

Lab 04: Generating a list of target users

Table of Contents

Lab 04: Generating a list of target users	1
Goals	1
Requirements.....	1
Introduction	1
1. Identify the organization's email address format	1
2. Collect a list of employee names	3
3. Turn the employee names you collected into email addresses	13
4. Generating email addresses from common first and last names	14

Goals

- Generate a list of user IDs in use by the target organization that can be targeted in later attacks.

Requirements

- Kali Linux VM with Internet access.
- Recon-ng database created in previous exercises.

Introduction

During this exercise, you'll generate a list of user IDs to be targeted during later attacks. Since later lab exercises target Office 365, and Office 365 uses email addresses for user IDs, this exercise will focus on generating email addresses. Similar methods can also be used for generating usernames instead of email addresses when you're targeting other services.

1. Identify the organization's email address format

1. Run the command below in Recon-ng to display unique email addresses collected during the previous exercises.

```
db query select distinct email from contacts order by email asc
```

```
[recon-ng][default][whois_pocs] > db query select distinct email from contacts order by email asc
```

email
Bouunk@Microsoft.com
DeanIt@microsoft.com
GEOFFK@microsoft.com
JOCORTEZ@microsoft.com
Lloyd.kim@microsoft.com
V-JMASON@MICROSOFT.COM
a_rayf@microsoft.com
abdessemed@microsoft.net
abuse@microsoft.com
andrewbr@microsoft.com

Unique Email Addresses Displayed

- Examine the email addresses for patterns in the user naming scheme. Patterns may be simple or complex. You may also observe more than one pattern for the same domain or organization. Some examples of patterns you might observe are listed below:
 - {First Name}@{Domain}
 - {First Name}{Last Name}@{Domain}
 - {First Name}.{Last Name}@{Domain}
 - {First Initial}{Last Name}@{Domain}
 - {First 3 letters of Last Name}{First 8 letters of First Name}@{Domain}
- If the pattern isn't clear through manual observation, you can also search for the email addressing format on websites like Hunter.io.

<https://www.hunter.io>

contoso.com Find email addresses

Most common pattern: **{first}{l}@contoso.com** 2,112 email addresses

a ounting@contoso.com ✓ 20+ sources ▾

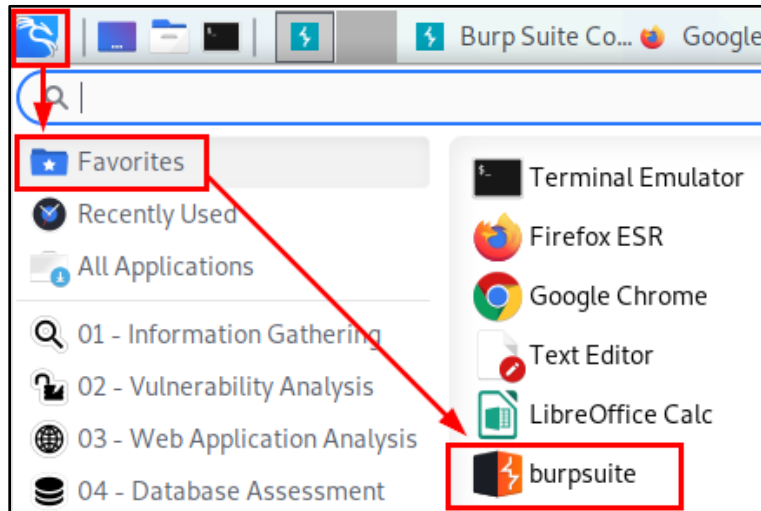
v ay@contoso.com ● 1 source ▾

Contoso.com Email Format Identified by Hunter.io

2. Collect a list of employee names

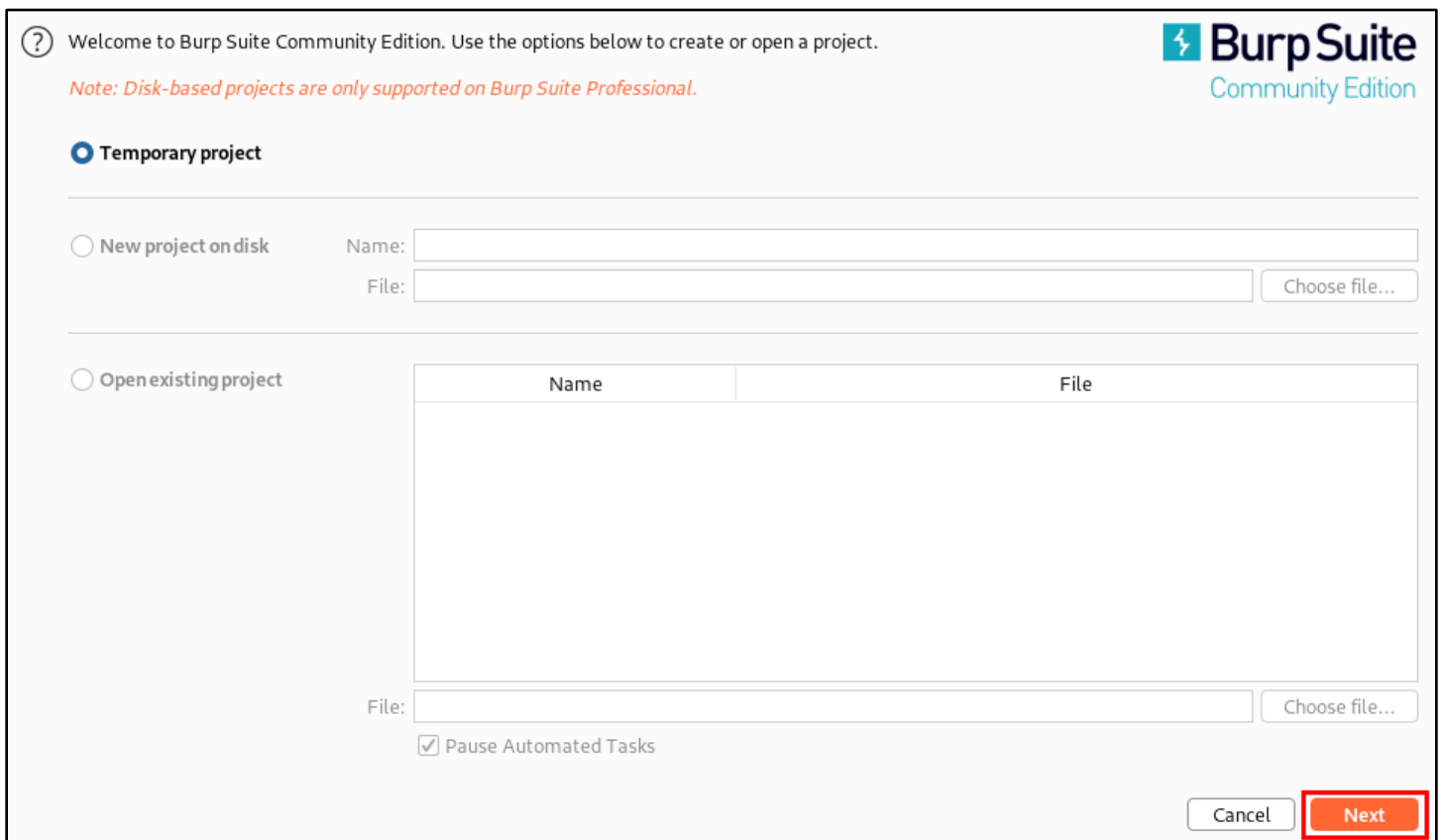
When email addressing formats are based on employees' names, additional email addresses can be guessed by collecting employee names from sites like LinkedIn and then generating an appropriate email address based on the schema. For example, if a company uses the format {First Initial}{Last Name}@companydomain.com, the email address of an employee named John Smith would most likely be jsmith@companydomain.com.

1. To collect employee names from the web, open Burp Suite from the Kali Linux applications menu.



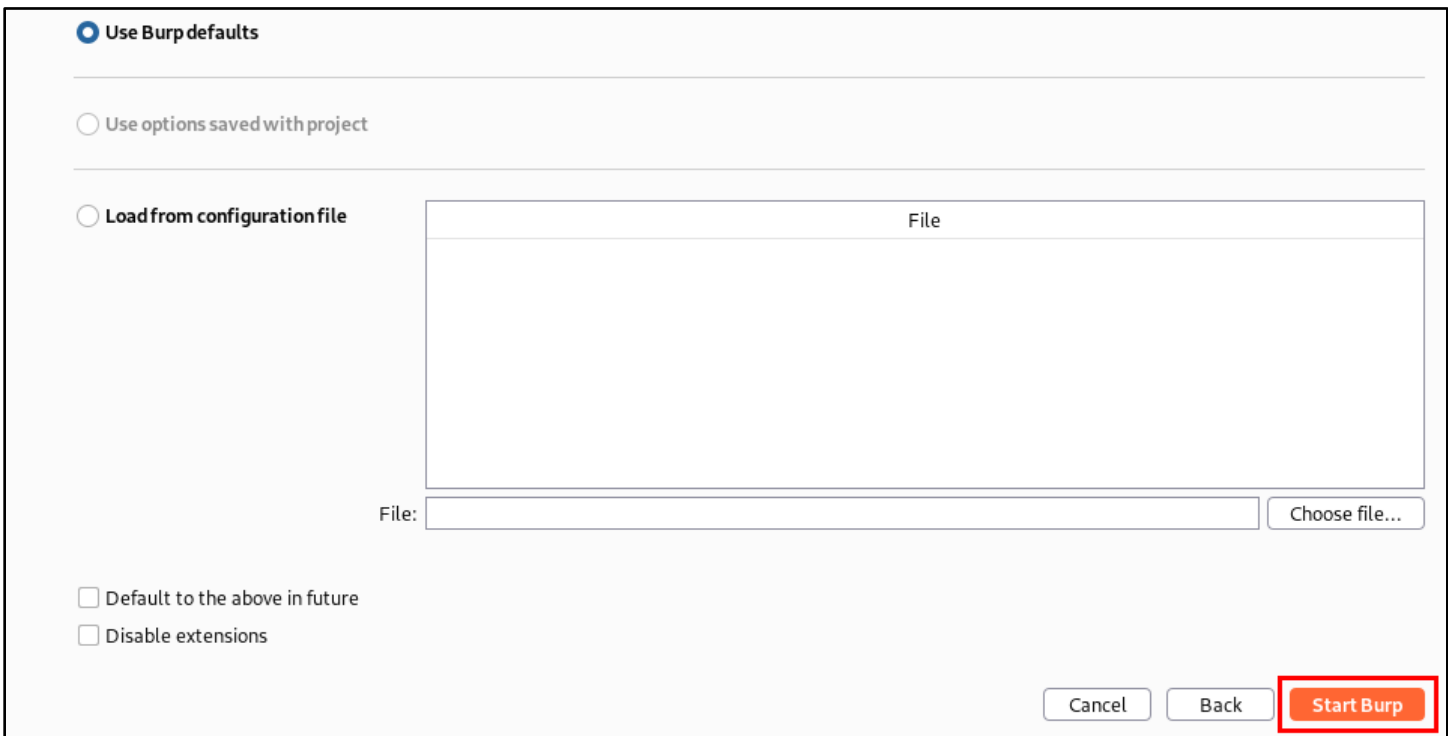
Burp Suite Location in the Applications Menu

2. Click the "Next" button after starting Burp Suite to create a temporary project in which to complete this exercise.



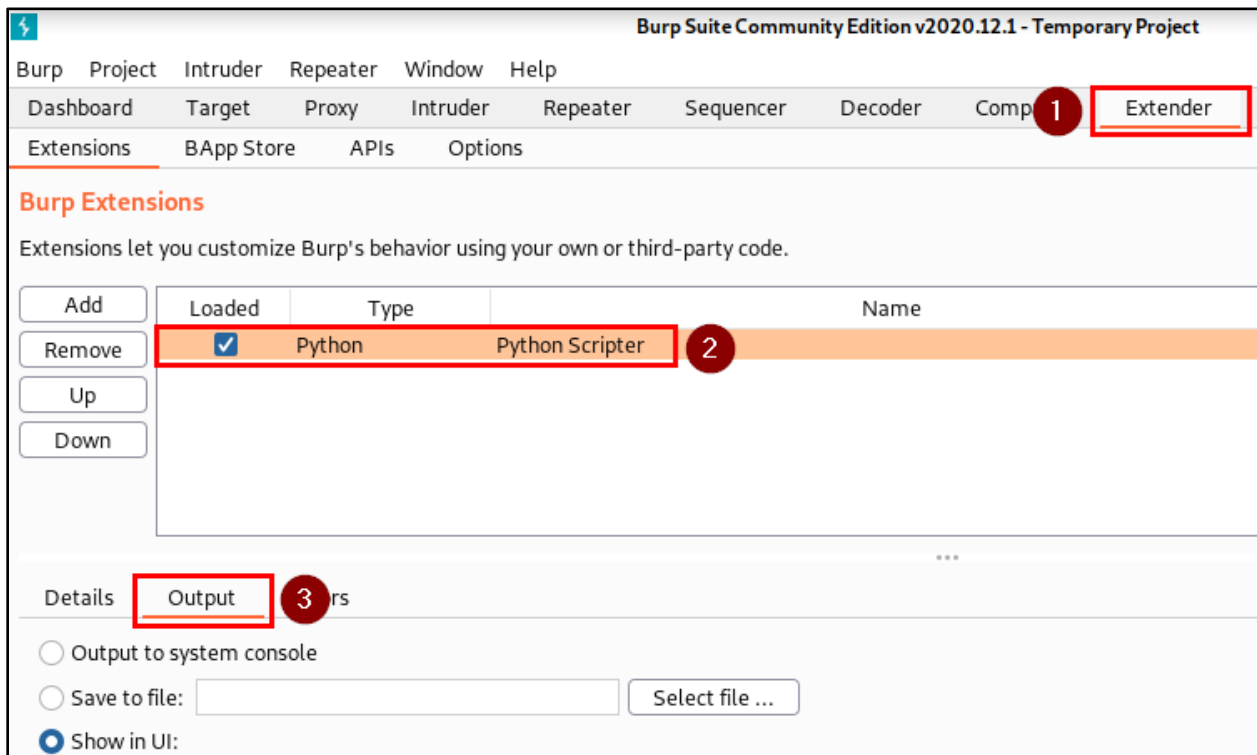
Clicking the Next Button in Burp Suite

3. In the next window that appears, click "Start Burp" to use the pre-configured settings.



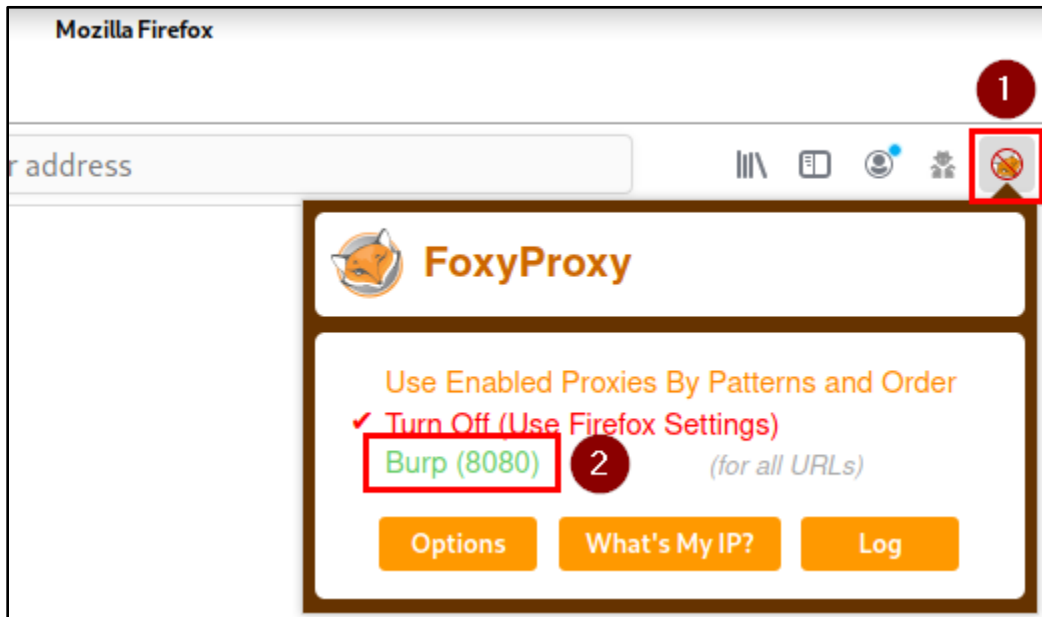
Location of the "Start Burp" Button

4. The Python Scripter extension has already been installed in Burp Suite, and a custom Python script has been loaded to scrape names from websites used in this lab. To see the names collected by Burp while they are scraped from the web, click on the "Extender" tab. Then make sure "Python Scripter" is loaded and selected under Burp Extensions, and click on the "Output" tab below.



Displaying Python Scripter Output in Burp Suite

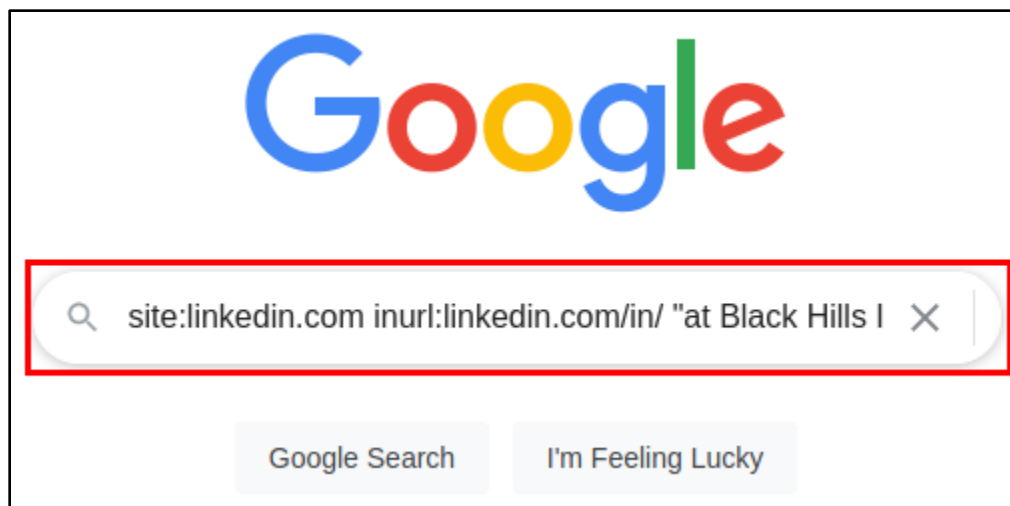
- In the Firefox web browser, click on the FoxyProxy icon in the toolbar, and then click on "Burp (8080)" in the FoxyProxy menu to begin proxying traffic through Burp Suite.



Enable Burp Suite in FoxyProxy

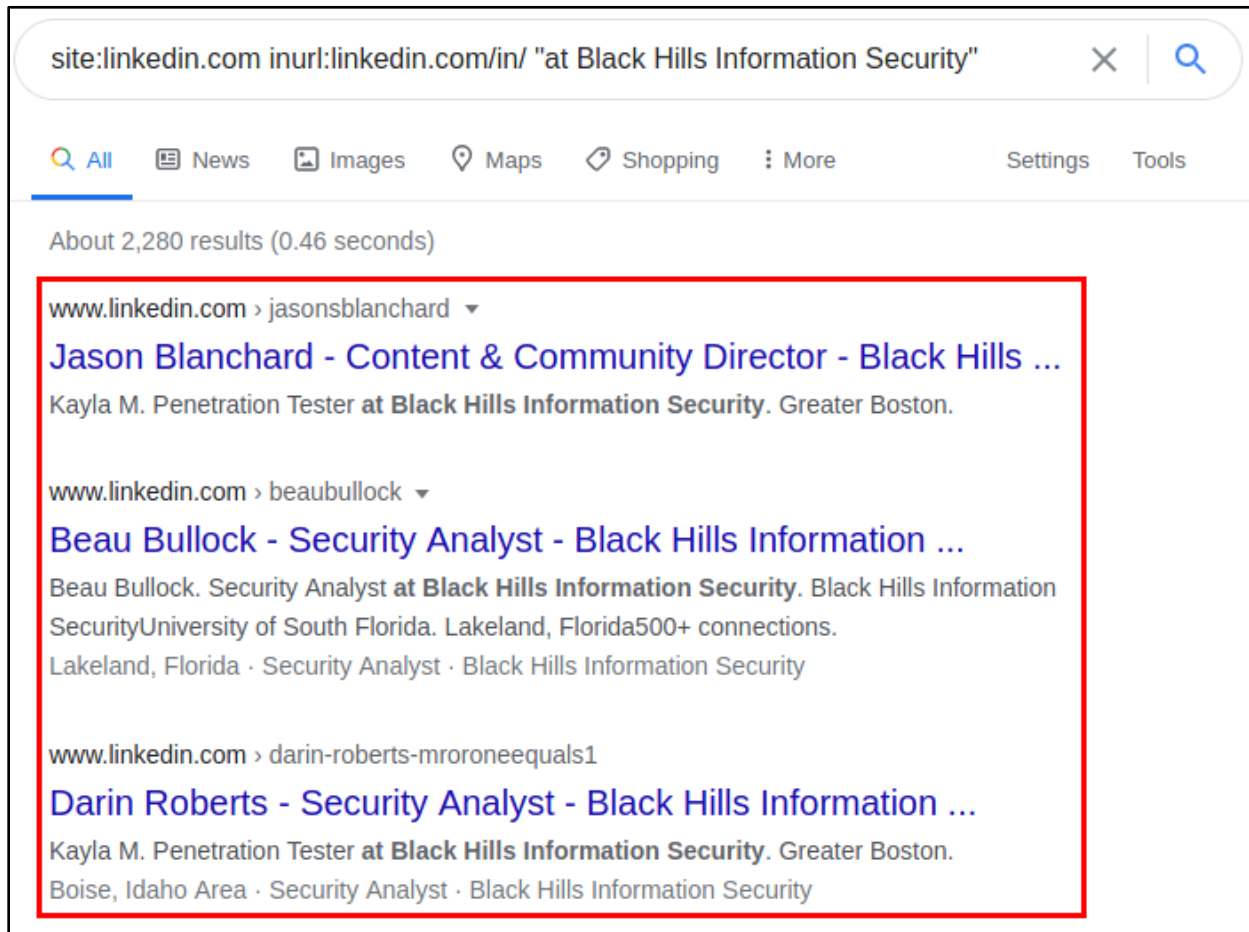
- Next, brows to www.google.com and enter the following web search. You can replace "Black Hills Information Security" with the name of your target organization.

`site:linkedin.com inurl:linkedin.com/in/ "at Black Hills Information Security"`



Targeted Search Query Submitted to Google

7. The search results shown by Google should include names and titles of employees at your target organization, similar to what is shown below.



LinkedIn User Profiles Shown in Google Search Results

8. If you look back in the Burp Extender window now, you should also see the Output tab populated with the names and titles from the page.

Burp Extensions

Extensions let you customize Burp's behavior using your own or third-party code.

	Loaded	Type	Name
<input checked="" type="checkbox"/>	Python	Python Scripter	

Details Output Errors

Output to system console

Save to file:

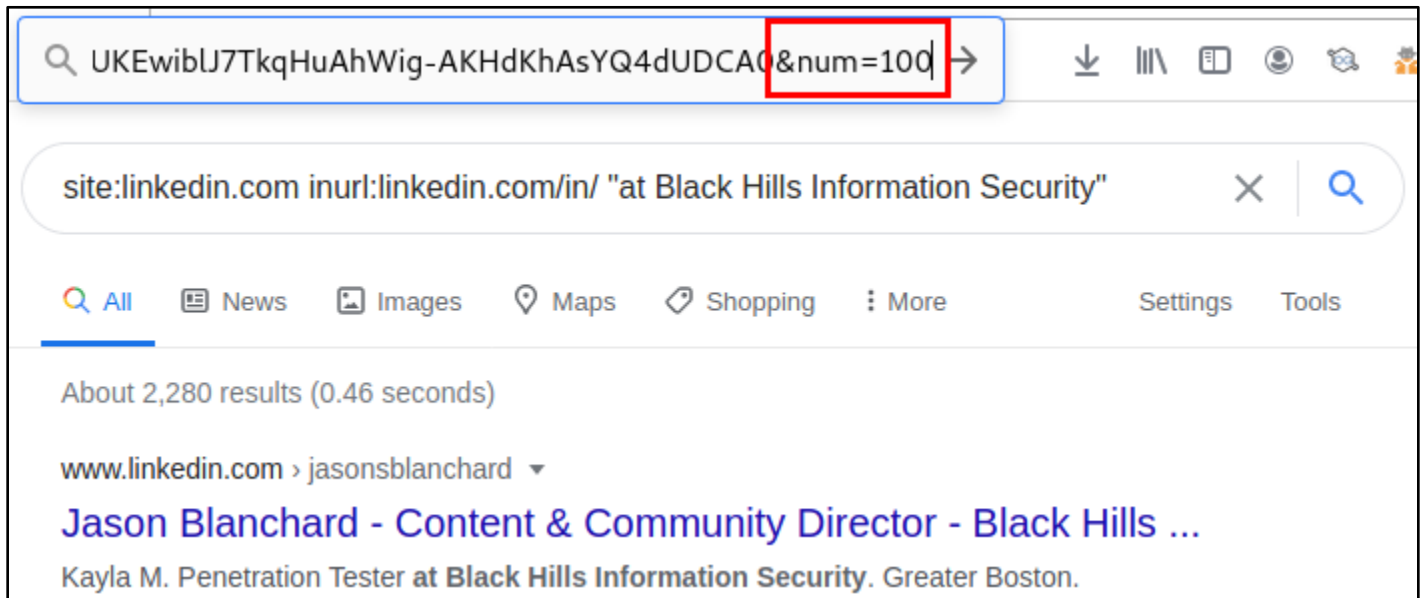
Show in UI:

```
1 "Jason Blanchard","Content Community Director","Black Hills ..."  
2 "Beau Bullock","Security Analyst","Black Hills Information ..."  
3 "Darin Roberts","Security Analyst","Black Hills Information ..."  
4 "Dakota Nelson","Security Consultant","Leviathan Security ..."  
5 "Craig Vincent","Security Analyst","Black Hills Information ..."  
6 "John Strand","Aurora, Colorado | Professional Profile | LinkedIn",""  
7 "Christopher (CJ) Cox","Chief Operating Officer","Black Hills ..."  
8 "Brian King","Security Analyst","Black Hills Information Security ..."  
9 "Bryan Strand","Sales Director/Consultant","Black Hills ..."  
10 "Samuel Carroll","Security Analyst","Black Hills Information ..."  
11
```

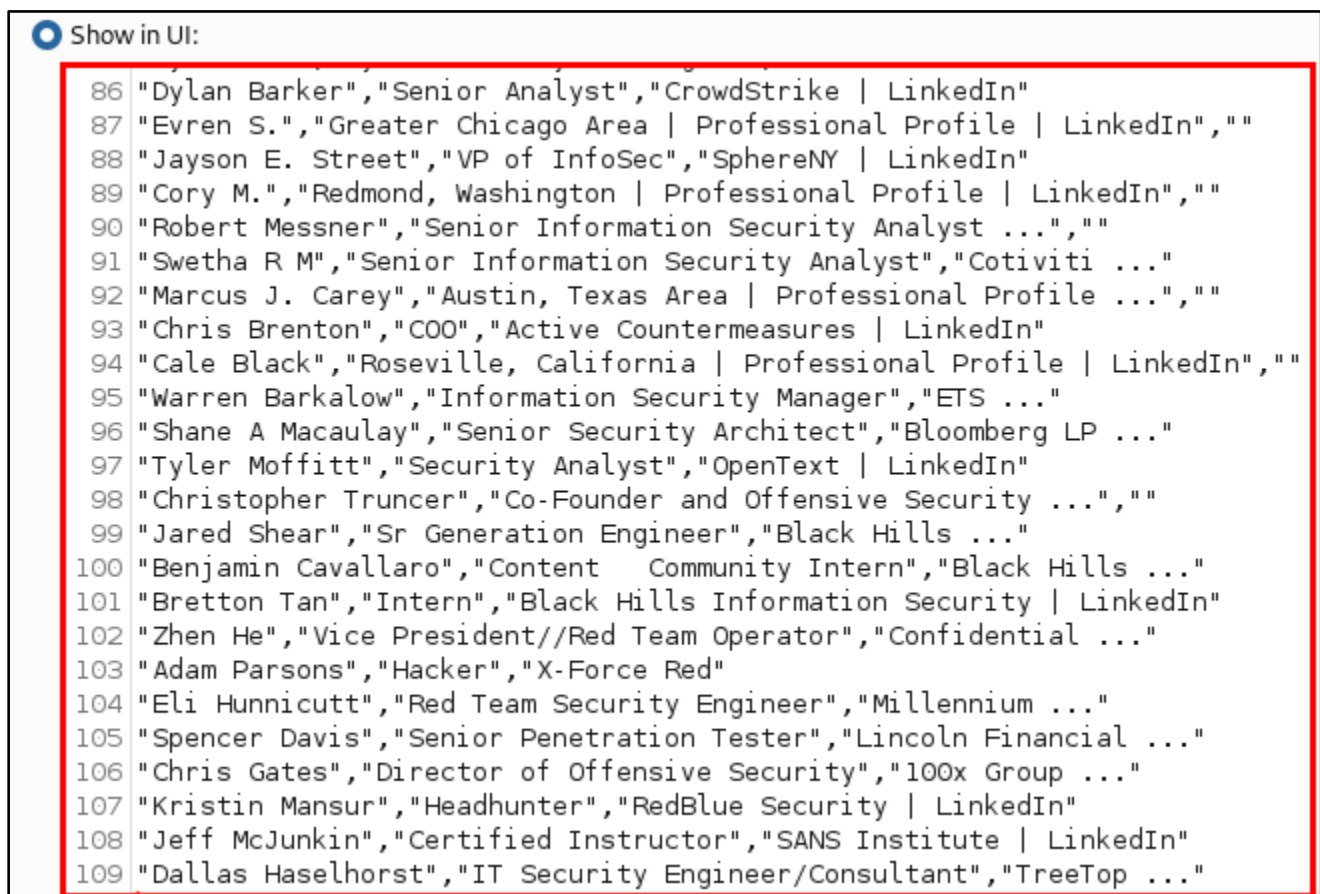
Contact Data Scraped from Search Results by Burp Suite

9. As you continue clicking through each page of Google search results, additional entries will be added to the list. You can speed up this process by clicking in your browser's address bar, pressing the "End" key on your keyboard (to move to the end of the URL), and adding "&num=100" to the end of the URL. Then press Enter to

load the modified URL. This will cause Google to display 100 results per page - greatly reducing the number of pages you need to click through.

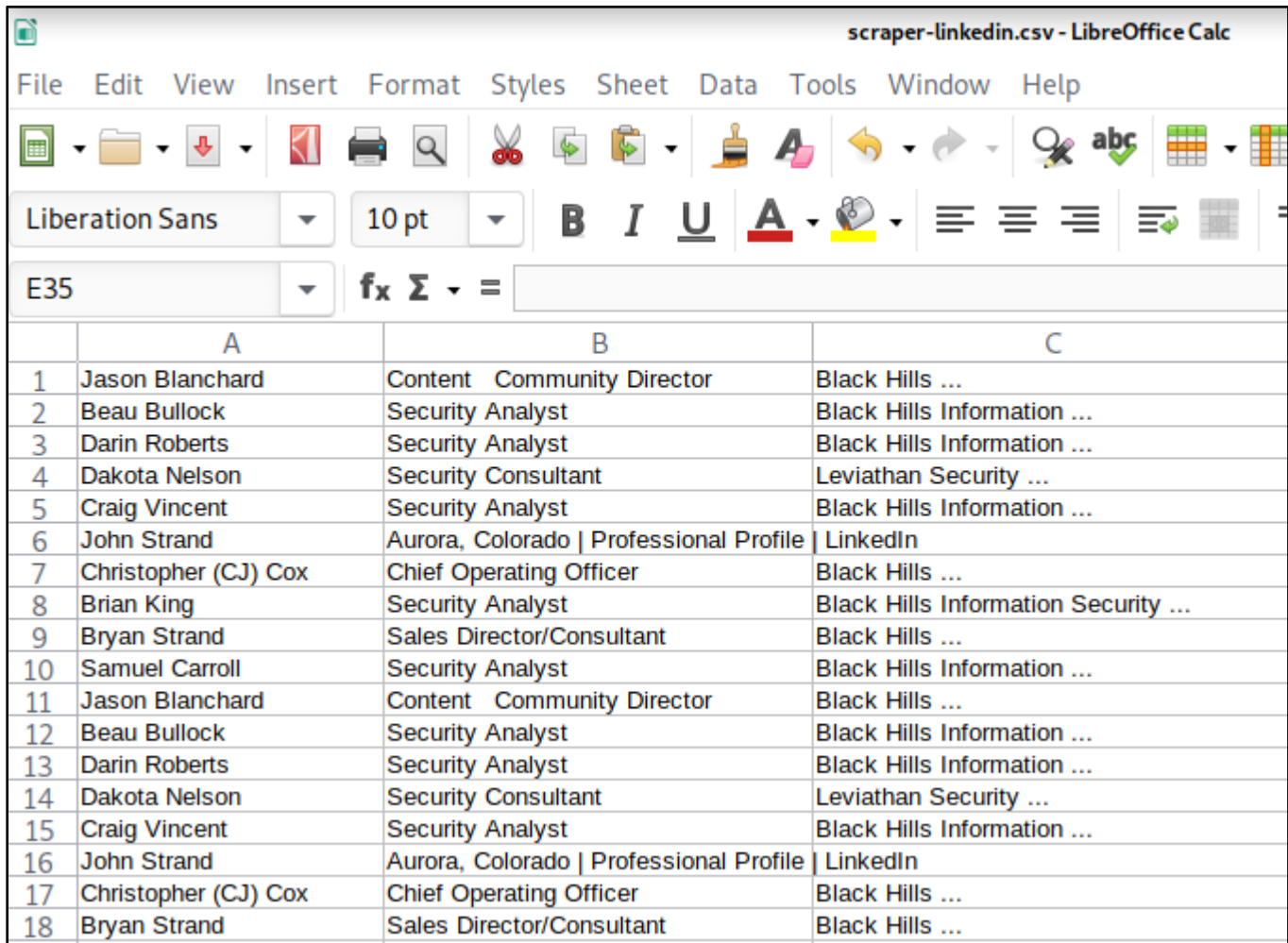


URL Modification to Display 100 Results Per Page



Additional Contacts Collected in Burp

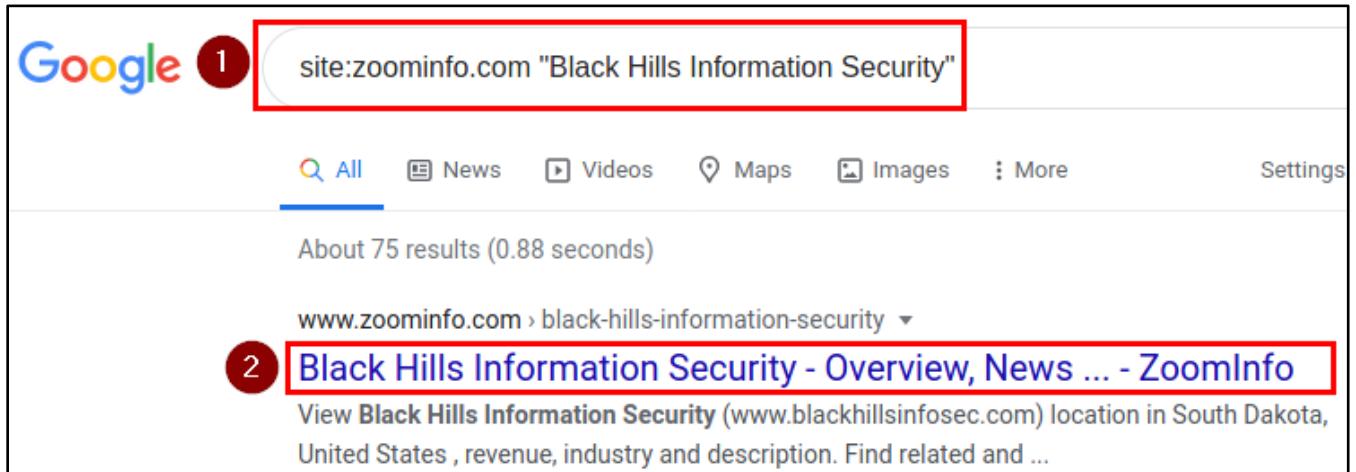
10. The output collected by Burp during is also saved in your Kali user's home directory, in a file named "scraperscraper-linkedin.csv". The CSV file can be opened in LibreOffice Calc as shown in the screenshot below.



"scraperscraper-linkedin.csv" Opened in LibreOffice Calc

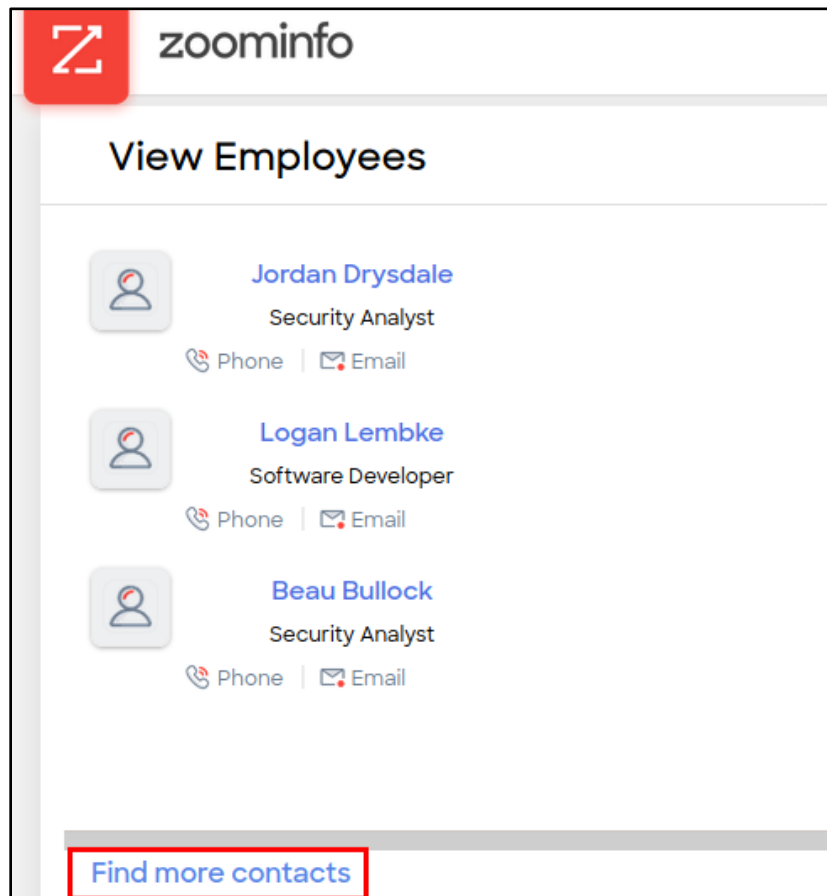
11. The Python script loaded in Burp also supports scraping employee names from ZoomInfo.com. To scrape ZoomInfo, first perform the following search on Google, again replacing "Black Hills Information Security" with your target company. Then click on the company overview in the search results.

site:zoominfo.com "Black Hills Information Security"












Using Google to Search for a Company on the ZoomInfo Website

12. On the ZoomInfo website, scroll down to the "View Employees" section, and click on "Find more contacts"



Clicking "Find more contacts" on ZoomInfo

13. The page that loads will display names and locations of the company employees.

Index of contact profiles from Black Hills Information Security				
1-25 of 45 results				
Contact Name	Contact Info	Job Title	Location	Last Update
 Jason Blanchard	 Email  Direct	Director, Content & Community	United States' South Dakota' Spearfish	12/25/2020
 Lisa Woody	 Email  Direct	Software Engineer	United States' South Dakota' Spearfish	12/25/2020
 Laura Seumanutafa	 Email  Direct	Director, Logistics	United States' South Dakota' Spearfish	12/25/2020

Names and Locations of Company Employees

14. Look back in your Burp Extender window again to see that this information has automatically been logged by Burp.

Details **Output** Errors

Output to system console

Save to file:

Show in UI:

```
3 "Jason Blanchard","Director, Content & Community","United States, South Dakota, Spearfish","12/25/2020",""
4 "Lisa Woody","Software Engineer","United States, South Dakota, Spearfish","12/25/2020",""
5 "Laura Seumanutafa","Director, Logistics","United States, South Dakota, Spearfish","12/25/2020",""
6 "Kevin Klingbile","Security Analyst","United States, South Dakota, Spearfish","12/25/2020",""
7 "Ethan Robish","Security Consultant","United States, South Dakota, Spearfish","12/25/2020",""
8 "Ralph May","Security Analyst","United States, South Dakota, Spearfish","12/25/2020",""
9 "John Strand","Owner and Security Analyst","United States, South Dakota, Spearfish","12/25/2020",""
10 "Derrick Rauch","Contractor","United States, South Dakota, Spearfish","12/25/2020",""
11 "James Lee","Hacker","United States, South Dakota, Spearfish","12/25/2020",""
12 "Cj Cox","Chief Operating Officer","United States, South Dakota, Spearfish","12/25/2020",""
13 "Melisa Wachs","Director","United States, South Dakota, Spearfish","12/25/2020",""
14 "Melissa Bruno","Security Analyst","United States, South Dakota, Spearfish","12/25/2020",""
15 "Jennipher Creed","Position In Accounting","United States, South Dakota, Spearfish","12/25/2020",""
16 "Bryan Strand","Manager, Business Capture & A Consultant","United States, South Dakota, Spearfish","12/25/2020",""
17 "Beau Bullock","Security Analyst","United States, South Dakota, Spearfish","12/25/2020",""
18 "Heather Doerges","Project Scheduler & Manager","United States, South Dakota, Spearfish","12/25/2020",""
19 "Marcello Salvati","Security Analyst","United States, South Dakota, Spearfish","12/25/2020",""
20 "Deb Wigley","Manager, Content & Community","United States, South Dakota, Spearfish","12/25/2020",""
21 "Brian King","Security Analyst","United States, South Dakota, Spearfish","12/25/2020",""
22 "Luke Baggett","Security Analyst","United States, South Dakota, Spearfish","12/25/2020",""
23 "Craig Vincent","Security Analyst","United States, South Dakota, Spearfish","12/25/2020",""
24 "Kent Ickler","Security Analyst & Systems Administrator","United States, South Dakota, Spearfish","12/25/2020",""
25 "Kelsey Bellew","Security Analyst","United States, South Dakota, Spearfish","12/25/2020",""
26 "Michael Allen","Penetration Tester and Security Analyst","United States, South Dakota, Spearfish","12/25/2020",""
27 "Justin Angel","Security Analyst","United States, South Dakota, Spearfish","12/25/2020",""
```

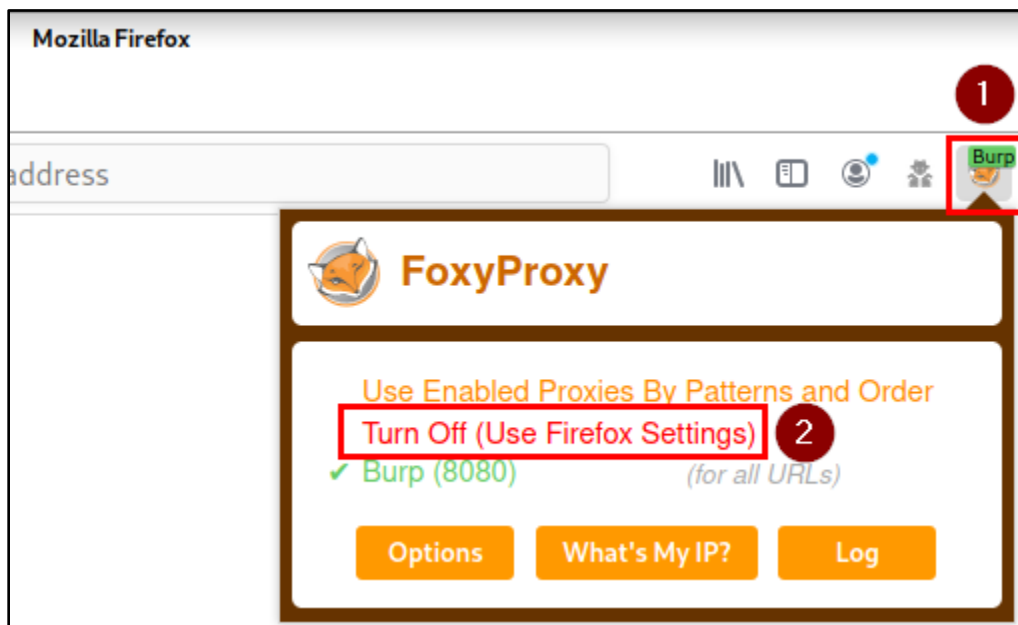
ZoomInfo Contact Data Logged by Burp

15. This information has also been saved to the "scraper-zoominfo.csv" file in the user's home directory. Like the LinkedIn results, this file can also be opened as a spreadsheet.

	A	B	C	D
1	Jason Blanchard	Director, Content & Community	United States, South Dakota, Spearfish	12/25/2020
2	Lisa Woody	Software Engineer	United States, South Dakota, Spearfish	12/25/2020
3	Laura Seumanutafa	Director, Logistics	United States, South Dakota, Spearfish	12/25/2020
4	Kevin Klingbile	Security Analyst	United States, South Dakota, Spearfish	12/25/2020
5	Ethan Robish	Security Consultant	United States, South Dakota, Spearfish	12/25/2020
6	Ralph May	Security Analyst	United States, South Dakota, Spearfish	12/25/2020
7	John Strand	Owner and Security Analyst	United States, South Dakota, Spearfish	12/25/2020
8	Derrick Rauch	Contractor	United States, South Dakota, Spearfish	12/25/2020
9	James Lee	Hacker	United States, South Dakota, Spearfish	12/25/2020
10	Cj Cox	Chief Operating Officer	United States, South Dakota, Spearfish	12/25/2020
11	Melisa Wachs	Director	United States, South Dakota, Spearfish	12/25/2020
12	Melissa Bruno	Security Analyst	United States, South Dakota, Spearfish	12/25/2020
13	Jennifer Creed	Position In Accounting	United States, South Dakota, Spearfish	12/25/2020
14	Bryan Strand	Manager, Business Capture & A Consultant	United States, South Dakota, Spearfish	12/25/2020
15	Beau Bullock	Security Analyst	United States, South Dakota, Spearfish	12/25/2020
16	Heather Doerges	Project Scheduler & Manager	United States, South Dakota, Spearfish	12/25/2020
17	Marcello Salvati	Security Analyst	United States, South Dakota, Spearfish	12/25/2020
18	Deb Wigley	Manager, Content & Community	United States, South Dakota, Spearfish	12/25/2020
19	Brian King	Security Analyst	United States, South Dakota, Spearfish	12/25/2020
20	Luke Baggett	Security Analyst	United States, South Dakota, Spearfish	12/25/2020
21	Craig Vincent	Security Analyst	United States, South Dakota, Spearfish	12/25/2020
22	Kent Ickler	Security Analyst & Systems Administrator	United States, South Dakota, Spearfish	12/25/2020
23	Kelsey Bellew	Security Analyst	United States, South Dakota, Spearfish	12/25/2020
24	Michael Allen	Penetration Tester and Security Analyst	United States, South Dakota, Spearfish	12/25/2020
25	Justin Angel	Security Analyst	United States, South Dakota, Spearfish	12/25/2020

ZoomInfo Data Opened in LibreOffice Calc

16. You can now close the Burp Suite window. In Firefox, use the FoxyProxy menu to turn off proxying.



Turn Off Proxying with FoxyProxy

3. Turn the employee names you collected into email addresses

1. Next, you'll need to convert the list of names you've collected into email addresses. To help with collecting first and last names from the CSV files generated in the previous section, a script named "scrape2names" is included with your Kali VM. Run the command below in a terminal window to generate a list of first and last names from the scraped data.

```
cat scraper-linkedin.csv scraper-zoominfo.csv | scrape2names | tee ~/first-last-names.txt
```

```
(kali@kali)-[~]
└─$ cat scraper-linkedin.csv scraper-zoominfo.csv | scrape2names | tee ~/first-last-names.txt
Abby Rabenberg
Adam Parsons
Alexander King
Beau Bullock
Benjamin Cavallaro
Benjamin Donnelly
Ben Kaufman
Black Official
Brad Geesaman
Bretton Tan
Brian Anderson
Brian Fehrman
Brian King
```

Names Collected from CSV Files

- Once you've generated the "first-last-names.txt" file, use the "username.py" script to generate email addresses from the list of names. The "username.py" script accepts Python3 format strings to define how the first and last name are modified when generating the output. Some example commands and output are included below.

Example of generating {First Initial}{Last Name}@contoso.com:

```
username.py ~/first-last-names.txt '{first[0]}{last}@contoso.com'
```

```
(kali@kali) - [~]
└─$ username.py ~/first-last-names.txt '{first[0]}{last}@contoso.com'
ARabenberg@contoso.com
AParsons@contoso.com
AKing@contoso.com
```

Generation of Email Addresses in the Format, {First Initial}{Last Name}@contoso.com

Example of generating {First Name}.{Last Name}@acme.com:

```
username.py ~/first-last-names.txt '{first}.{last}@acme.com'
```

```
(kali@kali) - [~]
└─$ username.py ~/first-last-names.txt '{first}.{last}@acme.com'
Abby.Rabenberg@acme.com
Adam.Parsons@acme.com
Alexander.King@acme.com
```

Generation of Email Addresses in the Format, {First Name}.{Last Name}@acme.com

4. Generating email addresses from common first and last names

In situations where you haven't found many employee names relative to the size of the target organization, you might want to make additional guesses. To generate additional names, lists of common first and last names collected by the U.S Census Bureau and Social Security Administration have been included in the "/opt/name-lists" folder on your Kali VM. The "brute-user.py" script is also included to help with generating email addresses from the lists.

The "brute-user.py" script works very similarly to "username.py", and examples of usage are included below.

Example of generating {First Name}.{Last Name}@example.com

```
brute-user.py /opt/name-lists/first_names-male_and_female.txt /opt/name-lists/last_names-top_100.txt '{first}.{last}@example.com' | head
```

```
(kali@kali) - [~]
└─$ brute-user.py /opt/name-lists/first_names-male_and_female.txt /opt/name-lists/last_names-top_100.txt '{first}.{last}@example.com' | head
james.smith@example.com
james.johnson@example.com
james.williams@example.com
james.brown@example.com
james.jones@example.com
james.garcia@example.com
```

Brute-Force Email Address Generation in the Format, {First Name}.{Last Name}@example.com

Example of generating {First Initial}{Last Name}@bhis.co:

```
brute-user.py /opt/name-lists/first_names-male_and_female.txt /opt/name-  
lists/last_names-top_100.txt '{first[0]}.{last}@bhis.co'
```

```
(kali㉿kali) - [~]  
└─$ brute-user.py /opt/name-lists/first_names-male_and_female.txt /opt/name-lists/last_names-top_100.txt '{first[0]}.{last}@bhis.co' | head  
j.smith@bhis.co  
j.johnson@bhis.co  
j.williams@bhis.co  
j.brown@bhis.co  
j.jones@bhis.co  
j.garcia@bhis.co  
j.miller@bhis.co  
j.davis@bhis.co  
j.rodriquez@bhis.co  
j.martinez@bhis.co
```

Brute-Force Email Address Generation in the Format, {First Initial}{Last Name}@bhis.co