Matthew Moltzau

Ryan Vacca

Julia Vrooman

**Final Project : Cyber Security Programming : Group 2**

**Progress Report**

Project 6. Spoofed ARP Detector. The goal of this project is to develop a tool that detects ARP spoofing in LANs and WLANs. You must demo your tool in a testbed LAN.

**Topics**

**- A more in-depth definition of your project, based on your new findings.**

Based on our research into this topic and labs in class, we have found a tool that will greatly benefit our final project. The software tool implemented in Linux is “arpwatch”. This tool monitors address resolution protocol traffic on a computer network and generates a log file of IP-Address and MAC address pairings on the network. The log file lives in the directory “/var/log/syslog” and we can access the a new MAC/IP pairing at the end of the file by using the command < # tail -f /var/log/syslog >. This will be of great use for determining a spoofing. We plan to in our full implementation to access this file and have the new pairing trip our detector function running on node 1, “Kali VM 1”. We have a backup option in which to access the ARP table with the command <arp -a>. Obviously, we will have to parse this table with combinations of unix commands within our python program to extract the change in ARP/IP pairings manually.

**- A progress report, in accordance with the initial guidelines you submitted before.**

*Timeline:*

*SEP: Discuss program architecture1, python version lockdown2.*

*OCT: Configure Virtual network(s)3, research ARP python spoofing4, develop prototype5.*

Up to this point we have remained on schedule that we proposed in the timeline of our project proposal.

1. We have discussed the program architecture and decided on how the program and node layout will be implemented. A bash script will be running on the detector node making calls to “arpwatch” for determining spoofing. We are still discussing the timing for the calls but we plan to check the ARP table as of right now every second. ARP spoofing performed on node 2 will use wireshark to confirm the spoofing was successful by showing the intercepted packet traffic. The actual spoofing will be done in a python program that will make an ARP() request with the victims IP-Address as one of the input parameters. We plan to run attach the detector to the network, then the attacker, and finally the victim to the LAN in that order. This will allow the attacker to as soon as a new IP-Address is discovered on the LAN, have it become the victim. Which in our case will be the Windows Server.
2. We have landed on Python 3 for our programming language version.
3. We have configured the virtual network which consist of two Kali Linux virtual machines and a Windows Server virtual machine.

* Kali VM 1: This virtual machine will serve as node 1 on the LAN, to simulate the router in which we will run our ARP spoofing detector program on.
* Kali VM 2: This virtual machine will serve as node 2 on the LAN, to simulate the attacker in which we will run the program that performs the ARP spoofing.
* Windows Server VM: This virtual machine will serve as node 3 on the LAN, to simulate the victim that will have their IP-Address hi-jacked to the victims MAC address.

1. We have researched the ARP spoofing techniques and have found that we can manually make an ARP() request to hijack the victims IP-Address.
2. Our prototype has not yet connected all the pieces of the project together. So far we can access the ARP table and fin IP/MAC pairs on the LAN. We have also implemented a function to make a comparison against the previos scan and the current scan to decipher changes to the table. The actual spoofing node still needs work to effectively spoof the victims IP-Address. We have made sure on all team members virtual machine configurations have the appropriate software needed to perform the project, IE wireshark, arpwatch, bash shell, python3.

**- Remaining tasks and your detailed plans toward them.**

NOV: Polish prototype into final product, testing, documentation.

DEC: Prepare products for turn-in, practice delivery mechanism for class presentation.

Moving forward we need to get the full protoype implementation working so that we can start testing accuracy and intended output. The rest of November will be to then finish the program