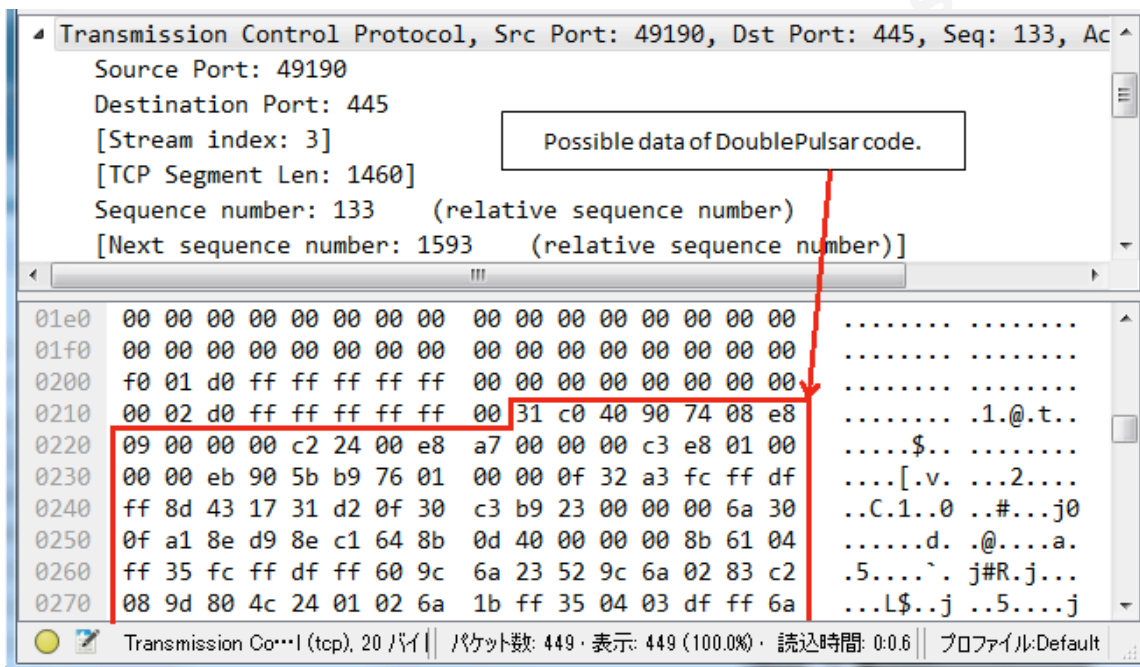


Figure 6 “DoublePulsar” code sent by “EternalBlue”

The “DoublePulsar” performs 32-bit XOR encryption on payload transmission. When analyzing the WannaCry's worm, I found a strange process to XOR with a 32 bit data before sending the payload. I thought that this 32 bit data might be the key. Therefore, I recorded this data. I further analyzed, and captured communication using Wireshark. Then, I found that the 32 bit record matching that the key is transmitted with the first communication and the last communication. The transmitted data was decrypted by XOR with the 32 bit key. As a result, the execution code was output. For this reason, I was convinced that the “DoublePulsar” is encrypting 32 bit XOR. In addition, I also found that the 32 bit key is created at random every time. You may point out that the 32 bit key encryption is vulnerable. But, detection on real time communication is difficult even with weak encryption. This means that detection by the payload of the “DoublePulsar” which is XOR encrypted with random 32 bits key is not appropriate.

From the reverse engineering results of the “EternalBlue” and the “DoublePulsar,” our strategy should be metadata detection rather than payload detection.

Obtaining this knowledge is an advantage of reverse engineering. In the next section, I will use this knowledge to examine best practices for detection.

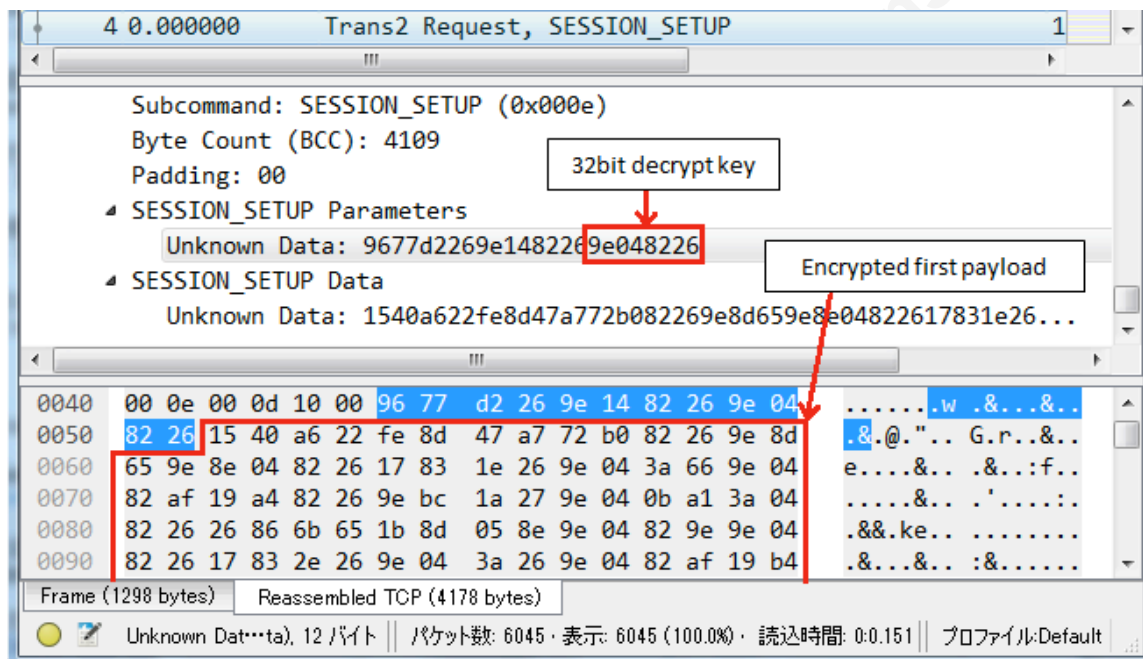


Figure 7 First communication record by "DoublePulsar"

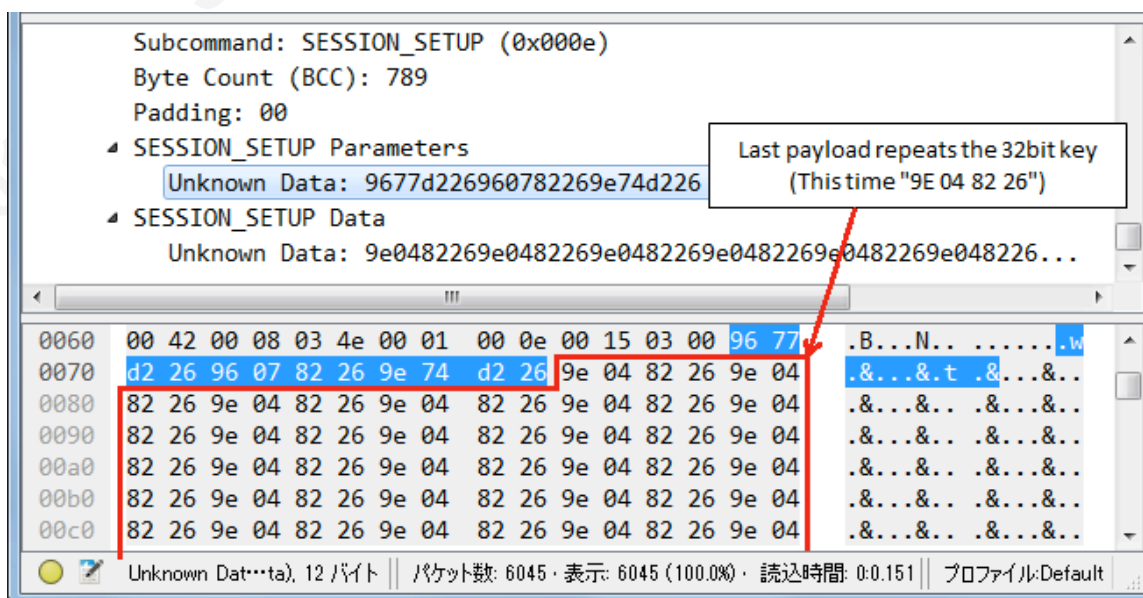


Figure 8 Last communication record by "DoublePulsar"

Encrypted data

15 40 A6 22 FE 8D 47 A7 72 B0 82 26 9E 8D 65 9E ...



XOR by key "9E 04 82 26"

8B 44 24 04 60 89 C5 81 EC B4 00 00 00 89 E7 B8 ...

Figure 9 Decrypting payload data by 32bit key

Stack point setting and store parameters to local variants.

```

seg000:0000003B  mov     eax, 0
seg000:00000040  loc_40:  ; DATA XREF: |
seg000:00000040  mov     [edi+0ACh], eax
seg000:00000046  mov     eax, 0
seg000:0000004B  mov     [edi+0B0h], eax
seg000:00000051  mov     eax, 188h
seg000:00000056  mov     [edi+94h], eax
seg000:0000005C  mov     ebx, dword ptr fs:loc_35+3
seg000:00000063  loc_63:  ; DATA XREF: |
seg000:00000063  mov     ax, [ebx+6]
seg000:00000065  shl     eax, 10h
seg000:00000067  mov     ax, [ebx]
seg000:00000069  and     ax, 0F000h
seg000:00000071  loc_71:  ; CODE XREF: |
seg000:00000071  mov     ebx, [eax]
seg000:00000073  loc_73:  ; DATA XREF: |
seg000:00000073  cmp     bx, 5A4Dh
seg000:00000078  jz      short loc_81
seg000:0000007A  sub     eax, 1000h
seg000:0000007F  jmp     short loc_71
seg000:00000081  ; -----
seg000:00000081  loc_81:
seg000:00000081  mov     mov
seg000:00000084  ; -----
seg000:00000086  ; ===== S U
seg000:00000086
seg000:00000086
seg000:00000086

```

5A=ASCII:"Z", 4D=ASCII:"M"
It is little endian.
It will compare "MZ"
↓
This routine is looking for "MZ" magic number to find executable.

Figure 10 32 bit XOR encrypted and transmitted code in WannaCry

3. Snort Detection Rules from Reverse Engineering Knowledge

The behavior of the worm was discovered by reverse engineering and communication capture. I tried to detect communication of the “EternalBlue” and the “DoublePulsar” using this knowledge. I created the Snort definition. Then I ran the “EternalBlue” exploit and the “DoublePulsar” backdoor in the verification environment and verified the detection results.

3.1. Hit illegal SMB2 transmission sent when sending the “EternalBlue” exploit

The “EternalBlue” is known to communicate a certain pattern in order to find vulnerable targets. For such communications, the parameters are not normally often seen. As a result of looking for communication that could be used for detection, I found that SMB2 is conducting a “funny” communication. In this communication, although it can be seen SMB2 by SMB header, all subsequent parameters are 0. The most characteristic is that the “StructureSize” (2 bytes) after the SMB header is “00 00.” Microsoft's SMB2 manual specifies that this parameter must be “64” (0x40h). I think that it is an unauthorized communication necessary for an exploit at the “EternalBlue,” and it can't be a normal communication.

I will consider the grounds of the detection rule as follows:

- Communication is “SMB2”
- StructureSize is “00 00”
- The data that follows (as necessary) is also “00”

I considered whether there is a possibility of false positive detection under this condition. According to the manual of SMB2, "Server Message Block (SMB) Protocol Version2 and 3", the structure size must be "40 00." Also, even if another pattern can be created in the structure size in the future, I think that it will not become "00 00." For this reason, I think that this condition will not be met for normal communication, so it can be used for detection of exploit by the “EternalBlue.”

Hirokazu Murakami, h-murakami@sig-c.co.jp

The snort definition based on this is as follows:

```
alert tcp any any -> any 445 (msg:" SMB2 packet data is possible to be
EternalBlue Exploit "; content:"|fe|SMB|00 00 00 00 00 00 00 00|"; offset:4;
sid:1000001; rev:1;)
```

0000	00 0c 29 37 4f fb 00 0c 29 b4 5f 89 08 00 45 00	..)70...)._...E.
0010	00 b8 a9 e7 40 00 40 06 0c ef c0 a8 01 0d c0 a8@.@.
0020	01 0c 9f 23 01 bd 43 37 9b 49 6c d0 05 4d 80 18	...#...C7 .I1..M..
0030	00 e5 fc c2 00 00 01 01 08 0a 00 04 32 ad 00 002...
0040	85 61 00 00 ff f7 fe 53 4d 42 00 00 00 00 00 00	.a.....S MB.....
0050	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0080	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0090	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

ProtocolId (4 bytes): The protocol identifier. The value MUST be (in network order) 0xFE, 'S', 'M', and 'B'.

StructureSize (2 bytes): MUST be set to 64, which is the size, in bytes, of the [SMB2 header](#) structure.

CreditCharge (2 bytes): In the SMB 2.0.2 dialect, this field MUST NOT be used and MUST be reserved. The sender MUST set this to 0, and the receiver MUST ignore it. In all other dialects, this field indicates the number of credits that this request consumes.

(ChannelSequence/Reserved)/Status (4 bytes): In a request, this field is interpreted in different ways depending on the SMB2 dialect.

(Quoted from the manual "[MS-SMB 2]: Server Message Block (SMB) Protocol Versions 2 and 3")

Figure 11 Abnormal data transmitted on SMB2 by "EternalBlue" exploit

3.2. Hit the "EternalBlue" vulnerability check request and vulnerable response

If there is a vulnerability of the "EternalBlue," we know that value of STATUS_INSUFF_SERVER_RESOURCES will be returned for TRANS_PEEK_NMPIPE request with FID of "0" which is supposed to be required. This communication is used by attackers to check if the target is vulnerable. If we detect this communication, we can see that an attacker is exploring.

I will consider this request and response detection rule as follows:

Request:

- Communication is "SMB"

- Request command is SMB_COM_TRANSACTION(0x25h)
- Request subcommand is TRANS_PEEK_NMPIPE(0x0023)
- FID is 0

Response:

- Communication is "SMB"
- Return command is SMB_COM_TRANSACTION(0x25h)
- Return value is STATUS_INSUFF_SERVER_RESOURCES (0xC0000205)

Source	Destination	Protocol	Length	Info
192.168.1.12	192.168.1.13	SMB	116	Tree Connect AndX Response
192.168.1.13	192.168.1.12	SMB Pipe	144	PeekNamedPipe Request, FID: 0x0000
192.168.1.12	192.168.1.13	SMB	105	Trans Response, Error: STATUS_INSUFF_SERVER_RESOURCES
192.168.1.13	192.168.1.12	TCP	66	42497 → 445 [ACK] Seq=887 Ack=710 Win=31 Len=0 TSval=...
192.168.1.13	192.168.1.12	SMB	149	Trans2 Request, SESSION_SETUP
192.168.1.12	192.168.1.13	SMB	105	Trans2 Response, SESSION_SETUP, Error: STATUS_NOT_IMPLEMENTED

Source	Destination	Protocol	Length	Info
192.168.1.13	192.168.1.11	SMB	140	Tree Connect AndX Request
192.168.1.11	192.168.1.13	SMB	116	Tree Connect AndX Response
192.168.1.13	192.168.1.11	SMB Pipe	144	PeekNamedPipe Request, FID: 0x0000
192.168.1.11	192.168.1.13	SMB	105	Trans Response, Error: STATUS_INVALID_HANDLE
192.168.1.13	192.168.1.11	TCP	66	42429 → 445 [FIN, ACK] Seq=1003 Ack=886 Win=31...

Figure 12 "EternalBlue" vulnerability check request and response

I considered whether there is a possibility of false positive detection under this condition. It is a possibility of false positive detection by response under this condition. The return value "STATUS_INSUFF_SERVER_RESOURCES" may be returned for another request not related to the vulnerability. For this reason, I think that it can be accurately detected by combining with an incorrect TRANS_PEEK_NMPIPE request being sent immediately before. With these two detections, we can know that an attacker is checking the "EternalBlue" vulnerability, and judge that the checked target is vulnerable.

The snort definition based on this is as follows:

```
alert tcp any any -> any 445 (msg:"EternalBlue Vulnerable check request";
content:"|ff|SMB|25|"; offset:4; content:"|23 00 00 00|"; distance:56;
sid:1000002; rev:1;)
```



```

alert tcp any 445 -> any any (msg:"EternalBlue Vulnerable check return";
content:"|ff|SMB";offset:4; content:"|25 05 02 00 c0|"; distance:0;
sid:1000003; rev:1;)

```

3.3. Hit transmission of Trans2 SESSION_SETUP by the “DoublePulsar”

The WannaCry's worm infects the “DoublePulsar” backdoor by exploiting the “EternalBlue” on a vulnerable target. After that, the WannaCry sends a payload to infect target with itself. At this time we know that the Trans2 SESSION_SETUP command is being used. According to the manual of CIFS of Microsoft, this command is not implemented and is not used. For this reason, it is unthinkable to use this subcommand in a normal application, and I will consider that communication by the “DoublePulsar” can be detected by detecting the communication of this subcommand. On the other hand, the payload sent by the “DoublePulsar” is encrypted with 32 bit XOR. It is difficult to create

2.2.6.15 TRANS2_SESSION_SETUP (0x000E)

This Transaction2 subcommand was introduced in the **NT LAN Manager** dialect. This subcommand is reserved but not implemented.

Clients SHOULD NOT send requests using this command code. Servers receiving requests with this command code SHOULD return STATUS_NOT_IMPLEMENTED (ERRDOS/ERRbadfunc).

(Quoted from the manual "[MS-CIFS]:Common Internet File System (CIFS) Protocol")

a signature.

Figure 13 Description of TRANS 2 _SESSION SETUP subcommand on CIFS manual

I will consider the grounds of the detection rule as follows:

- Communication is “SMB”
- Request command is SMB_COM_TRANSACTION2 (0x32h)
- Request subcommand is TRANS2_SESSION_SETUP (0x000e)

I considered whether there is a possibility of false positive detection under this condition. According to the CIFS manual, "Common Internet File System (CIFS) Protocol," the TRANS2_SESSION_SETUP subcommand is not implemented and can't be used by ordinary applications. For this reason, I think that if a communication using

Hirokazu Murakami, h-murakami@sig-c.co.jp

this subcommand is detected, there is no problem even if alerting on it. If an attacker changes a subcommand accepted by a new variant, it can't be detected by this rule. However, I think that it is difficult to change, for setting up a backdoor without affecting the SMB function. Also, high skill is required to change the code. Attackers below a certain level would consider continuing to use the existing "DoublePulsar."

The snort definition based on this is as follows:

```
alert tcp any any -> any 445 (msg:"Send Data is possible to send to
DoublePulsar Backdoor"; content:"|ff|SMB|32|"; offset:4; content:"|0e 00|";
distance:56; sid:1000004; rev:1;)
```

3.4. Hit request to check whether the "DoublePulsar" is infected

I considered a method which can judge communication of the "DoublePulsar" in detail. This will be the index of the progress of the attack. The WannaCry refers to the parameter "Multiplex ID" and judges whether it is infected or not. For this reason, I hypothesized that the "DoublePulsar" may use "Multiplex ID" for exchanging commands during sending and receiving. I compared the communication of checking for infection and the payload communication. The "Multiplex ID" of checking infection is 0x41, and the "Multiplex ID" for payload communication is 0x42.

I will consider the detection rule for requests checking whether the "DoublePulsar" has been infected as follow:

- Communication is "SMB"
- Request command is SMB_COM_TRANSACTION2 (0x32h)
- Request subcommand is TRANS2_SESSION_SETUP (0x000e)
- Multiplex ID is 0x41

I considered whether there is a possibility of false positive detection under this condition. In the future, when making variants, there is a possibility to change the value of the command. In this case, detection can't be performed. On the other hand, it

becomes clear that the attacker is taking the action of checking whether the “DoublePulsar” is infected or not.

The snort definition based on this is as follows:

```
alert tcp any any -> any 445 (msg:"Possible DoublePulsar infection check request"; content:"|ff|SMB|32|"; offset:4; content:"|41 00|"; distance:25; content:"|0e 00|";distance:29; sid:1000005; rev:1;)
```

3.5. Hit response when the “DoublePulsar” is infected

I considered how to hit the response of the “DoublePulsar” infection check request. As I pointed in section 2.2, there is a process of judging whether the value of the “Multiplex ID” of received data is 0x51h or not in the WannaCry. If the “Multiplex ID” remains 0x41h, we can know that the target is not infected by the “DoublePulsar.” As a problem, there is no parameter in the received data that can determine which subcommand's response to the request. For this reason, I will substitute another parameter. I know that we will return STATUS_NOT_IMPLEMENTED (0xc0000002) status in this response. This status code is returned only when calling an unimplemented subcommand, it is not expected to occur in normal communication. It is judged by this parameter and the “Multiplex ID” is 0x51h. If the “DoublePulsar” infection check request and this detection are hit, we can judge that the target machine is likely to be infected with the “DoublePulsar” backdoor. I will consider the grounds of the detection rule as follows:

- Communication is “SMB”
- Request command is SMB_COM_TRANSACTION2 (0x32h)
- Return status is STATUS_NOT_IMPLEMENTED (0xc0000002)
- Multiplex ID is 0x51

I considered whether there is a possibility of false positive detection under this condition. There is a possibility of false positive detection by matching with a response of completely different request. It is necessary to judge whether it is false positive detection or not by using in combination with the request of 3.4.

The snort definition based on this is as follows:

```

alert tcp any 445 -> any any (msg:"Possible DoublePulsar infected response";
content:"|ff|SMB|32|"; offset:4; content:"|02 00 00 c0|"; distance:0;
content:"|51 00|"; distance:21; sid:1000006; rev:1;)

```

3.6. Hit the communication sending payload to the “DoublePulsar”

Because the payload data is encrypted 32 bits XOR whose key is changed each time, detection is difficult by payload data. But, the subcommand and the “Multiplex ID” in the metadata does not change, I think that it can be detected. If this is detected, we can see that the payload is being sent and malware infection is coming up. If this communication is cut off, it will also be possible to stop malware infections at the moment. It is necessary to compare the possibility of detection miss and malware infection prevention.

I will consider the grounds of the detection rule as follows:

- Communication is "SMB"
- Request command is SMB_COM_TRANSACTION2 (0x32h)
- Request subcommand is TRANS2_SESSION_SETUP (0x000e)
- Multiplex ID is 0x42

The possibility of false positive detection under this condition is the same as 3.4.

The snort definition based on this is as follows:

```

alert tcp any any -> any 445 (msg:"Possible send Payload to DoublePulsar";
content:"|ff|SMB|32|"; offset:4; content:"|42 00|"; distance:25;
content:"|0e 00|"; distance:29; sid:1000007; rev:1;)

```

4. Operation verification of Snort definition

I verified Snort definition of the “EternalBlue” and the “DoublePulsar.” I made two kinds of verification. One, I generated the “EternalBlue” exploit communication using Kali Linux's Metasploit framework. The next step, I generated the “EternalBlue” and the “DoublePulsar” communication by infecting the target machine with the WannaCry.

The environment is as follows:

- Target machine A (Windows 7: EternalBlue vulnerability addressed) 192.168.1.11
- Target machine B (Windows 7: EternalBlue vulnerable) 192.168.1.12
- Machines running Snort (Ubuntu 16.04: Snort 2.9.7.0) 192.168.1.10
- Machine running Kali Linux (Kali 2017.2, EternalBlue exploit package applied) 192.168.1.13
- WannaCry infected terminal (Windows 7) 192.168.1.21

4.1. Communication of normal data

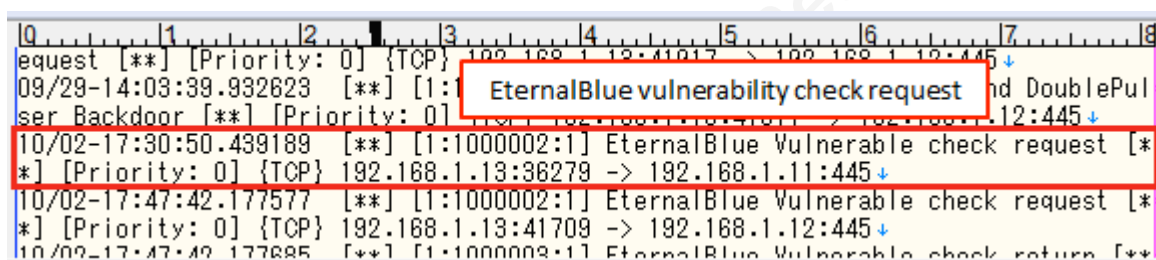
I tried to send a jpg file from target machine B to target machine A. As a result, Snort did not detect anything. Also, we can see that file transmission and reception between Windows 7 is SMB2 protocol. For commonly supported Windows to Windows communication, SMB2 will be preferentially used when not specifically configured. For this reason, if the snort definition is based on SMB1, I consider that there are not false positive detections in the normal operation in commonly supported Windows.

No.	Time	Source	Destination	Protocol	Length	Info	Dest Port
84	0.499356	192.168.1...	192.168.1...	SMB2	146	Close Request File:	445
85	0.499398	192.168.1...	192.168.1...	SMB2	182	Close Response	53708
86	0.499810	192.168.1...	192.168.1...	SMB2	410	Create Request File: 01_kokoro_sachiel_004.jpg	445
87	0.500065	192.168.1...	192.168.1...	SMB2	386	Create Response File: 01_kokoro_sachiel_004.jpg	53708
88	0.500406	192.168.1...	192.168.1...	SMB2	162	GetInfo Request FS_INFO/FileFsSizeInformation File: ...	445
89	0.500443	192.168.1...	192.168.1...	SMB2	154	GetInfo Response	53708

Figure 14 Normal file transmission / reception by SMB

4.2. The “EternalBlue” vulnerability check for not vulnerable machine

I tried to check communication of the “EternalBlue” vulnerability from Kali Linux to target machine A without a vulnerability. As a result, I confirmed that it hit the “the EternalBlue vulnerability check request” definition.



```

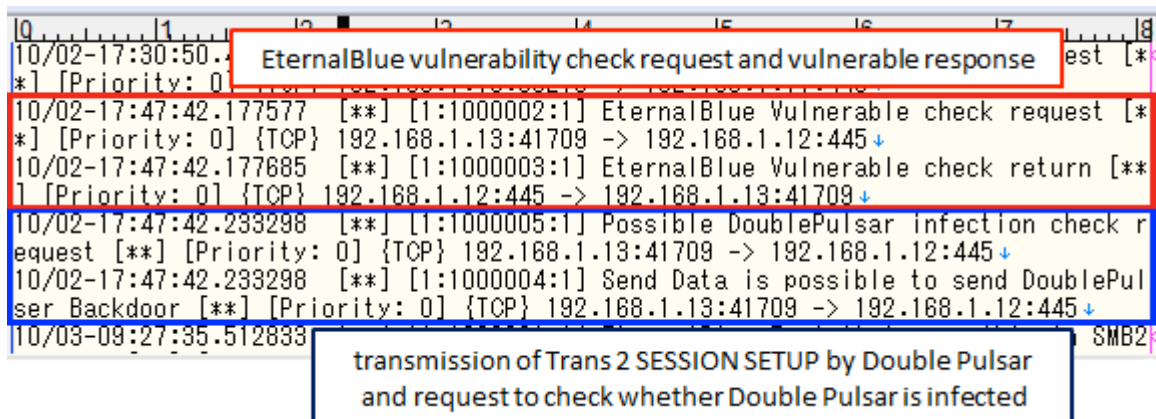
10/02-17:30:50.439189 192.168.1.13:36279 -> 192.168.1.11:445
10/02-17:47:42.177577 192.168.1.13:41709 -> 192.168.1.12:445
10/02-17:47:42.177685 192.168.1.13:41709 -> 192.168.1.12:445

```

Figure 15 Detection of checks on target machine without “EternalBlue” vulnerability

4.3. The “EternalBlue” vulnerability check for vulnerable machine

I tried the check communication of the “EternalBlue” vulnerability from Kali Linux to the vulnerable target machine B. As a result, I confirmed that it hit the “the EternalBlue vulnerability check request and vulnerable response” definition. And I confirmed that it hit the definition of “transmission of Trans 2 SESSION SETUP by the DoublePulsar” and the “request to check whether the DoublePulsar is infected” after that. This log shows that Kali Linux was acting as “checking the vulnerability of the EternalBlue and checking whether it is infected with the DoublePulsar if judging that it is vulnerable.”



```

10/02-17:30:50.439189 192.168.1.13:36279 -> 192.168.1.11:445
10/02-17:47:42.177577 192.168.1.13:41709 -> 192.168.1.12:445
10/02-17:47:42.177685 192.168.1.13:41709 -> 192.168.1.12:445
10/02-17:47:42.233298 192.168.1.13:41709 -> 192.168.1.12:445
10/02-17:47:42.233298 192.168.1.13:41709 -> 192.168.1.12:445
10/03-09:27:35.512833 192.168.1.13:41709 -> 192.168.1.12:445

```

Figure 16 Detection of checks on target machine with EternalBlue vulnerability

4.4. The “EternalBlue” exploits for vulnerable machines

I tried the “EternalBlue” exploit from Kali Linux to vulnerable target machine B. In this case, the “DoublePulsar” was not used for the payload. As a result of the exploit, it hit the definition “illegal SMB2 transmission sent when sending the EternalBlue exploit” multiple times. This is due to sending multiple times by the “EternalBlue” exploits. There is a problem that many logs appear in one exploit. But I have found that it can be detected. If this detection is found after detection of “vulnerable,” it is possible that an attack was made and it was successful. In this case, I think that it will be the reason to raise the priority of investigation.

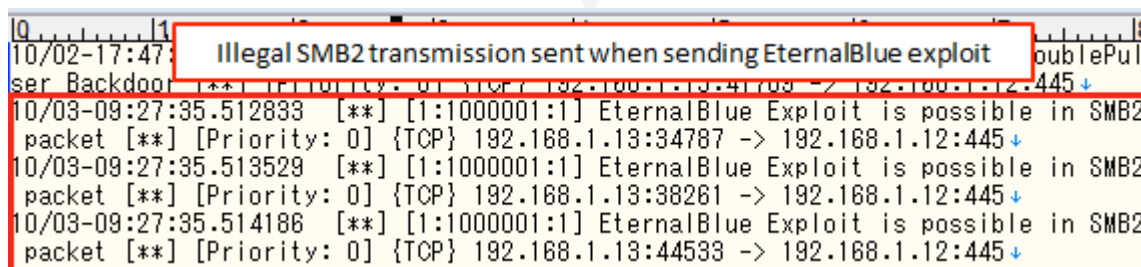


Figure 17 Detection log when exploited by vulnerability of “EternalBlue”

4.5. WannaCry's attack against a vulnerable machine

I attacked target machine B with the WannaCry specimen analyzed in this paper. After the exploit is executed, we can see that the WannaCry infected target machine B in the process situation. The snort log at this time is as shown in Figure 18 and Figure 19.

First, the WannaCry checked if there is a vulnerability in the “EternalBlue.” In addition, it returned a response in case of the vulnerability. This hit the definition “the EternalBlue vulnerability check request and vulnerable response” (Figure 18 red rectangle).

Second, the WannaCry checked whether the “DoublePulsar” was infected at that time. In this case, only the request was detected because it was uninfected. This hit the definition “transmission of Trans 2 SESSION SETUP by the DoublePulsar” and “request to check whether the “DoublePulsar” is infected” (Figure 18 blue rectangle).

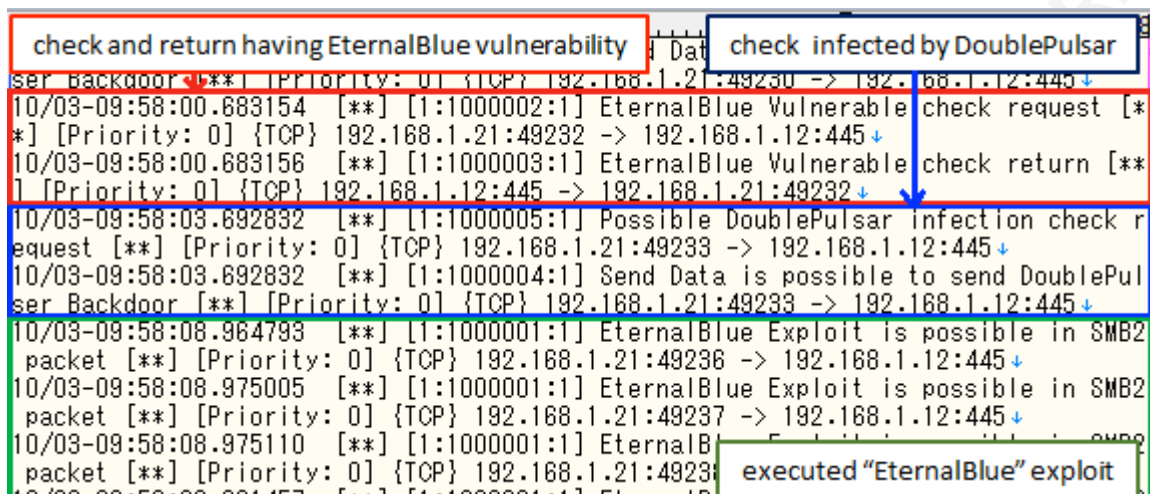


Figure 18 Detection of WannaCry infection (Part 1)

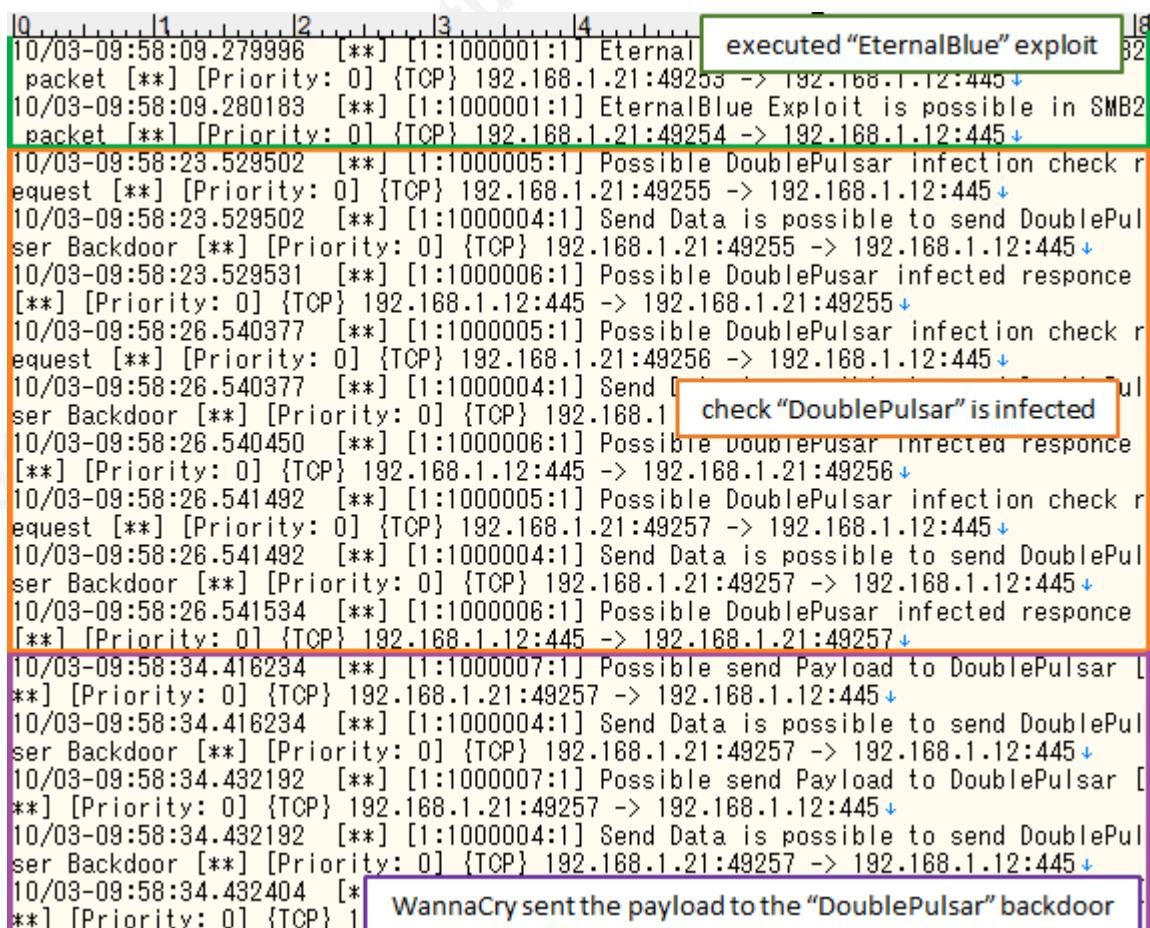


Figure 19 Detection of WannaCry infection (Part 2)

Third, WannaCry executed “EternalBlue” exploit. This hit the definition "illegal SMB 2 transmission sent time sending the EternalBlue exploit" (Figure 18 green rectangle).

Fourth, the WannaCry again checked whether the “DoublePulsar” was infected at that time. In this case, requests and responses were detected due to infection. This hit the definition “request to check whether the DoublePulsar is infected” and “response when the DoublePulsar is infected”. It also hit the detection “transmission of Trans2 SESSION_SETUP by the DoublePulsar” at the same time. The malware repeatedly checked 3 times, it detected three times (Figure 19 orange rectangle).

Finally, the WannaCry sent the payload to the “DoublePulsar” backdoor. This hit the definition “the communication sending payload to the DoublePulsar.” It also hit the detection "transmission of Trans2 SESSION_SETUP by the DoublePulsar" at the same time (Figure 19 purple rectangle).

4.6. Operation verification result

As a result, while it was difficult to see due to malware transmitting it more than once, we could detect communication the “EternalBlue” and the “DoublePulsar” in this Snort definition. And we were able to capture the behavior of the WannaCry infection. By combining multiple detection rules based on the characteristics obtained by analysis, I think that we can reduce false positives and capture signs of attacks.

5. Conclusion

I analyzed results from the WannaCry in a simulation with the minimum verification environment. As a result, I found that there is a possibility to detect not only the WannaCry but also malware communication using the “EternalBlue” and the “DoublePulsar.” With these rules, I think that it is possible to stop attacks by blocking by IPS those attached that are not likely to be false positives. For example, I think that attacks can be stopped if you interrupt illegal SMB2 communication of the “EternalBlue.” Besides, I think that you can prevent infection by blocking DoublePulsar's payload transmission.

In creating this rule, we aimed to be as versatile as possible by using analysis results. I focused on communication metadata, focusing on the unique “habit” of exploit and backdoor communication. Even if an attacker made a subspecies, I attempted to capture the part that can't be changed as an exploit's technique or the backdoor function, or that is difficult to change.

The problem is that detailed analysis is necessary, but it takes a very long time. In recent years there is a tendency to produce many variants. So I think that it is difficult to apply all of them to detailed analysis. For this reason, we shall use multiple security solutions and deal with it in additional layers, if anti-virus software etc. is not enough. In a long-lived or serious case, we need to consider the response to do a detailed analysis.

If there are multiple subspecies but the mechanism of action is the same malware, I think that it has a “vital point.” In this case, I think that usage of the “EternalBlue” and the “DoublePulsar” will be “vital points.” Once we hold down this, even if an attacker creates a similar subspecies, we can handle it all. I think that we can “Rot malware with the root.” From this point of view, I believe that it is worthwhile to analyze one representative specimen firmly.

References

- Trend Micro Corp. *MS17-010: EternalBlue's Large Non-Paged Pool Overflow in SRV Driver* (<http://blog.trendmicro.com/trendlabs-security-intelligence/ms17-010-eternalblue/>)
- Microsoft Corp. *[MS-CIFS]: Common Internet File System (CIFS) Protocol* (<https://msdn.microsoft.com/en-us/library/ee442092.aspx>)
- Microsoft Corp. *[MS-SMB2]: Server Message Block (SMB) Protocol Versions 2 and 3* (<https://msdn.microsoft.com/ja-jp/library/cc246482.aspx>)
- Snort Project. *Snort - Network Intrusion Detection & Prevention System* (<https://www.snort.org/>)
- Offensive Security. *Kali Linux | Penetration Testing and Ethical Hacking Linux Distribution* (<https://www.kali.org/>)
- 新井 悠(Arai, Y.), 岩村 誠(Iwamura, M.), 川古谷 裕平(Kawakoya, Y.), 青木 一司(Aoki, K.), & 星澤 裕二(Hoshizawa, Y.) (2010).
アナライジング・マルウェア フリーツールを使った感染事案対処
(*Analyzing Malware : Fighting against infection incidents with free tools*).
- SIG Corp. (2017) *EternalBlue と DoublePulsar の通信の検知の試み* (*Attempt to detect EternalBlue and DoublePulsar communications*) Retrieved January 5, 2018 from <https://www.sig-c.co.jp/services/column/analysis-eternalblue.php>

Appendix

EternalBlue and DoublePulsar detection snort log

```

10/02-17:30:50.439189 [**] [1:1000002:1] EternalBlue Vulnerable check request [**] [Priority: 0] {TCP}
192.168.1.13:36279 -> 192.168.1.11:445
10/02-17:47:42.177577 [**] [1:1000002:1] EternalBlue Vulnerable check request [**] [Priority: 0] {TCP}
192.168.1.13:41709 -> 192.168.1.12:445
10/02-17:47:42.177685 [**] [1:1000003:1] EternalBlue Vulnerable check return [**] [Priority: 0] {TCP}
192.168.1.12:445 -> 192.168.1.13:41709
10/02-17:47:42.233298 [**] [1:1000005:1] Possible DoublePulsar infection check request [**] [Priority:
0] {TCP} 192.168.1.13:41709 -> 192.168.1.12:445
10/02-17:47:42.233298 [**] [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor [**]
[Priority: 0] {TCP} 192.168.1.13:41709 -> 192.168.1.12:445
10/03-09:27:35.512833 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority:
0] {TCP} 192.168.1.13:34787 -> 192.168.1.12:445
10/03-09:27:35.513529 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority:
0] {TCP} 192.168.1.13:38261 -> 192.168.1.12:445
10/03-09:27:35.514186 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority:
0] {TCP} 192.168.1.13:44533 -> 192.168.1.12:445
10/03-09:27:35.514794 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority:
0] {TCP} 192.168.1.13:39839 -> 192.168.1.12:445
10/03-09:27:35.515458 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority:
0] {TCP} 192.168.1.13:38077 -> 192.168.1.12:445
10/03-09:27:35.516067 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority:
0] {TCP} 192.168.1.13:41383 -> 192.168.1.12:445
10/03-09:27:35.522071 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority:
0] {TCP} 192.168.1.13:41817 -> 192.168.1.12:445
10/03-09:27:35.523195 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority:
0] {TCP} 192.168.1.13:40379 -> 192.168.1.12:445
10/03-09:27:35.524150 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority:
0] {TCP} 192.168.1.13:38855 -> 192.168.1.12:445
10/03-09:27:35.525186 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority:
0] {TCP} 192.168.1.13:36877 -> 192.168.1.12:445
10/03-09:27:35.526243 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority:
0] {TCP} 192.168.1.13:37487 -> 192.168.1.12:445
10/03-09:27:35.526967 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority:
0] {TCP} 192.168.1.13:45675 -> 192.168.1.12:445
10/03-09:27:35.534571 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority:
0] {TCP} 192.168.1.13:32877 -> 192.168.1.12:445
10/03-09:27:35.544214 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority:
0] {TCP} 192.168.1.13:35039 -> 192.168.1.12:445
10/03-09:27:35.545252 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority:
0] {TCP} 192.168.1.13:35111 -> 192.168.1.12:445
10/03-09:27:35.546054 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority:
0] {TCP} 192.168.1.13:38759 -> 192.168.1.12:445
10/03-09:27:35.547065 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority:
0] {TCP} 192.168.1.13:44103 -> 192.168.1.12:445
10/03-09:27:35.548040 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority:
0] {TCP} 192.168.1.13:37003 -> 192.168.1.12:445
10/03-09:53:05.264450 [**] [1:1000002:1] EternalBlue Vulnerable check request [**] [Priority: 0] {TCP}
192.168.1.21:49199 -> 192.168.1.12:445
10/03-09:53:05.264493 [**] [1:1000003:1] EternalBlue Vulnerable check return [**] [Priority: 0] {TCP}
192.168.1.12:445 -> 192.168.1.21:49199
10/03-09:53:08.265124 [**] [1:1000005:1] Possible DoublePulsar infection check request [**] [Priority:
0] {TCP} 192.168.1.21:49200 -> 192.168.1.12:445
10/03-09:53:08.265124 [**] [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor [**]
[Priority: 0] {TCP} 192.168.1.21:49200 -> 192.168.1.12:445

```

```

10/03-09:53:13.552606 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49203 -> 192.168.1.12:445
10/03-09:53:13.563006 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49204 -> 192.168.1.12:445
10/03-09:53:13.563114 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49205 -> 192.168.1.12:445
10/03-09:53:13.579388 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49206 -> 192.168.1.12:445
10/03-09:53:13.615335 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49207 -> 192.168.1.12:445
10/03-09:53:13.615534 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49208 -> 192.168.1.12:445
10/03-09:53:13.643470 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49209 -> 192.168.1.12:445
10/03-09:53:13.643652 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49210 -> 192.168.1.12:445
10/03-09:53:13.678366 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49211 -> 192.168.1.12:445
10/03-09:53:13.693350 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49212 -> 192.168.1.12:445
10/03-09:53:13.709295 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49213 -> 192.168.1.12:445
10/03-09:53:13.719775 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49214 -> 192.168.1.12:445
10/03-09:53:13.739924 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49215 -> 192.168.1.12:445
10/03-09:53:13.809569 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49217 -> 192.168.1.12:445
10/03-09:53:13.840900 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49218 -> 192.168.1.12:445
10/03-09:53:13.857381 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49219 -> 192.168.1.12:445
10/03-09:53:13.872129 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49220 -> 192.168.1.12:445
10/03-09:53:13.872326 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49221 -> 192.168.1.12:445
10/03-09:54:55.227738 [**] [1:1000005:1] Possible DoublePulsar infection check request [**] [Priority: 0]
{TCP} 192.168.1.21:49230 -> 192.168.1.12:445
10/03-09:54:55.227738 [**] [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor [**]
[Priority: 0] {TCP} 192.168.1.21:49230 -> 192.168.1.12:445
10/03-09:58:00.683154 [**] [1:1000002:1] EternalBlue Vulnerable check request [**] [Priority: 0] {TCP}
192.168.1.21:49232 -> 192.168.1.12:445
10/03-09:58:00.683156 [**] [1:1000003:1] EternalBlue Vulnerable check return [**] [Priority: 0] {TCP}
192.168.1.12:445 -> 192.168.1.21:49232
10/03-09:58:03.692832 [**] [1:1000005:1] Possible DoublePulsar infection check request [**] [Priority: 0]
{TCP} 192.168.1.21:49233 -> 192.168.1.12:445
10/03-09:58:03.692832 [**] [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor [**]
[Priority: 0] {TCP} 192.168.1.21:49233 -> 192.168.1.12:445
10/03-09:58:08.964793 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49236 -> 192.168.1.12:445
10/03-09:58:08.975005 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49237 -> 192.168.1.12:445
10/03-09:58:08.975110 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49238 -> 192.168.1.12:445
10/03-09:58:08.991457 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49239 -> 192.168.1.12:445
10/03-09:58:09.026637 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49240 -> 192.168.1.12:445
10/03-09:58:09.026822 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49241 -> 192.168.1.12:445

```

```

10/03-09:58:09.055023 /** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet /** [Priority: 0]
{TCP} 192.168.1.21:49242 -> 192.168.1.12:445
10/03-09:58:09.055232 /** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet /** [Priority: 0]
{TCP} 192.168.1.21:49243 -> 192.168.1.12:445
10/03-09:58:09.089934 /** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet /** [Priority: 0]
{TCP} 192.168.1.21:49244 -> 192.168.1.12:445
10/03-09:58:09.104759 /** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet /** [Priority: 0]
{TCP} 192.168.1.21:49245 -> 192.168.1.12:445
10/03-09:58:09.120676 /** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet /** [Priority: 0]
{TCP} 192.168.1.21:49246 -> 192.168.1.12:445
10/03-09:58:09.131197 /** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet /** [Priority: 0]
{TCP} 192.168.1.21:49247 -> 192.168.1.12:445
10/03-09:58:09.151200 /** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet /** [Priority: 0]
{TCP} 192.168.1.21:49248 -> 192.168.1.12:445
10/03-09:58:09.219927 /** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet /** [Priority: 0]
{TCP} 192.168.1.21:49250 -> 192.168.1.12:445
10/03-09:58:09.251135 /** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet /** [Priority: 0]
{TCP} 192.168.1.21:49251 -> 192.168.1.12:445
10/03-09:58:09.268085 /** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet /** [Priority: 0]
{TCP} 192.168.1.21:49252 -> 192.168.1.12:445
10/03-09:58:09.279996 /** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet /** [Priority: 0]
{TCP} 192.168.1.21:49253 -> 192.168.1.12:445
10/03-09:58:09.280183 /** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet /** [Priority: 0]
{TCP} 192.168.1.21:49254 -> 192.168.1.12:445
10/03-09:58:23.529502 /** [1:1000005:1] Possible DoublePulsar infection check request /** [Priority: 0]
{TCP} 192.168.1.21:49255 -> 192.168.1.12:445
10/03-09:58:23.529502 /** [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor /**
[Priority: 0] {TCP} 192.168.1.21:49255 -> 192.168.1.12:445
10/03-09:58:23.529531 /** [1:1000006:1] Possible DoublePulsar infected response /** [Priority: 0] {TCP}
192.168.1.12:445 -> 192.168.1.21:49255
10/03-09:58:26.540377 /** [1:1000005:1] Possible DoublePulsar infection check request /** [Priority: 0]
{TCP} 192.168.1.21:49256 -> 192.168.1.12:445
10/03-09:58:26.540377 /** [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor /**
[Priority: 0] {TCP} 192.168.1.21:49256 -> 192.168.1.12:445
10/03-09:58:26.540450 /** [1:1000006:1] Possible DoublePulsar infected response /** [Priority: 0] {TCP}
192.168.1.12:445 -> 192.168.1.21:49256
10/03-09:58:26.541492 /** [1:1000005:1] Possible DoublePulsar infection check request /** [Priority: 0]
{TCP} 192.168.1.21:49257 -> 192.168.1.12:445
10/03-09:58:26.541492 /** [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor /**
[Priority: 0] {TCP} 192.168.1.21:49257 -> 192.168.1.12:445
10/03-09:58:26.541534 /** [1:1000006:1] Possible DoublePulsar infected response /** [Priority: 0] {TCP}
192.168.1.12:445 -> 192.168.1.21:49257
10/03-09:58:34.416234 /** [1:1000007:1] Possible send Payload to DoublePulsar /** [Priority: 0] {TCP}
192.168.1.21:49257 -> 192.168.1.12:445
10/03-09:58:34.416234 /** [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor /**
[Priority: 0] {TCP} 192.168.1.21:49257 -> 192.168.1.12:445
10/03-09:58:34.432192 /** [1:1000007:1] Possible send Payload to DoublePulsar /** [Priority: 0] {TCP}
192.168.1.21:49257 -> 192.168.1.12:445
10/03-09:58:34.432192 /** [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor /**
[Priority: 0] {TCP} 192.168.1.21:49257 -> 192.168.1.12:445
10/03-09:58:34.432404 /** [1:1000007:1] Possible send Payload to DoublePulsar /** [Priority: 0] {TCP}
192.168.1.21:49257 -> 192.168.1.12:445
10/03-09:58:34.432404 /** [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor /**
[Priority: 0] {TCP} 192.168.1.21:49257 -> 192.168.1.12:445
10/03-09:58:34.555353 /** [1:1000007:1] Possible send Payload to DoublePulsar /** [Priority: 0] {TCP}
192.168.1.21:49257 -> 192.168.1.12:445
10/03-09:58:34.555353 /** [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor /**
[Priority: 0] {TCP} 192.168.1.21:49257 -> 192.168.1.12:445
10/03-09:58:34.556895 /** [1:1000007:1] Possible send Payload to DoublePulsar /** [Priority: 0] {TCP}
192.168.1.21:49257 -> 192.168.1.12:445

```

```

10/03-09:58:34.556895 ** [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor **
[Priority: 0] {TCP} 192.168.1.21:49257 -> 192.168.1.12:445
10/03-09:58:34.611741 ** [1:1000007:1] Possible send Payload to DoublePulsar ** [Priority: 0] {TCP}
192.168.1.21:49257 -> 192.168.1.12:445
10/03-09:58:34.611741 ** [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor **
[Priority: 0] {TCP} 192.168.1.21:49257 -> 192.168.1.12:445
10/03-09:59:16.846672 ** [1:1000002:1] EternalBlue Vulnerable check request ** [Priority: 0] {TCP}
192.168.1.12:49170 -> 192.168.1.11:445
10/03-09:59:19.901617 ** [1:1000005:1] Possible DoublePulsar infection check request ** [Priority: 0]
{TCP} 192.168.1.12:49199 -> 192.168.1.11:445
10/03-09:59:19.901617 ** [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor **
[Priority: 0] {TCP} 192.168.1.12:49199 -> 192.168.1.11:445
10/03-12:01:29.978198 ** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet ** [Priority: 0]
{TCP} 192.168.1.13:42673 -> 192.168.1.12:445
10/03-12:01:29.978872 ** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet ** [Priority: 0]
{TCP} 192.168.1.13:43689 -> 192.168.1.12:445
10/03-12:01:29.979558 ** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet ** [Priority: 0]
{TCP} 192.168.1.13:42023 -> 192.168.1.12:445
10/03-12:01:29.980234 ** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet ** [Priority: 0]
{TCP} 192.168.1.13:40343 -> 192.168.1.12:445
10/03-12:01:29.980939 ** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet ** [Priority: 0]
{TCP} 192.168.1.13:41999 -> 192.168.1.12:445
10/03-12:01:29.981592 ** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet ** [Priority: 0]
{TCP} 192.168.1.13:39949 -> 192.168.1.12:445
10/03-12:01:29.982227 ** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet ** [Priority: 0]
{TCP} 192.168.1.13:40277 -> 192.168.1.12:445
10/03-12:01:29.983687 ** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet ** [Priority: 0]
{TCP} 192.168.1.13:43001 -> 192.168.1.12:445
10/03-12:01:29.984360 ** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet ** [Priority: 0]
{TCP} 192.168.1.13:36811 -> 192.168.1.12:445
10/03-12:01:29.985439 ** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet ** [Priority: 0]
{TCP} 192.168.1.13:45765 -> 192.168.1.12:445
10/03-12:01:29.986598 ** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet ** [Priority: 0]
{TCP} 192.168.1.13:44067 -> 192.168.1.12:445
10/03-12:01:29.990519 ** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet ** [Priority: 0]
{TCP} 192.168.1.13:33309 -> 192.168.1.12:445
10/03-12:01:29.996519 ** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet ** [Priority: 0]
{TCP} 192.168.1.13:43965 -> 192.168.1.12:445
10/03-12:01:29.997113 ** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet ** [Priority: 0]
{TCP} 192.168.1.13:44559 -> 192.168.1.12:445
10/03-12:01:29.997757 ** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet ** [Priority: 0]
{TCP} 192.168.1.13:35105 -> 192.168.1.12:445
10/03-12:01:30.003536 ** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet ** [Priority: 0]
{TCP} 192.168.1.13:36505 -> 192.168.1.12:445
10/03-12:01:30.004924 ** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet ** [Priority: 0]
{TCP} 192.168.1.13:39803 -> 192.168.1.12:445
10/03-12:01:30.006470 ** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet ** [Priority: 0]
{TCP} 192.168.1.13:32815 -> 192.168.1.12:445
10/03-16:23:13.930717 ** [1:1000002:1] EternalBlue Vulnerable check request ** [Priority: 0] {TCP}
192.168.1.13:40439 -> 192.168.1.12:445
10/03-16:23:13.930766 ** [1:1000003:1] EternalBlue Vulnerable check return ** [Priority: 0] {TCP}
192.168.1.12:445 -> 192.168.1.13:40439
10/03-16:23:14.006456 ** [1:1000005:1] Possible DoublePulsar infection check request ** [Priority: 0]
{TCP} 192.168.1.13:40439 -> 192.168.1.12:445
10/03-16:23:14.006456 ** [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor **
[Priority: 0] {TCP} 192.168.1.13:40439 -> 192.168.1.12:445
10/03-16:34:39.527673 ** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet ** [Priority: 0]
{TCP} 192.168.1.13:37395 -> 192.168.1.12:445
10/03-16:34:39.528403 ** [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet ** [Priority: 0]
{TCP} 192.168.1.13:45835 -> 192.168.1.12:445

```

```

10/03-16:34:39.529108 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.13:41067 -> 192.168.1.12:445
10/03-16:34:39.529848 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.13:39229 -> 192.168.1.12:445
10/03-16:34:39.530580 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.13:34999 -> 192.168.1.12:445
10/03-16:34:39.531189 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.13:33057 -> 192.168.1.12:445
10/03-16:34:39.531751 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.13:33675 -> 192.168.1.12:445
10/03-16:34:39.532298 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.13:37729 -> 192.168.1.12:445
10/03-16:34:39.532888 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.13:41365 -> 192.168.1.12:445
10/03-16:34:39.540878 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.13:34477 -> 192.168.1.12:445
10/03-16:34:39.541666 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.13:46043 -> 192.168.1.12:445
10/03-16:34:39.542319 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.13:44607 -> 192.168.1.12:445
10/03-16:34:39.547513 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.13:39443 -> 192.168.1.12:445
10/03-16:34:39.548161 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.13:43241 -> 192.168.1.12:445
10/03-16:34:39.548788 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.13:42171 -> 192.168.1.12:445
10/03-16:34:39.550253 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.13:36995 -> 192.168.1.12:445
10/03-16:34:39.550931 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.13:40293 -> 192.168.1.12:445
10/03-16:34:39.551574 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.13:45683 -> 192.168.1.12:445
10/03-16:47:55.284043 [**] [1:1000002:1] EternalBlue Vulnerable check request [**] [Priority: 0] {TCP}
192.168.1.21:49259 -> 192.168.1.12:445
10/03-16:47:55.284116 [**] [1:1000003:1] EternalBlue Vulnerable check return [**] [Priority: 0] {TCP}
192.168.1.12:445 -> 192.168.1.21:49259
10/03-16:47:58.290865 [**] [1:1000005:1] Possible DoublePulsar infection check request [**] [Priority: 0]
{TCP} 192.168.1.21:49260 -> 192.168.1.12:445
10/03-16:47:58.290865 [**] [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor [**]
[Priority: 0] {TCP} 192.168.1.21:49260 -> 192.168.1.12:445
10/03-16:48:03.562124 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49263 -> 192.168.1.12:445
10/03-16:48:03.572424 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49264 -> 192.168.1.12:445
10/03-16:48:03.572521 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49265 -> 192.168.1.12:445
10/03-16:48:03.588832 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49266 -> 192.168.1.12:445
10/03-16:48:03.624894 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49267 -> 192.168.1.12:445
10/03-16:48:03.625065 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49268 -> 192.168.1.12:445
10/03-16:48:03.653079 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49269 -> 192.168.1.12:445
10/03-16:48:03.653298 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49270 -> 192.168.1.12:445
10/03-16:48:03.687882 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49271 -> 192.168.1.12:445
10/03-16:48:03.702818 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49272 -> 192.168.1.12:445

```

```

10/03-16:48:03.718711 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49273 -> 192.168.1.12:445
10/03-16:48:03.729130 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49274 -> 192.168.1.12:445
10/03-16:48:03.749287 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49275 -> 192.168.1.12:445
10/03-16:48:03.818014 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49277 -> 192.168.1.12:445
10/03-16:48:03.848877 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49278 -> 192.168.1.12:445
10/03-16:48:03.865516 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49279 -> 192.168.1.12:445
10/03-16:48:03.877801 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49280 -> 192.168.1.12:445
10/03-16:48:03.877987 [**] [1:1000001:1] EternalBlue Exploit is possible in SMB2 packet [**] [Priority: 0]
{TCP} 192.168.1.21:49281 -> 192.168.1.12:445
10/03-16:48:18.117792 [**] [1:1000005:1] Possible DoublePulsar infection check request [**] [Priority: 0]
{TCP} 192.168.1.21:49282 -> 192.168.1.12:445
10/03-16:48:18.117792 [**] [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor [**]
[Priority: 0] {TCP} 192.168.1.21:49282 -> 192.168.1.12:445
10/03-16:48:18.117880 [**] [1:1000006:1] Possible DoublePulsar infected response [**] [Priority: 0] {TCP}
192.168.1.12:445 -> 192.168.1.21:49282
10/03-16:48:21.128221 [**] [1:1000005:1] Possible DoublePulsar infection check request [**] [Priority: 0]
{TCP} 192.168.1.21:49283 -> 192.168.1.12:445
10/03-16:48:21.128221 [**] [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor [**]
[Priority: 0] {TCP} 192.168.1.21:49283 -> 192.168.1.12:445
10/03-16:48:21.128296 [**] [1:1000006:1] Possible DoublePulsar infected response [**] [Priority: 0] {TCP}
192.168.1.12:445 -> 192.168.1.21:49283
10/03-16:48:21.129165 [**] [1:1000005:1] Possible DoublePulsar infection check request [**] [Priority: 0]
{TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:21.129165 [**] [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor [**]
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:21.129204 [**] [1:1000006:1] Possible DoublePulsar infected response [**] [Priority: 0] {TCP}
192.168.1.12:445 -> 192.168.1.21:49284
10/03-16:48:29.317330 [**] [1:1000007:1] Possible send Payload to DoublePulsar [**] [Priority: 0] {TCP}
192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.317330 [**] [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor [**]
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.317544 [**] [1:1000007:1] Possible send Payload to DoublePulsar [**] [Priority: 0] {TCP}
192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.317544 [**] [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor [**]
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.317720 [**] [1:1000007:1] Possible send Payload to DoublePulsar [**] [Priority: 0] {TCP}
192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.317720 [**] [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor [**]
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.319068 [**] [1:1000007:1] Possible send Payload to DoublePulsar [**] [Priority: 0] {TCP}
192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.319068 [**] [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor [**]
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.320115 [**] [1:1000007:1] Possible send Payload to DoublePulsar [**] [Priority: 0] {TCP}
192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.320115 [**] [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor [**]
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.362976 [**] [1:1000007:1] Possible send Payload to DoublePulsar [**] [Priority: 0] {TCP}
192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.362976 [**] [1:1000004:1] Send Data is possible to send DoublePulsar Backdoor [**]
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.363169 [**] [1:1000007:1] Possible send Payload to DoublePulsar [**] [Priority: 0] {TCP}
192.168.1.21:49284 -> 192.168.1.12:445

```

```

10/03-16:48:29.363169 1000004:1 Send Data is possible to send DoublePulser Backdoor
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.364384 1000007:1 Possible send Payload to DoublePulsar
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.364384 1000004:1 Send Data is possible to send DoublePulser Backdoor
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.364671 1000007:1 Possible send Payload to DoublePulsar
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.364671 1000004:1 Send Data is possible to send DoublePulser Backdoor
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.364781 1000007:1 Possible send Payload to DoublePulsar
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.364781 1000004:1 Send Data is possible to send DoublePulser Backdoor
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.364969 1000007:1 Possible send Payload to DoublePulsar
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.364969 1000004:1 Send Data is possible to send DoublePulser Backdoor
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.369101 1000007:1 Possible send Payload to DoublePulsar
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.369101 1000004:1 Send Data is possible to send DoublePulser Backdoor
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.369261 1000007:1 Possible send Payload to DoublePulsar
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.369261 1000004:1 Send Data is possible to send DoublePulser Backdoor
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.369420 1000007:1 Possible send Payload to DoublePulsar
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.369420 1000004:1 Send Data is possible to send DoublePulser Backdoor
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.369576 1000007:1 Possible send Payload to DoublePulsar
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.369576 1000004:1 Send Data is possible to send DoublePulser Backdoor
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.369760 1000007:1 Possible send Payload to DoublePulsar
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.369760 1000004:1 Send Data is possible to send DoublePulser Backdoor
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.369933 1000007:1 Possible send Payload to DoublePulsar
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.369933 1000004:1 Send Data is possible to send DoublePulser Backdoor
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.450929 1000007:1 Possible send Payload to DoublePulsar
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.450929 1000004:1 Send Data is possible to send DoublePulser Backdoor
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.503408 1000007:1 Possible send Payload to DoublePulsar
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.503408 1000004:1 Send Data is possible to send DoublePulser Backdoor
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.503744 1000007:1 Possible send Payload to DoublePulsar
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.503744 1000004:1 Send Data is possible to send DoublePulser Backdoor
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.503927 1000007:1 Possible send Payload to DoublePulsar
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.503927 1000004:1 Send Data is possible to send DoublePulser Backdoor
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.567608 1000007:1 Possible send Payload to DoublePulsar
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445

```



```

10/03-16:48:29.567608 ** [1:1000004:1] Send Data is possible to send DoublePulser Backdoor **
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.628288 ** [1:1000007:1] Possible send Payload to DoublePulser ** [Priority: 0] {TCP}
192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.628288 ** [1:1000004:1] Send Data is possible to send DoublePulser Backdoor **
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.629025 ** [1:1000007:1] Possible send Payload to DoublePulser ** [Priority: 0] {TCP}
192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.629025 ** [1:1000004:1] Send Data is possible to send DoublePulser Backdoor **
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.629452 ** [1:1000007:1] Possible send Payload to DoublePulser ** [Priority: 0] {TCP}
192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.629452 ** [1:1000004:1] Send Data is possible to send DoublePulser Backdoor **
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.686462 ** [1:1000007:1] Possible send Payload to DoublePulser ** [Priority: 0] {TCP}
192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.686462 ** [1:1000004:1] Send Data is possible to send DoublePulser Backdoor **
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.690557 ** [1:1000007:1] Possible send Payload to DoublePulser ** [Priority: 0] {TCP}
192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:48:29.690557 ** [1:1000004:1] Send Data is possible to send DoublePulser Backdoor **
[Priority: 0] {TCP} 192.168.1.21:49284 -> 192.168.1.12:445
10/03-16:49:11.110946 ** [1:1000002:1] EternalBlue Vulnerable check request ** [Priority: 0] {TCP}
192.168.1.12:49172 -> 192.168.1.11:445
10/03-16:49:14.115172 ** [1:1000005:1] Possible DoublePulser infection check request ** [Priority: 0]
{TCP} 192.168.1.12:49208 -> 192.168.1.11:445
10/03-16:49:14.115172 ** [1:1000004:1] Send Data is possible to send DoublePulser Backdoor **
[Priority: 0] {TCP} 192.168.1.12:49208 -> 192.168.1.11:445
10/03-16:52:05.724450 ** [1:1000002:1] EternalBlue Vulnerable check request ** [Priority: 0] {TCP}
192.168.1.12:51268 -> 192.168.1.11:445
10/03-16:52:08.732899 ** [1:1000005:1] Possible DoublePulser infection check request ** [Priority: 0]
{TCP} 192.168.1.12:51301 -> 192.168.1.11:445
10/03-16:52:08.732899 ** [1:1000004:1] Send Data is possible to send DoublePulser Backdoor **
[Priority: 0] {TCP} 192.168.1.12:51301 -> 192.168.1.11:445

```