Network Anomaly Detection

Introduction

This project will introduce algorithms to analyze and fingerprint network traffic flows. This is an important skill as during network monitoring or the forensic analysis of an incident that might come across some protocol where the event is not recognized. The project will describe some simple algorithms that aggregate network packets into higher level behavior thus allowing users to fingerprint the characteristic behavior of network protocols to detect anomalies.

Project Objective

In this project, the main objective is to setup a traffic fingerprinting of TLS flows during the initial handshake process. Prior to the encrypted traffic, the server and the client must negotiate the specific levels that each machine has access to including certificates, key exchanges, compression and verification records to ensure that both parties can communicate when packets are encrypted.

Description and Methodology

The project will record the first byte of the SSL record payload in the initial the handshake protocol to reveals which stage of the process is being performed.
Capturing the first byte will allow us to build a dataset for various types of applications and the method of connections used to negotiate the encryption. Thus, allowing us to visualize different method applications use to ensure an encrypted connection.

Resources

We will use Wireshark to monitor the packets being sent in and out of the client. Scapy and specifically pcap library to do packet sniffing. From the Packet Sniffing and Spoofing Lab, Scapy will allow us to write python programs to capture packets and generate a dataset using python tools such as numpy and matplotlib to visualize the data. The use of the Netwox tool developed by SEED labs to implement a simple reset attack.

Learning Experience

This project will allow us to further review the SSL/TLS security protocol. By building the python programs using the specified tool, we can understand the underlying process that applications use to negotiate encryption. Launch an RST attack to show that even encrypted connections are potentially subject to compromise.

Deliverables

Implement a module that monitors the state of each packet belonging to a TLS connection and increments a counter for each transition it observes. These values are then converted into probabilities where we attach the source-destination IP and port combination.
Using the previous module, we will fingerprint different applications. We then visualize the fingerprints obtained and compare them to identify commonalities and differences.
Initiate a TCP session hijacking between the machine attempting to negotiate the TLS connection and the attacking machine to run malicious commands and disconnect the session.

References/Bibliography

This proposal is based on a project from a class on Advanced Networks Security (CS4155)
https://raw.githubusercontent.com/umeer/AdvancedNetworkSecurityProjects/master/Project%205/Project%20description/Project%205%20-%20Description.pdf
Changes were made to the overall selection of the deliverables as the document has an entire section about Markov chains that was omitted. Some additional tasks in that project were also omitted as the minimum viable product would be scope creeped into a much longer and more difficult final project.