MITRE ENGENUITY | Center for Threat Informed Defense

Lab 4.3: Automating Adversary TTPs

Introduction

Adversaries have been found to perform complicated TTPs in the course of an attack against a target. Emulating these TTPs can prove to be a lengthy and error-prone manual process. TTP automation is a valuable skill in making the execution of such TTPs easily repeatable and free of errors.

In this lab, we will demonstrate the automation of the initial access TTP emulated in Lab 4.2. You will follow provided samples of code to observe how automation of that TTP may be achieved using Python and Bash scripting. Deep coding or scripting knowledge is not required to follow along with this lab.

Objectives:

1. Describe the reason for and value of automation.
2. Create and execute the APT 29 initial access TTP using automated tooling.

Estimated Completion Time:

- 10 minutes

Requirements

1. Kali VM – used as the attack platform to generate the payload and receive the reverse shell.
2. Windows Workstation VM – used as the victim workstation to execute the APT 29 emulated payload.

Malware Warning

Fundamentally, this course entails executing publicly known adversary TTPs so that we can assess and improve cybersecurity. As a result, many of our tools and resources will likely be flagged as malicious by security products. We make every effort to ensure that our adversary emulation content is trusted and safe for the purpose of offensive security testing.

As a precaution, you should not perform these labs on any system that contains sensitive data. Additionally, you should never use capabilities and/or techniques taught in this course without first obtaining explicit written permission from the system/network owner(s).

Formatted: Font: (Default) Arial
Deleted: 4
Formatted: No Spacing
Deleted: A
Formatted: Font: (Default) Arial, 22 pt, Font color: Custom Color(RGB(112,76,159))
Formatted: Heading 1, Font Alignment: Auto
Formatted: Font: (Default) Arial, 22 pt
Deleted: 0…MITRE Engenuity. Approved for public release. Document number MAD013For internal use only
Deleted: For internal use only

MITRE ENGENUITY | Center for Threat Informed Defense

Overview

In Lab 4.2, we created an initial access payload following a TTP that APT_29 has been found to use. It was a very complex process for a single TTP, taking upwards of 30 minutes to walk through and execute. It also involved many steps, some of which required close attention to detail. With this combination of complexity and length, emulating this TTP not only becomes cumbersome, but also prone to error.

This is not an isolated case either. Adversaries continue to develop their tooling and procedures, which become increasingly sophisticated as they work to bypass defenses. Emulating those tools and procedures will also prove to be an increasingly complex and lengthy process.

To decrease both the amount of time taken to reproduce a TTP and the chances of error, we automate the TTP emulation process. Admittedly, developing the automation does take some time. However, we save time and reduce pressure during engagements where we need to reproduce the TTP.

Walkthrough

For this TTP, the heavy lifting has already been done. Several Python and Bash scripts have been written to automate the process performed in Lab 4.2. The Python scripts perform the bulk of the work, such as creation of the base LNK object, preparation of and appending the loader scripts, and compressing the entire payload. The Bash scripts connect the pieces, performing setup, triggering the creation of the payload, and finally performing cleanup.

The custom-built scripts in Lab 4.3 are all either very easy to read or are heavily commented. They should be easy to understand on your own even without deep knowledge of coding. We'll walk through a few of the scripts here to gain familiarity with the overall processes of the automation.

Step 1: Access the Lab Environment

1. Access the range environment by clicking the following link:
   a. TBD

2. Login to the Kali attack platform using the following credentials:
   a. Username: `attacker`
   b. Password: `ATT&CK`

3. Open a terminal and navigate to the lab directory:

```
cd ~/Desktop/AdversaryEmulation/labs/lab_4.3/
```


Commented [DHJ1]: The automation scripts for this are in the lab 4.2 directory. Should this be updated to lab 4.2? Or, are we planning to pull the scripts out into a lab 4.3 directory? Based on answer, we need to update the lab.

Commented [DHJ2R1]: Lab 4.3 in lab_4.4 branch.

Deleted: the

Deleted: 4

Deleted: repo

Commented [ML3]: There should probably be a step number and instruction here after this sentence.

See the next line for an example.

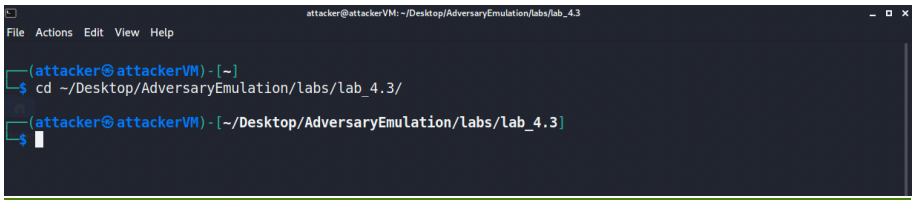
Commented [DHJ4R3]: In lab 4.2 we don't use section numbers in the section headings. I tried to go through and make it consistent with lab 4.2 so it has the same look and feel. If you really want numbers we can put them in, but, it would be different than lab 4.2. Or, we could add them to lab 4.2 too. Just let me know what you want to do.

Commented [DHJ5R3]: Added step numbers.

Formatted: Default Paragraph Font

Deleted: 0…MITRE Engenuity. Approved for public release. Document number MAD013For internal use only

Deleted: For internal use only



4. Download latest lab updates, if any:

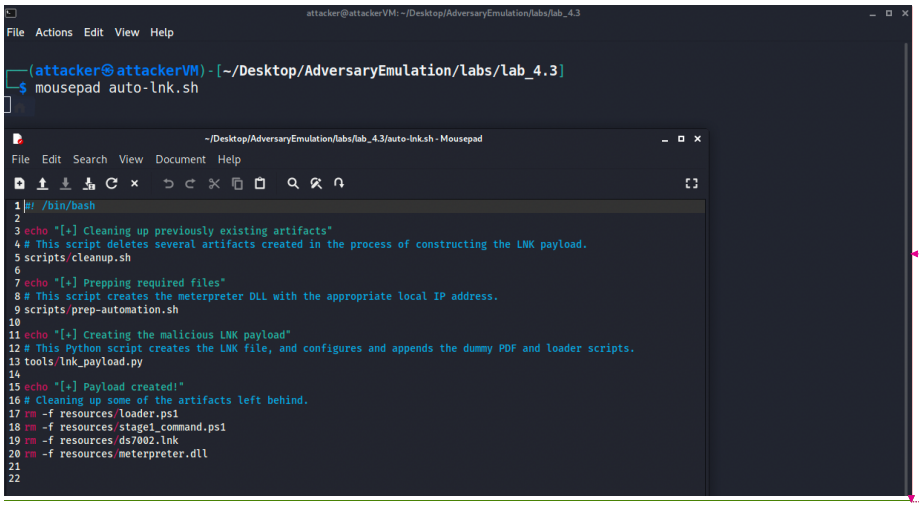
```
git pull
```

5. Lastly, ensure that Windows Security is disabled as it periodically re-enables itself. Please see Troubleshooting at the end of Lab 4.2 for instructions on how to do this.

Step 2: Examine auto_lnk.sh

1. Let's start by taking a look at the overall automation script, `auto-lnk.sh`. Open `auto-lnk.sh`.

```
mousepad auto-lnk.sh
```



The first thing `auto-lnk.sh` does is run `cleanup.sh` (line 5), which cleans up the working directories of any files from previous attempts at building the LNK payload. It then runs `prep-automation.sh` (line 9), which calls `msfvenom` to create a `meterpreter` payload in DLL

Commented [DHJ6]: Add reminder to double check defender and settings are turned off since turns on by itself.

Deleted: ¶

Commented [DHJ7]: The previous lab follows a differ... [1]

Formatted: Heading 2, Font Alignment: Auto

Deleted: Step 1.

Formatted: Font: (Default) Arial

Deleted: Navigate to <folder> and open auto_lnk.sh...etc

Formatted ... [2]

Formatted: Font: (Default) Courier New, 10.5 pt

Deleted: _

Formatted: Font: (Default) Courier New, 10.5 pt

Deleted: Navigate to the ... [3]

Formatted: Font: (Default) Courier New, 10.5 pt

Deleted: _

Commented [DHJ8]: Update screenshot below to ha... [5]

Deleted: AdversaryEmulation/labs/lab_4... [4]

Formatted: Font: (Default) Courier New, 10.5 pt

Formatted: Font: 10.5 pt

Deleted: _

Deleted: ¶

Formatted ... [6]

```
#! /bin/bash

echo "[+] Cleaning up previously exist
# This script deletes several artifact
scripts/cleanup.sh

echo "[+] Prepping required files"
# This script creates the meterpreter
scripts/prep-automation.sh

echo "[+] Creating the malicious LNK p
# This Python script creates the LNK f
tools/lnk_payload.py

echo "[+] Payload created!"
# Cleaning up some of the artifacts le
rm -f resources/loader.ps1
rm -f resources/stage1_command.ps1
rm -f resources/ds7002.lnk
rm -f resources/meterpreter.dll
```

Deleted:

Deleted: #! /bin/bash¶

Deleted: ¶ ... [7]

Formatted: Font: (Default) Courier New, 10.5 pt

Deleted: _

Formatted: Font: (Default) Courier New, 10.5 pt

Formatted: Font: (Default) Courier New, 10.5 pt

Deleted: 0...MITRE Engenuity. Approved for ... [8]

Deleted: For internal use only

format. Next, it runs `lnk_payload.py` (line 13), which is the custom Python script that builds the LNK payload. We'll go into more detail on `lnk_payload.py` in the next section. Lastly, it cleans up artifacts left behind by `lnk_payload.py` along with the `meterpreter` DLL (lines 17-20).

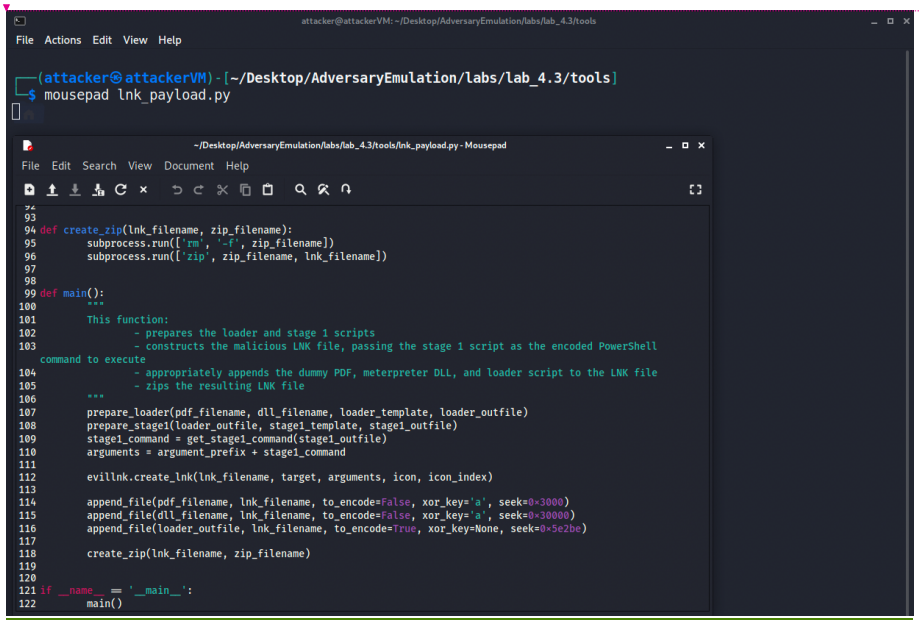
Which directory is lnk_payload.py found in?

a) scripts
b) tools
c) resources

Step 3: Examine lnk_payload.py

1. Here, we'll be taking a deeper look into `lnk_payload.py` to understand the steps this script takes to automate the process of creating the LNK payload. Navigate to the `tools/` directory and open `lnk_payload.py`.

   ```
   mousepad lnk_payload.py
   ```



As this is a longer script, we'll look only at the `main` function to understand the operational flow.

Moved (insertion) [2]
Deleted: ,
Commented [ML10]: Make callouts to the appropriate line number(s) when appropriate

Look at this thing… (line 5)

Deleted: The first thing that `auto_lnk.sh` does is cleans up the working directories of any files from previous attempts at building the LNK payload. It then calls `msfvenom` to create a `meterpreter` payload in DLL format. Next, it runs `lnk_payload.py`, which is the custom Python script that builds the LNK payload. We'll go into more detail on `lnk_payload.py` in the next section. Lastly, it cleans up artifacts left behind by `lnk_payload.py` along with the `meterpreter` DLL.¶
Moved up [2]: Next, it runs `lnk_payload.py`, which is the custom Python script that builds the LNK payload. We'll go into more detail on `lnk_payload.py` in the next section. Lastly, it cleans up artifacts left behind by `lnk_payload.py` along with the `meterpreter` DLL.¶
Deleted: ¶
Commented [ML12]: Highlight the correct answer in green
Deleted: S
Moved (insertion) [1]
Deleted: ¶
Moved up [1]: <#>tools¶
Formatted: Font color: Accent 6
Formatted: Heading 2, Font Alignment: Auto
Deleted: Step 2.
Deleted: Header
Deleted: Step 2 task guidance. What is the student about to do?
Formatted: Font: (Default) Courier New, 10.5 pt
Deleted: AdversaryEmaulation/labs/lab_4.3/
Deleted: ,
Formatted: Indent: Left: 0.5"
Deleted: ¶
Deleted: lnk_payload.py ¶
Deleted: ¶
Deleted: The entire function is presented below. The function is also presented in pieces along with explanations of each section so you can follow along easily.
Deleted: 0…MITRE Engenuity. Approved for public release. Document number MAD013For internal use only
Deleted: For internal use only

Note: If you are digging into the code, many variables are defined in `configs.py`.

First, the `main` function prepares the loader and Stage 1 PowerShell scripts by calling the `prepare_loader()` and `prepare_stage1()` functions (lines 108-109). These functions fill in the appropriate file sizes for the placeholders in each of the script templates and then obfuscates the resulting script using PyFuscation.

Next, `get_stage1_command()` is called (line 110), which reads the contents of the obfuscated Stage 1 script. The contents are then encoded as UTF-16LE, and then encoded into Base64.

The encoded string is inserted into a PowerShell command designed to execute the script (line 110), which is passed to `evillink.create_lnk()` (line 113). That function creates the actual LNK file with the PowerShell command to execute.

Once the LNK file is created, `main` appends the PDF and DLL to it, XOR encrypting both with 'a' (lines 115-116). It then appends the loader PowerShell script with Base64 encoding (line 117).

Finally, `main` zips up the LNK file to create our Zip payload (line 119).

Which function encodes the Stage 1 script?

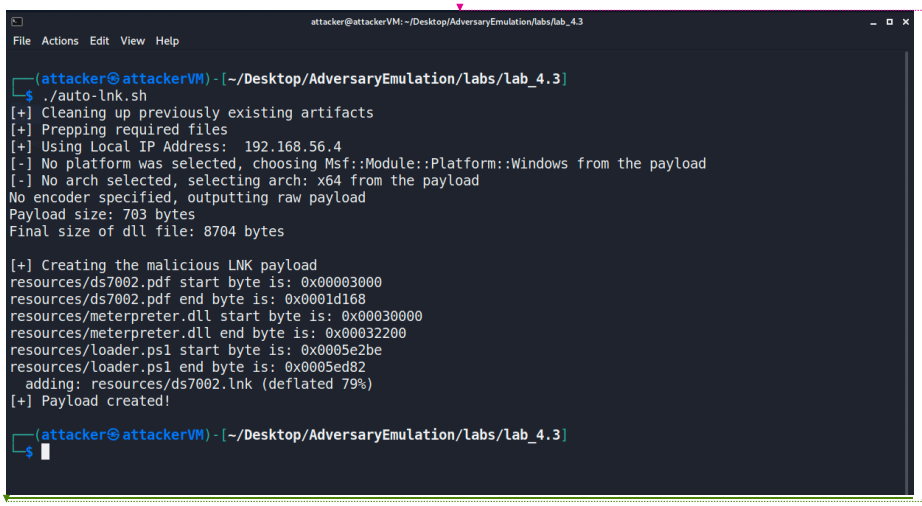
a) `append_file()`
b) `prepare_stage1()`
c) `get_stage1_command()`

Step 4: Execution - Create the Payload Using Automation

Now that we've done the hard work of understanding how these scripts works, we can actually execute them without fear of being a script kiddie and see just how much automation changes everything.

1. To create our initial access payload using the automated tooling, navigate to the `lab_4.3` directory and execute `auto_lnk.sh`.

Deleted: ¶
Deleted: AdversaryEmulation/labs/lab_4. ... [9]
Deleted: def main():¶ ... [10]
Deleted: ¶ ... [11]
Deleted: 9
Deleted: 8
Formatted ... [12]
Deleted: 8
Deleted: 7
Deleted: ¶ ... [13]
Formatted ... [14]
Deleted: 10
Deleted: 09
Deleted: s
Deleted: ¶ ... [15]
Deleted: 1
Deleted: .
Deleted: 3
Deleted: 2
Deleted: ¶ ... [16]
Deleted: 5
Deleted: 4-
Deleted: 6
Deleted: 5
Deleted: 7
Deleted: 6
Deleted: ¶ ... [17]
Deleted: 9
Deleted: 8
Deleted: ¶ ... [18]
Deleted: s
Formatted ... [19]
Deleted: ¶ ... [20]
Formatted ... [21]
Formatted ... [22]
Deleted: 5
Formatted ... [23]
Deleted: ¶ ... [24]
Formatted ... [25]
Deleted: ¶ ... [26]
Deleted: ,
Formatted ... [27]
Deleted: 4
Deleted: i
Deleted: 0…MITRE Engenuity. Approved for ... [28]



With one command, we've created and configured our entire payload, and demonstrated the value of automation!

How easy was it to recreate the LNK payload using the provided automation compared to the manual method?

a) Very easy

Step 5: Deploy the Payload

We can automate part of the process of deploying the payload as well. We've provided a script called `setup_servers.sh` that starts up both the Python3 HTTP server as well as the Metasploit Handler. Execute `setup_servers.sh`.

Deleted: kali@kali:~/AdversaryEmulation/labs/lab_
[+] Cleaning up previously existing arti
[+] Prepping required files
[+] Using Local IP Address:
[-] No platform was selected, choosing M
the payload
[-] No arch selected, selecting arch: x6
Error: One or more options failed to val
[+] Creating the malicious LNK payload
ds7002.pdf start byte is: 0×00003000
ds7002.pdf end byte is: 0×0001d168
meterpreter.dll start byte is: 0×0003000
meterpreter.dll end byte is: 0×00030000
loader.ps1 start byte is: 0×0005e2be
loader.ps1 end byte is: 0×0005ed7e
adding: ds7002.lnk (deflated 79%)
[+] Payload created!
kali@kali:~/AdversaryEmulation/labs/lab_

Deleted: kali@kali:~/AdversaryEmulation/labs/lab
[+] Cleaning up previously existing art
[+] Prepping required files
[+] Using Local IP Address:
[-] No platform was selected, choosing
the payload
[-] No arch selected, selecting arch: x
Error: One or more options failed to va
[+] Creating the malicious LNK payload
resources/ds7002.pdf start byte is: 0×0
resources/ds7002.pdf end byte is: 0×000
resources/meterpreter.dll start byte is
resources/meterpreter.dll end byte is:
resources/loader.ps1 start byte is: 0×0
resources/loader.ps1 end byte is: 0×000
adding: resources/ds7002.lnk (deflate
[+] Payload created!
kali@kali:~/AdversaryEmulation/labs/lab

Deleted: ¶
¶

Deleted: ¶

Formatted: Font color: Accent 6

Deleted: 6

Deleted: Run

Formatted: Font: (Default) Courier New, 10.5 pt

Formatted: Font: (Default) Courier New, 10.5 pt

Deleted: 0…MITRE Engenuity. Approved for public release. Document number MAD013For internal use only

Deleted: For internal use only

Step 6: Execute the Payload

While the actual execution of the payload can be automated as well in a test scenario, the focus of this lab is only on the automation of the preparation of TTPs. As such, execution of the payload will be almost? identical to the execution step (Step 12) in Lab 4.2.

The only difference is:

- When you open `ds7002.zip`, `ds7002.lnk` will be found in the `resources/` directory.

Step 7: Shutdown Adversary Infrastructure

Formatted: Centered

Formatted: Font: (Default) Arial

Formatted: Heading 2, Font Alignment: Auto

Formatted: Highlight

Formatted: No bullets or numbering

Deleted: ¶
To run the payload, follow the instructions beginning with Step 10 in Lab 4.2 with two minor modifications.¶
¶
When running Step 11, make sure to start the web server from the `lab_4.3` directory.

Formatted: Indent: Left: 0.5", No bullets or numbering

Commented [DHJ13]: Do we want the script generating the LNK file in a resources/ sub-directory?

Deleted: When running Step 12, w

Formatted: Bulleted + Level: 1 + Aligned at: 0.25" + Indent at: 0.5"

Formatted: Font: (Default) Courier New, 10.5 pt

Formatted: Font: (Default) Courier New, 10.5 pt

Formatted: Font: (Default) Courier New, 10.5 pt

Formatted: Font: (Default) Arial, 18 pt, Font color: Custom Color(RGB(127,127,127))

Formatted: Heading 2, Font Alignment: Auto

Deleted: 0…MITRE Engenuity. Approved for public release. Document number MAD013For internal use only

Deleted: For internal use only

The last piece of this lab is to close the servers that were started up to deploy our payload. To do this, open up a new terminal on the `AttackerVM`, navigate to the `AdversaryEmulation/labs/lab_4.3/scripts` directory, and run `shutdown_scripts.sh` as `sudo`.

```
┌──(attacker㉿attackerVM)-[~]
└─$ cd AdversaryEmulation/labs/lab_4.3/scripts/

┌──(attacker㉿attackerVM)-[~/AdversaryEmulation/labs/lab_4.3/scripts]
└─$ sudo ./shutdown_servers.sh
[sudo] password for attacker:
[+] Stopped Python3 HTTP server.
[+] Stopped msfconsole.

┌──(attacker㉿attackerVM)-[~/AdversaryEmulation/labs/lab_4.3/scripts]
└─$
```

With that, we've concluded lab 4.3!

Summary

In this lab, we used automated tooling to create the initial access payload developed in Lab 4.2. In using these scripts, we saw the value that automation can provide in repeating complex and lengthy procedures.

Formatted: Font: (Default) Calibri, 12 pt, Font color: Auto
Formatted: Font color: Auto
Formatted: Font: (Default) Calibri, 12 pt, Font color: Auto
Formatted: Font color: Auto
Formatted: Font: (Default) Calibri, 12 pt, Font color: Auto
Formatted: Font color: Auto
Formatted: Font: (Default) Calibri, 12 pt, Font color: Auto
Formatted: Font color: Auto
Formatted: Font: (Default) Calibri, 12 pt, Font color: Auto
Formatted: Centered
Formatted: Font: (Default) Calibri, 12 pt, Font color: Auto
Formatted: Default Paragraph Font, Font: (Default) Arial, 22 pt, Font color: Custom Color(RGB(112,76,159))
Formatted: Heading 1, Font Alignment: Auto
Formatted: Font: (Default) Calibri, 12 pt, Font color: Auto
Formatted: No bullets or numbering
Formatted: Font: (Default) Calibri, 12 pt
Deleted: 0…MITRE Engenuity. Approved for public release. Document number MAD013For internal use only
Deleted: For internal use only

Page 3: [1] Commented [DHJ7] **Dan Haynes** **9/13/21 11:32:00 AM**

The previous lab follows a different look in feel. Specifically, this would be a sub-heading of walkthrough and the first sentence would have a step number. The follow changes are bringing the two labs in alignment.

Page 3: [2] Formatted **Dan Haynes** **9/13/21 11:34:00 AM**

Numbered + Level: 1 + Numbering Style: 1, 2, 3, … + Start at: 1 + Alignment: Left + Aligned at: 0.25" + Indent at: 0.5"

Page 3: [3] Deleted **Dan Haynes** **9/20/21 5:17:00 PM**

1.

Page 3: [4] Deleted **Dan Haynes** **9/13/21 11:40:00 AM**

Page 3: [5] Commented [DHJ8] **Dan Haynes** **9/13/21 11:40:00 AM**

Update screenshot below to have same look-and-feel.

Page 3: [6] Formatted **Arunagiri Govardhen** **9/7/21 10:18:00 AM**

Border: Top: (No border), Bottom: (No border), Left: (No border), Right: (No border)

Page 3: [7] Deleted **Arunagiri Govardhen** **9/7/21 10:16:00 AM**

Page 1: [8] Deleted **Dan Haynes** **10/5/21 1:58:00 PM**

Page 5: [9] Deleted **Dan Haynes** **9/13/21 12:37:00 PM**

Page 5: [10] Deleted **Arunagiri Govardhen** **9/7/21 10:25:00 AM**

Page 5: [11] Deleted **Dan Haynes** **9/13/21 12:37:00 PM**

Page 5: [11] Deleted **Dan Haynes** **9/13/21 12:37:00 PM**

Page 5: [12] Formatted **Dan Haynes** **9/20/21 5:23:00 PM**

Font: 10.5 pt

Page 5: [13] Deleted **Arunagiri Govardhen** **9/7/21 10:27:00 AM**

Page 5: [14] Formatted **Dan Haynes** **9/20/21 5:11:00 PM**

Border: Top: (No border), Bottom: (No border), Left: (No border), Right: (No border)

Page 5: [15] Deleted **Arunagiri Govardhen** **9/7/21 10:27:00 AM**

Page 5: [16] Deleted **Arunagiri Govardhen** **9/7/21 10:28:00 AM**

Page 5: [17] Deleted **Arunagiri Govardhen** **9/7/21 10:28:00 AM**

Page 5: [18] Deleted **Arunagiri Govardhen** **9/7/21 10:28:00 AM**

Page 5: [19] Formatted **Arunagiri Govardhen** **8/31/21 11:16:00 AM**

Font color: Accent 6

Page 5: [20] Deleted **Govardhen Arunagiri** **2/8/22 10:51:00 AM**

Page 5: [21] Formatted **Dan Haynes** **10/5/21 12:24:00 PM**

Font: (Default) Arial

Page 5: [22] Formatted **Dan Haynes** **10/5/21 12:24:00 PM**

Heading 2, Font Alignment: Auto

Page 5: [23] Formatted **Dan Haynes** **10/5/21 12:24:00 PM**

Font: (Default) Arial

Page 5: [24] Deleted **Dan Haynes** **9/13/21 12:53:00 PM**

Page 5: [25] Formatted **Dan Haynes** **10/5/21 12:24:00 PM**

Font: (Default) Arial, 18 pt, Font color: Custom Color(RGB(127,127,127))

Page 5: [26] Deleted **Dan Haynes** **9/13/21 12:53:00 PM**

Page 5: [27] Formatted **Dan Haynes** **9/14/21 8:54:00 AM**

Numbered + Level: 1 + Numbering Style: 1, 2, 3, … + Start at: 1 + Alignment: Left + Aligned at: 0.25" + Indent at: 0.5"

Page 1: [28] Deleted **Dan Haynes** **10/5/21 1:58:00 PM**

Page 1: [28] Deleted **Dan Haynes** **10/5/21 1:58:00 PM**