SharePoint Federation and Extranet Workshop
Unit 4
Office 365

Presented by Peter Carson
President, Envision IT

June 18, 2014

envision IT
Leaders in SharePoint Solutions

Agenda

Morning Session

- Unit 0 – Introductions
- Unit 1 – Federation
- Unit 2 – Reference Scenarios
- Unit 3 – Customer Case Studies

Afternoon Session

- Unit 4 - Office 365
- Unit 5 – SharePoint On Premises
- Unit 6 – Open Discussion

Office 365 Authentication Options

Cloud Identity

Windows Azure Active Directory

No Integration

Directory and Password Synchronization

Windows Azure Active Directory

DirSync and Password Sync

On Premise Identity

Integration with no federation

Federated Identity

Windows Azure Active Directory

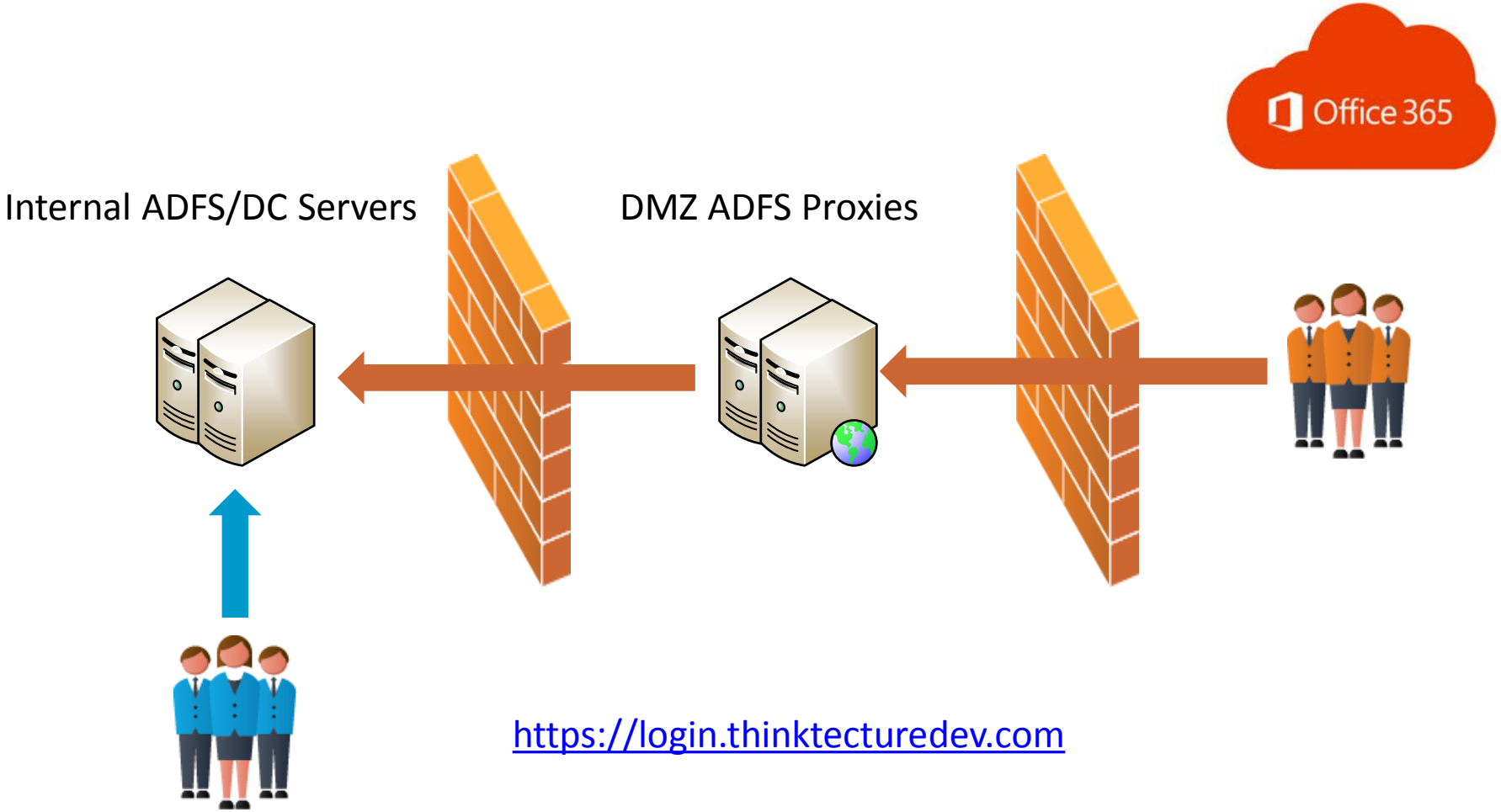
Federation

User Sync

On Premise Identity

Single federated identity and credentials

ADFS Servers



ADFS Server Considerations

- Should have at least two each of ADFS and ADFS Proxy Servers
- If there is only one Internet connection, consider putting them in Azure
- Having Lync and Exchange in the cloud and not being able to authenticate because your Internet pipe is down is a problem

ADFS Proxy

- ADFS Proxy role is part of ADFS 2.0 and 2.1
  - Windows Server 2008 SP2 or 2012
- Replaced with WAP in ADFS 30
  - Windows Server 2012 R2
  - Web Application Proxy
  - Reverse Proxy replacement for the discontinued Microsoft Forefront UAG and TMG

Setting Up Your Domain With Office 365

- DirSync is still used to create the shadow WAAD accounts
- Set up a trust between ADFS and Windows Azure AD
  - Setup through PowerShell
  - Requires you to setup some TXT DNS records to prove you own the domain
- Accounts need to be licensed in the Office 365 portal
- Useful links
  - http://technet.microsoft.com/en-us/library/jj205462.aspx
  - http://technet.microsoft.com/library/jj151815.aspx

Setting Up Your Domain With Office 365

1. Open the Windows Azure Active Directory Module.
2. Run $cred=Get-Credential. When the cmdlet prompts you for credentials, type your cloud service administrator account credentials.
3. Run Connect-MsolService –Credential $cred. This cmdlet connects you to Windows Azure AD. Creating a context that connects you to Windows Azure AD is required before running any of the additional cmdlets installed by the tool.
4. Run Set-MsolAdfscontext -Computer <AD FS primary server>, where <AD FS primary server> is the internal FQDN name of the primary AD FS server. This cmdlet creates a context that connects you to AD FS.
5. Run New-MsolFederatedDomain –DomainName <domain>, where <domain> is the domain to be added and enabled for single sign-on. This cmdlet adds a new top-level domain or subdomain that will be configured for federated authentication.
6. Using the information provided by the results of the New-MsolFederatedDomain cmdlet, contact your domain registrar to create the required DNS record. This verifies that you own the domain. Note that this may take up to 15 minutes to propagate, depending on your registrar. It can take up to 72 hours for changes to propagate through the system. For more information, see Verify a domain at any domain name registrar.
7. Run New-MsolFederatedDomain a second time, specifying the same domain name to finalize the process.

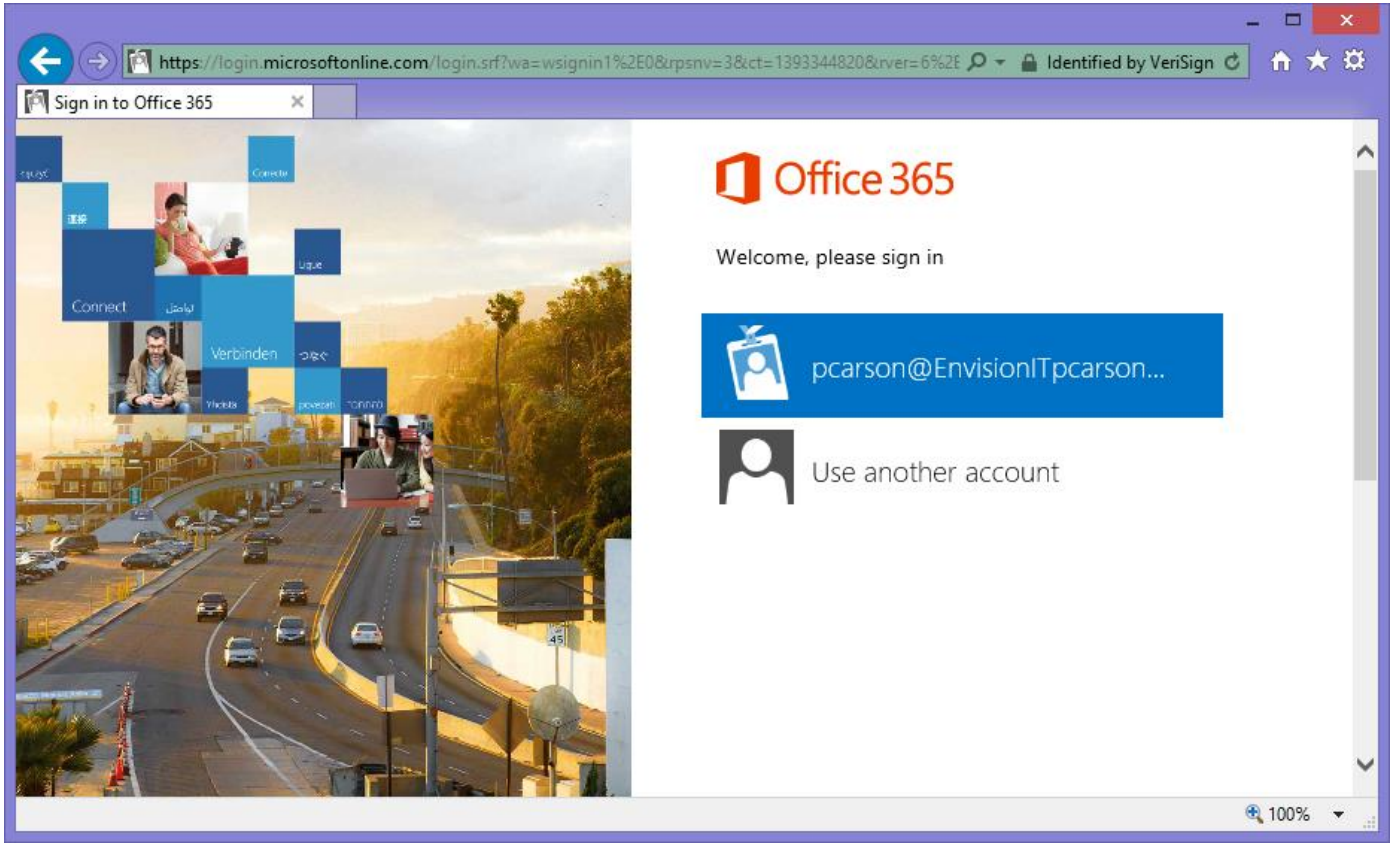
Example PowerShell

```
import-module MSOnline

$msolcred = get-credential
connect-msolservice -credential $msolcred

Get-MsolUser

Get-MsolDomain

Get-MsolDomainVerificationDns -DomainName envisionit.com
```

Demo 1 Scenario

- Sample site at https://envisionit.sharepoint.com/sites/eumwebinar
- Internal user in authenticated automatically through the ADFS Server
- Need to go through the Microsoft home realm discovery page
  - This can be overridden for Exchange Online but not SharePoint Online (yet)

Microsoft Home Realm Discovery



Smart Links

- Run Fiddler as you are logging in
- Capture the 302 redirect
  - https://fedsrv.envisionit.com/adfs/ls/?cbcxt=&popupui=&vv=&username=peter%40envisionit.com&mkt=&lc=1033&wfresh=&wa=wsignin1.0&wtrealm=urn:federation:MicrosoftOnline&wctx=wa%3Dwsignin1%252E0%26rpsnv%3D3%26ct%3D1395748967%26rver%3D6%252E1%252E6206%252E0%26wp%3DMBI%26wreply%3Dhttps%253A%252F%252Fenvisionit%252Esharepoint%252Ecom%252F%255Fforms%252Fdefault%252Easpx%26lc%3D1033%26id%3D500046%26%26bk%3D1395748967%26LoginOptions%3D3
- Remove the highlighted text
- Add the desired URL as double-encoded text

Smart Links

https://fedsrv.envisionit.com/adfs/ls/?wa=wsignin1.0&wtrealm=urn:federation:MicrosoftOnline&wctx=wa%3Dwsignin1%252E0%26rpsnv%3D3%26ct%3D1395748967%26rver%3D6%252E1%252E6206%252E0%26wp%3DMBI%26wreply%3Dhttps%253A%252F%252Fenvisionit%252Esharepoint%252Ecom%252Fsites%252Feumwebinar

Character	Double-Encoded Value
:	%253A
.	%252E
/	%252F

- A friendly URL and redirect can be setup to hide the complex URL
- Our Custom 404 product is a good option for this
- https://productdemo.envisionit.com/eumwebinar

Extranets in Office 365

- External sharing using Microsoft accounts or other federated users
- Licensing of external users as subscribers

External Sharing

Pros

- Supported by default by Office 365
- Up to 10,000 external users can access a SharePoint Online site for free using Microsoft accounts
- Also works for other federated users with subscriptions

Cons

- Need to enable external sharing
- Email invitations are sent out
- Can be authenticated or anonymous
- Hard to hide the Microsoft experience
- No groups or central management

Licensed External Users

Pros

- Centrally managed
- Group permissions
- Full control over the login experience

Cons

- Need to pay for a subscription

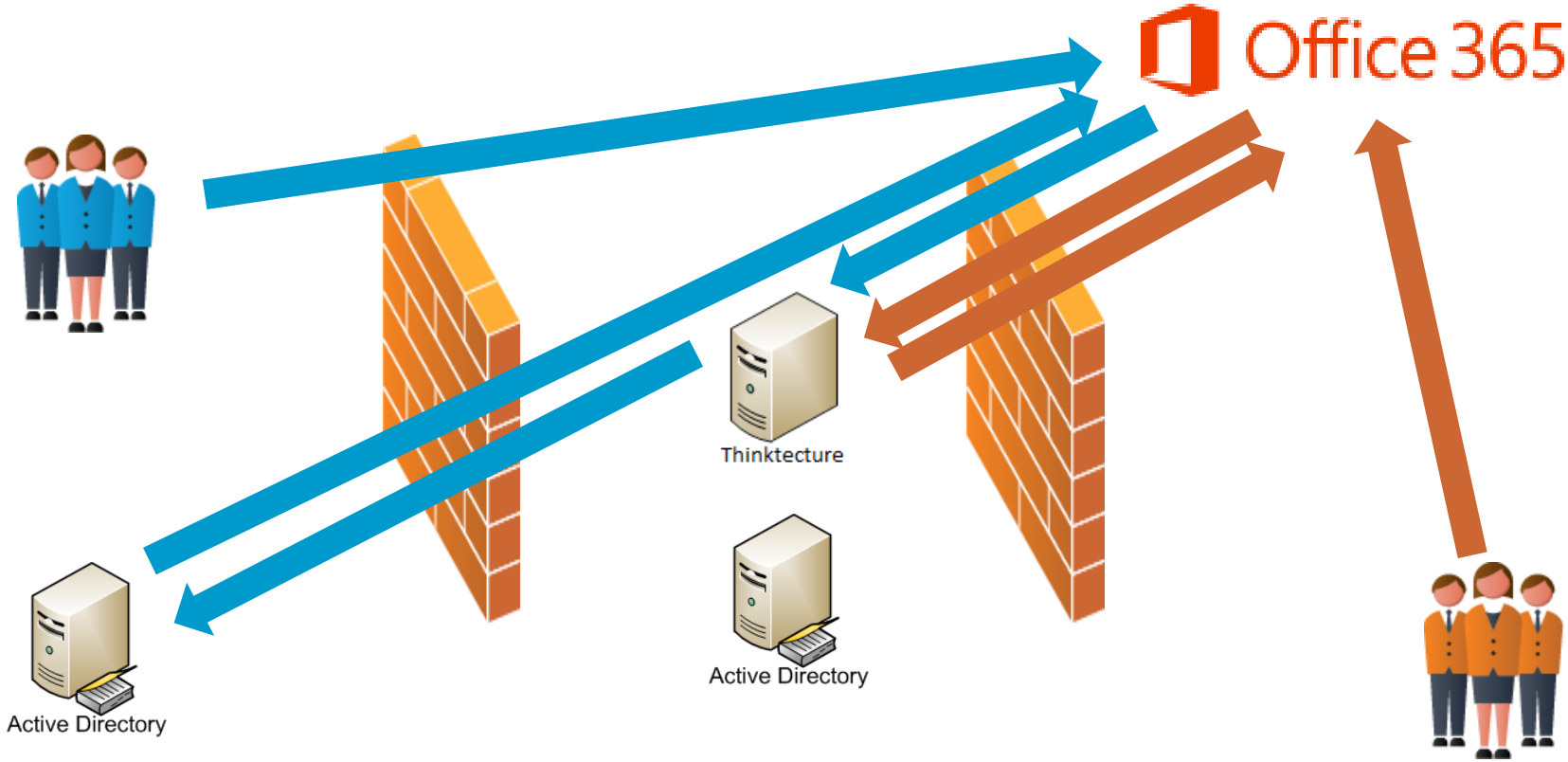
Demo 2 Scenario

- Sample site at https://thinktecturedev.sharepoint.com
- Federated with Thinktecture Identity Server
- Customized to work with Office 365
- Login is customized with branding and ability to login with email address
- Can still use ADFS for internal users
- Smart link is https://productdemo.envisionit.com/thinktecturedev

Why Thinktecture over ADFS?

- Open source allows any customization
- Fully brandable (ADFS allows branding within very particular parameters)
- Login with email address instead of AD username
- Use SQL instead of AD as the underlying user repository
- Ability to incorporate the home realm discovery into the login form

Authentication Process



Managing Your External O365 Users with EUM

- Delegate user management internally or externally to your organization
- Self-registration and approvals
- Full control over the accounts and login experience
- Delegated group management simplifies permissions
- Lost password reset
- Improved governance over your Extranet

Demo 3

- Self-registration and approval of a new external user
- https://eum.eitdev.org/landing/register/register.aspx
- We have not fully implemented the Graph API yet
- Licensing of the new user is proving problematic
- Using PowerShell for now

```
Get-MsolAccountSku | format-Table AccountSkuID, SkuPartNumber

$User = get-aduser userid
$immutableId = [convert]::ToBase64String($User.ObjectGuid.ToByteArray());
$DisplayName = $User.GivenName + " " + $User.Surname

New-MsolUser -DisplayName $DisplayName -UserPrincipalName $User.UserPrincipalName -ImmutableId
$immutableId -UsageLocation CA

Set-MsolUserLicense -UserPrincipalName $User.UserPrincipalName -AddLicenses
"Thinktecturedev:SHAREPOINTSTANDARD_YAMMER"

Get-MsolUser | select UserPrincipalName, DisplayName, isLicensed
```

Windows Azure AD Graph API

- Provides programmatic access to Azure AD through REST API endpoints
  - Create a new user in a directory
  - Get a user's detailed properties, such as their groups
  - Update a user's properties, such as their location and phone number, or change their password
  - Check a user's group membership for role-based access
  - Disable a user's account or delete it entirely
- Assigning subscriptions has been the challenge

Apps and Office 365

Apps and Office 365

- Three main types of Apps
  - SharePoint Hosted
    - Client side code only
  - Auto Hosted
    - Server code runs in an Azure instance provided by Office 365
    - No longer available as Preview has been ended by Microsoft
  - Provider Hosted
    - Use your own server environment to host your server side code
    - Doesn’t need to be Microsoft technology

Apps and Office 365

- No App code ever runs on the SharePoint farm
- Apps are selected and installed by the end user
- Need to explicitly trust the app to allow it to run
- OAuth is used to provide the end-user's authentication to the app and back to SharePoint

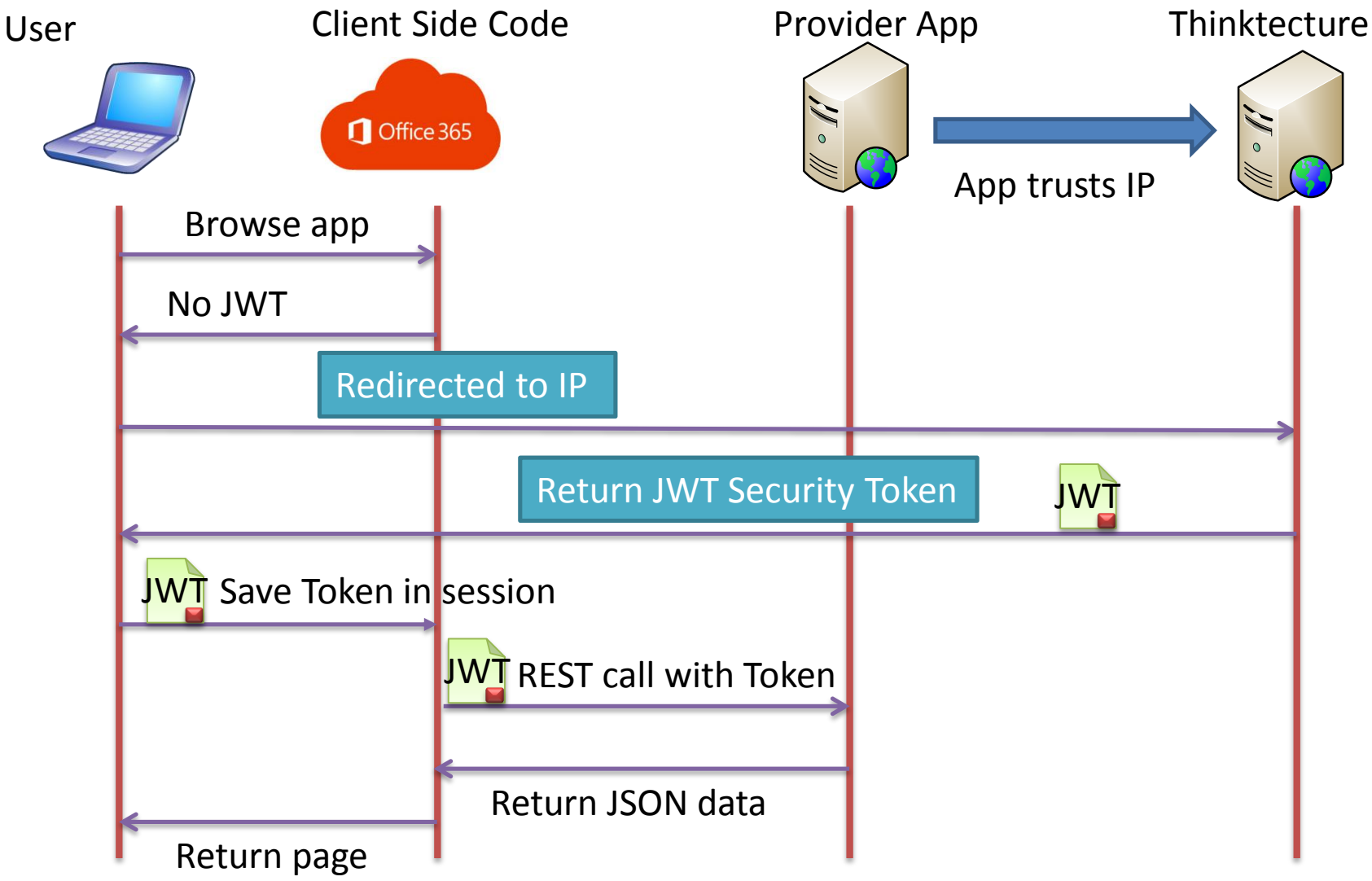
Challenges with SharePoint Apps

- For full functionality, apps need to be installed in each site where they are being used
- No way to programmatically install them
- This is a problem for apps that are used on many sites

Alternative App Model

- Client side code and REST APIs is the direction Microsoft is taking in general
- Use this approach for Apps too
- If Office 365 is authenticated using Thinktecture, that can be leveraged to authenticate provider hosted apps too
- Thinktecture can provide a JSON Web Token (JWT) to the client-side code
  - Similar to a SAML token
  - It is the model going forward with WebAPI
- This can be passed to and trusted by the REST API for authentication

App Authentication Process with JWT



App Development Architecture

- Client side development done in AngularJS
  - Open-source platform originally developed by Google
  - Visual Studio support is being built in by Microsoft
  - Ideal for presentation logic
- Provider-hosted server side code REST APIs
  - Built in Visual Studio as a WebAPI project
  - Best place to put business logic and data access
  - Can be a high-trust app using certificates to authenticate to SharePoint