Comp 8505 Final Project

Covert Communication Application

"We are what we pretend to be,
So we must be careful what we pretend to be."

- Mother Night
Kurt Vonnegut, Jr.

Objective:
(You may work in groups of two)

- To bring together several stealth software and backdoor concepts covered in class into a single covert communication application.
- To learn how to use such an application in allowing stealthy access to a network or to exfiltrate data from systems within a network.

Your Mission:

Design and implement a complete covert application that will allow a user to open a port (that is otherwise closed) on a firewall, communicate with a "disguised" backdoor application. The backdoor application will accept commands and execute them; the results of the command execution will be sent back to the remote client application.

Background:

- A covert channel is a secret communication pathway, i.e., the communication is concealed. The term originates from the design of highly secure, compartmentalized computer systems, such as those that handle classified information. Such systems are designed to stop one process from communication with another process. This is not an easy objective to achieve in practice simply because any detectable signal that can be influenced by two remote systems has the potential of becoming a channel of communication between them.

- A covert channel need not meet any rigorous standards of software engineering or meet any academic standards of stealth; it merely needs to be unanticipated so that it slips through the network defenses unnoticed. The channel must be robust enough to support exfiltration of data from a system and allow commands to be executed remotely through control messages.

- Covert channels must be custom designed, that is to say, they cannot be known protocols or existing software applications. A covert channel is usually some form of extension upon an existing protocol or application.

- There are two main purposes to using this type of software: to control operate a system remotely (rootkits), and to copy data from a system.

- Examples of remote control of a system include shutting a system down, enabling and disabling certain functions, and manipulating the kernel.

- Copying data from a remote system is referred to as exfiltration or exfil for short. Exfil can take place in many forms such as data transmissions over wireless channels, via data inserted into network protocol headers, etc.

- When carrying our remote penetration of networks, the single most important concern is to avoid detection. The two concepts that are key to avoiding exposure are **minimal footprint** and **unique structure**.

- Minimal footprint means that the tools used for remote penetration must have very little affect on the normal operation of the system.

- Unique structure means that the characteristics of the penetration tool must be unique and obscure. This implies that virus detection applications should not be able to match its operational signature.

Note that there will be a certain amount of flexibility in this project. If you have good ideas on how to implement this application then by all means discuss these with me during the labs.

Exfiltration/Covert Component

- This is the component that will be installed on a machine from which the data is to exfiltrated (**victim** machine).
- The covert application part of project is divided up into two distinct parts as described below.

Part 1

- This portion of the project will deal strictly with the design and implementation of the main backdoor itself.
- This application will accept packets regardless of whether or not any firewall rules are in place once its service port has been opened by a separate application. This means that you will be implementing this part using libpcap.
- The application itself will run as a disguised process; you are required to make it as obscure as possible so as to avoid detection.
- The application will only accept those packets that have been authenticated. The authentication will be in the form of an encrypted password in the packet. This can be embedded in the payload or within one of the protocol header fields.
- Once the packet has been authenticated it will extract a command (also encrypted) from the payload portion and execute the command.
- The results of the command will then be sent back to the **attacker** machine (application) using a **covert channel**.
- Note that the covert channel for sending the command execution results back must be separate from the original channel that was used to connect to the backdoor.
- One way to configure the application parameters is by means of a configuration file.

Part 2

- This portion of the project will implement the file exfiltration and port knocking component for the covert channel to access the **attacker** component and deliver the efiltrated data.
- The application will watch for the creating of a specific file in a specific directory and when that occurs, it will automatically send the file to the **attacker machine** on the other side.
- The file will be delivered covertly using a special sequence of packets or "knocks", which the **attacker** machine will authenticate and provide access to the requested port and application.
- Once the exfiltration is complete your application must close access to the ports again.
- Access to the ports may be time-based or controlled by a separate sequence of packets. In other words, the user can remotely specify how and when to close access to the backdoor application.

Attacker Component

- The **attacker application** must have all of the features to connect to and control the remote system via the backdoor running on a compromised **victim machine**.
- Aside from simple executing remote commands the overall application must provide an exfiltration function. The user will be able to specify that the remote system search (or watch a directory or file for events) for a particular file and send the contents of that file back to the client application covertly.
- The attacker application will accept and decode the knock sequence, provide access to the port and service that will be accepting all the encrypted data that will contain either the results of the command execution, or the exfiltrated file contents from the server.
- I suggest that all the attacker application parameters be selected and set using a configuration file.

Constraints

- Your implementation of this application can be for either the Linux or Windows platforms.
- The user must be able to select from a number of protocols (UDP, TCP, etc) and ports to carry out the penetration and exfiltration.
- You must include a detailed user guide (README will do) as part of your submission.
- Your submission must include a brief **technical report** commenting on your protocol that you designed for your implementation together with ways in which the covert activity could be detected and recommendations on how to stop such activity.

For **bonus marks** you might want to consider advanced features such as those available in other such applications (discussed in class). Analyze all the different programs and see what sort of features they provide.

DUE DATES:

Preliminary Design Work: November 28 – 1000 hrs

Must include all STD's, pseudocode, timelines and task breakdowns and associated deadlines.

In Lab Test: December 12, 2016 – 1030 hrs

You will be required to demonstrate your complete application during the lab.

To Be Submitted: (Due Date: December 12, 2016 – 1000 hrs)

- Submit a zip file containing all the code and documents as described below in the sharein folder for this course under "**Final Project**". Make sure the document is encrypted with your public key.
- Submit a complete, zipped package that includes your report, tools that you used, and any supporting data (dumps, etc), and references. Test results, complete with supporting data such as screen shots and traffic dumps
- Hand in complete and well-documented design work and documents in PDF format.
- Also provide all your code **listings** and an **executable.**

Final Project Evaluation

(1). **Design Work**:	/ 10
(2). **Testing (documentation and data):**	/ 15
(3). **Functionality**:	
Exfiltration/Covert Component	/ 40
Client:	/ 25
(4). **Documentation & Report**:	/ 10
Total:	/ 100