Capture of Beacon Frame and Probing Requests for Identification of Frequently Requested Access Points

Abstract:

Network security analysts are tasked with gathering and compiling information pertaining to unknown wireless networks. Many different tools can be utilized to provide this information, which may require manual intervention by the network security analyst to conduct a successful assessment. This manual interaction with the software tools may require prior working knowledge of not only the tools, but the protocols encountered during the network security evaluation. Our research is intended to allow an analyst to conduct their assessment independent of the previous working knowledge of the network under analysis, but of the wireless protocols encountered within the test environment. We intend to utilize existing open source software and build upon this foundation. Our objective is to construct a file representation of the network under test which will includes access point information, clients, and a collection of beacon probe requests broadcast by the clients. This file will be fabricated through the dissection of IEEE 802.1x frames encountered during the examination. This information can then be utilized to create rouge access points for further security assessments.

The following tools were used:

* TP-Link TL-WN821N wireless dongle
* RALINK USB Wifi RT5370 wireless dongle
* Github iSniff source code [1]
* Kali Linux
* VirtualBox VM software
* Python
* Scapy
* Git
* Wireshark

Methodology:

1. Start the Kali Linux virtual machine using VirtualBox
2. Open a new terminal and run the following commands:
   * 1. **ifconfig**

Used to determine if the USB dongle has been detected and mounted. The USB dongle was reported as wlan0 and will be referred to as such for the duration of the document.

* + 1. **ifconfig wlan0 down**

This will bring the wlan0 interface down so that we may set it into monitor mode.

* + 1. **macchanger –r wlan0**

Changes the factory interface MAC address to a random value.

* + 1. **iwconfig wlan0 mode monitor**

Sets the USB dongle into monitor mode.

* + 1. **ifconfig wlan0 up**

This will restore the wlan0 interface, now in monitor mode.

* + 1. **wireshark &**

Wireshark is then used to verify the interface is in monitor mode and capturing non-IP 802.1x traffic

1. Find script online and load script (saved as Tester.py)
   1. **python Tester.py**
      1. verify script is working and capturing traffic.
   2. Analyze script and determine points for code injection
   3. Debug code
   4. Run analysis
   5. During analysis enable wireless devices with known MAC addresses, AP connections
   6. Check file results for accuracy
   7. Repeat insertion and testing in steps 3.2->3.7 until injection complete

Beacon Frame:

Script:

Conclusion:

Works Cited:

[1] https://github.com/0x90/iSniff/blob/master/iSniff.py