Neutralizing Tofsee Spambot – Part 3 | Network-based kill switch

spamhaus.com/resource-center/neutralizing-tofsee-spambot-part-3-network-based-kill-switch/

April 6, 2023

Here's the final post in our three-part series on how to protect against Tofsee malware. This blog post concentrates on using a network kill switch - causing an out-of-bounds read error, leading to Tofsee crashing.

Playing catch-up?

If you missed the first two posts in this series, they focus on malware vaccines. These are proactive measures helping prevent malware infections by patching vulnerabilities in the system or blocking known attack vectors. Malware vaccines are not dissimilar to medical vaccines that provide the body immunity to a particular disease.

The first malware vaccine revealed by our researchers, concentrated on the binary file and the second one centered around the InMemoryConfig store.

What is a malware kill switch?

A malware kill switch is a feature that some malware authors include in their code that enables them to shut down the malware or prevent it from causing harm under certain circumstances, such as if the malware is spreading too quickly, causing damage to critical systems, or should their operations be tracked or compromised.

A good example is the case of the WannaCry ransomware attack in 2017 [https://www.bbc.co.uk/news/technology-41753022]; a researcher discovered a kill switch that the cybercriminal had built into the malware. By registering a specific domain name, the researcher could trigger the kill switch and stop the malware from spreading further.

What is a network-based kill switch?

In some cases, security researchers or organizations can develop a network-based kill switch for a specific malware threat. This reactive measure allows security experts to remotely disable or shut down malware infections, allowing them to neutralize the threat quickly and effectively if it is detected.

While using a network-based malware kill switch can be an effective way to limit the damage caused by malware, it is key to note that it may not be a foolproof solution. Malware can evolve rapidly, and attackers may be able to find ways to work around a kill switch or develop

new malware that is not vulnerable to it. Therefore, it is essential to use a variety of security measures, such as anti-malware software, firewalls, and regular security updates, to protect against malware infections.

How can a network kill switch be implemented for Tofsee?

One way to render Tofsee useless and kill it without access to the remote infected machine is to locate a bug in its binary code and crash the malware.

The first part of this process is to get our data parsed by Tofsee, and to do this, we need to follow its protocol specification.

Tofsee's protocol specification

Communication is bi-directional and encrypted using a custom algorithm that requires two state keys. These state keys are specific to each **SocketConnection** in Tofsee and are modified based on each Transmission Control Protocol (TCP) data transfer between the botnet command and control server (C&C) and the infected bot. This is known as rolling key encryption.

```
unsigned char *TofseeCrypt(unsigned char *input, int len, struct __tofseerollingkey *RollKey)
{
    unsigned char *output = (unsigned char *)malloc(sizeof(char) * len + 1);
    int i = 0;

    memset(output, 0x00, len + 1);

    for (i = 0; i < len; i++)
    {

        RollKey->RollingKey[RollKey->It % 8] += RollKey->ServerKey[RollKey->It % 128];
        output[i] = input[i] ^ RollKey->RollingKey[RollKey->It % 8];

        RollKey->It = RollKey->It + 1;
    }
    free(input);
    return output;
}
```

Encryption Algorithm

Tofsee has a complex way of communicating with a C&C – it sends various structures to "latch" the connection with the C&C server. To keep this blog post as short and sweet as possible, we will only reference the relevant ones required as an attack vector to crash the binary.

One is operation number 2 (OP2) the **receive resource** command.

Tofsee packets are encapsulated in a header packet defined below:

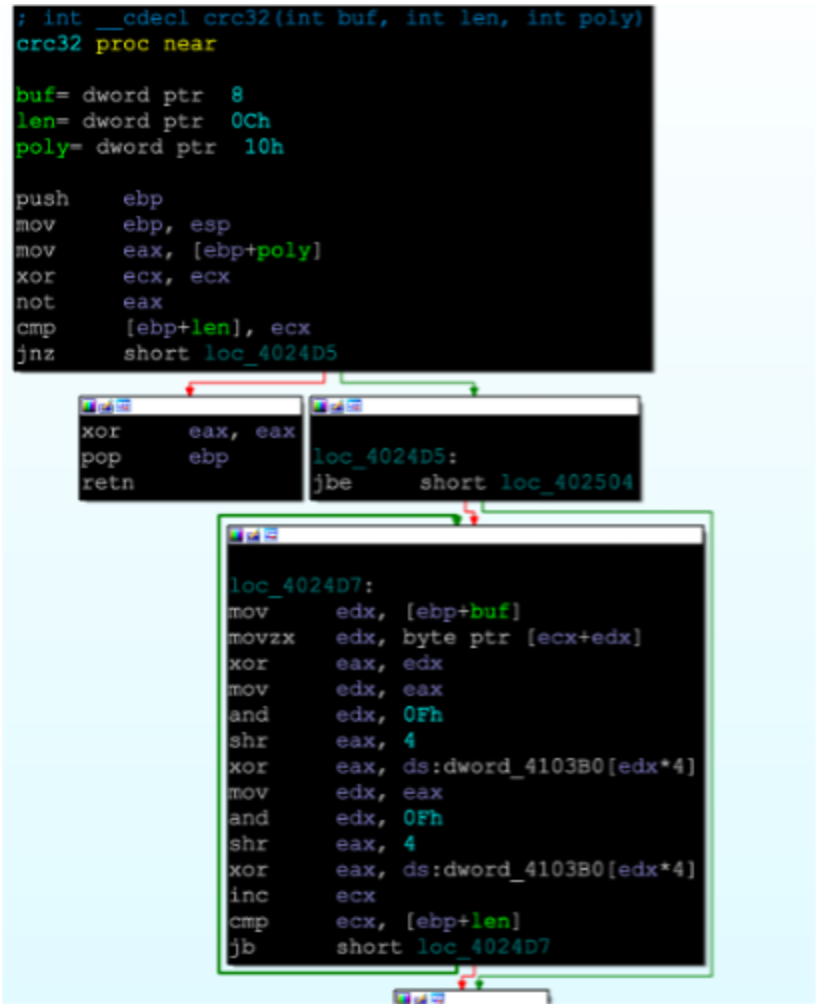
```
struct ResourceStruct
{
    int ResType;
    char name[16];
    int Crc32DataBlob;
    unsigned int SizeOfBlob;
    unsigned int unknownMask;
    unsigned int Const_unknonw; // 3 in the main module
    //: 0x24
    BYTE RESDATA[SizeOfBlob];
};

enum RESTYPE {  1 ==  PEER LIST }
```

Encapsulated packet for OP2

Taking advantage of this vulnerability

We can exploit this lack of a cross-check, i.e., in the code of the CRC32 hash function, where the length of data is not bound-checked, we can craft a packet with a size greater than the buffer, causing an out-of-bounds read error, leading to a crash.

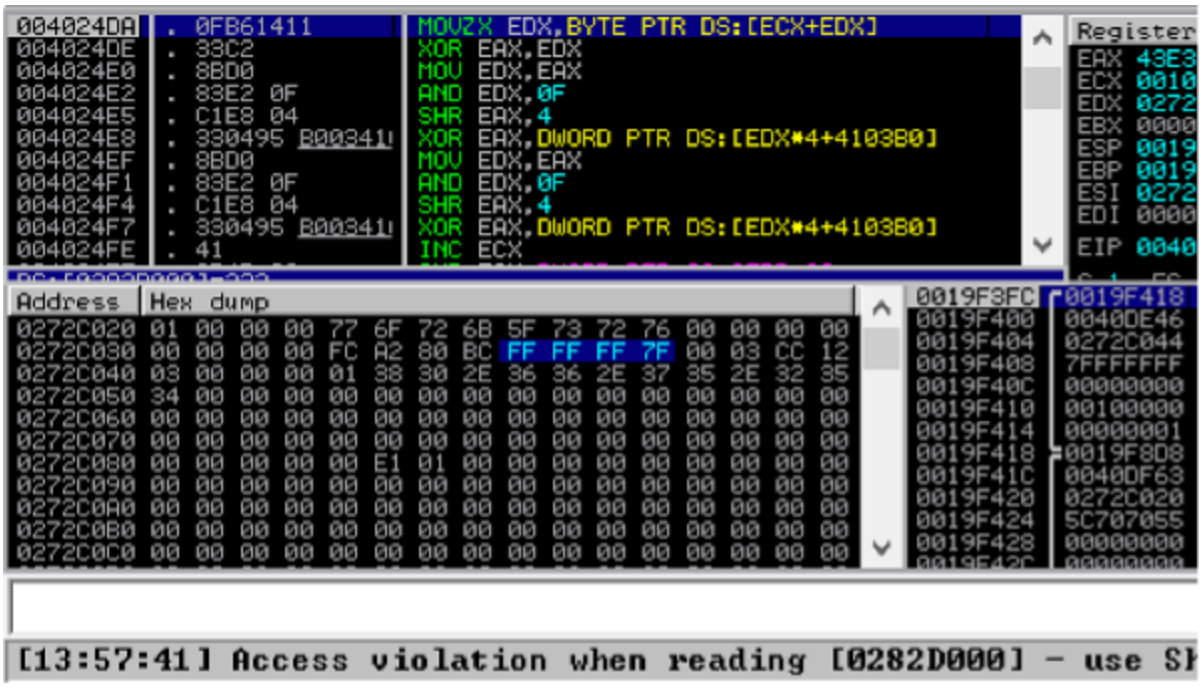
When the CRC32 hash function is called to calculate the hash of the packet's data, it continues reading and processing data from memory beyond the allocated buffer size, potentially crashing Tofsee. This function is present when an **InmemoryConfig Struct** is parsed and populated so that the resource received is stored in the memory.



No length verification checks

For a 4-byte integer, we have the freedom of corrupting the len variable in the range of 0x00-0xFFFFFFFF. This high-range value in the ResourceStructure packet would look something like this (complete with the manipulated len field):

This data is parsed by update_config_resource and eventually fed to the CRC32 hash calculation routine. Due to the manipulated value of len, an out-of-bound read exception is created, ultimately resulting in the binary crashing.



Final words

Both the vaccines discussed in this series and the kill switch are essential tools for protecting computer systems from the ever-evolving threat of the Tofsee malware.

While a malware vaccine can help to prevent infections, and a malware kill switch can help to minimize the damage caused by an ongoing attack, as we've previously discussed, neither tool is foolproof, and you should always use them in conjunction with other security measures.

Happy coding.