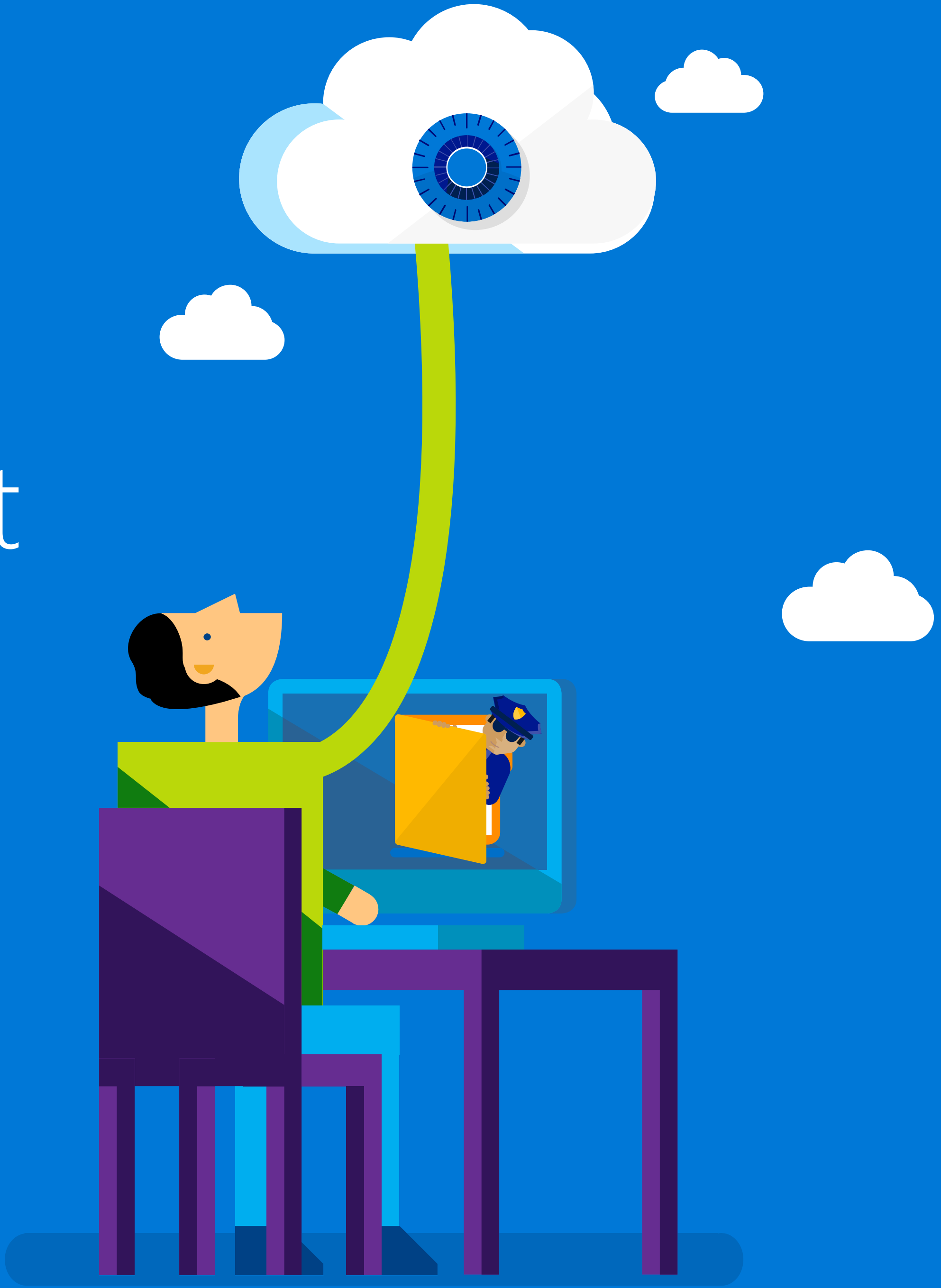
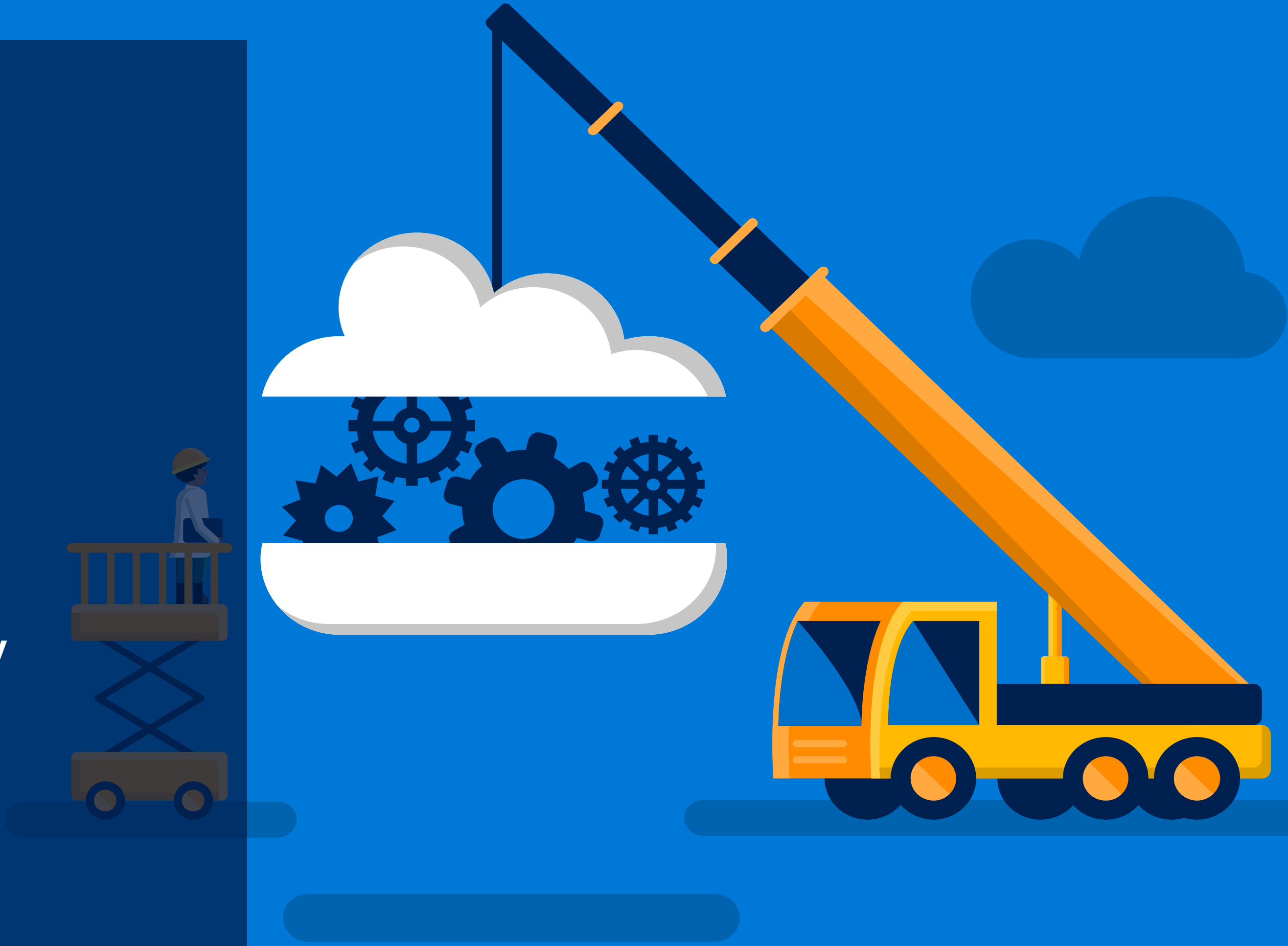
Securing Applications on Azure - What to do and what to avoid

William Eastbury
Technical Evangelist



Introduction and Housekeeping

Welcome to Microsoft UK,
2 Kingdom Street



Housekeeping

1. Fire Alarms
2. Emergency Exits
3. Toilets
4. Lunch and Dietary Requirements
5. Feedback forms and today's website http://aka.ms/azureevent (code = 901031)
6. Group Discussion and 1-2-1 Office Hours

Microsoft

Today's Brief.

Explore the top critical considerations when architecting services that are secure by design, both at the network layer, at the application level and with regard to Azure Directory Services.

Answer the twin questions of:-

"How do I identify and authorise users in the cloud?"
"What should I use to secure my applications and services?"

Microsoft

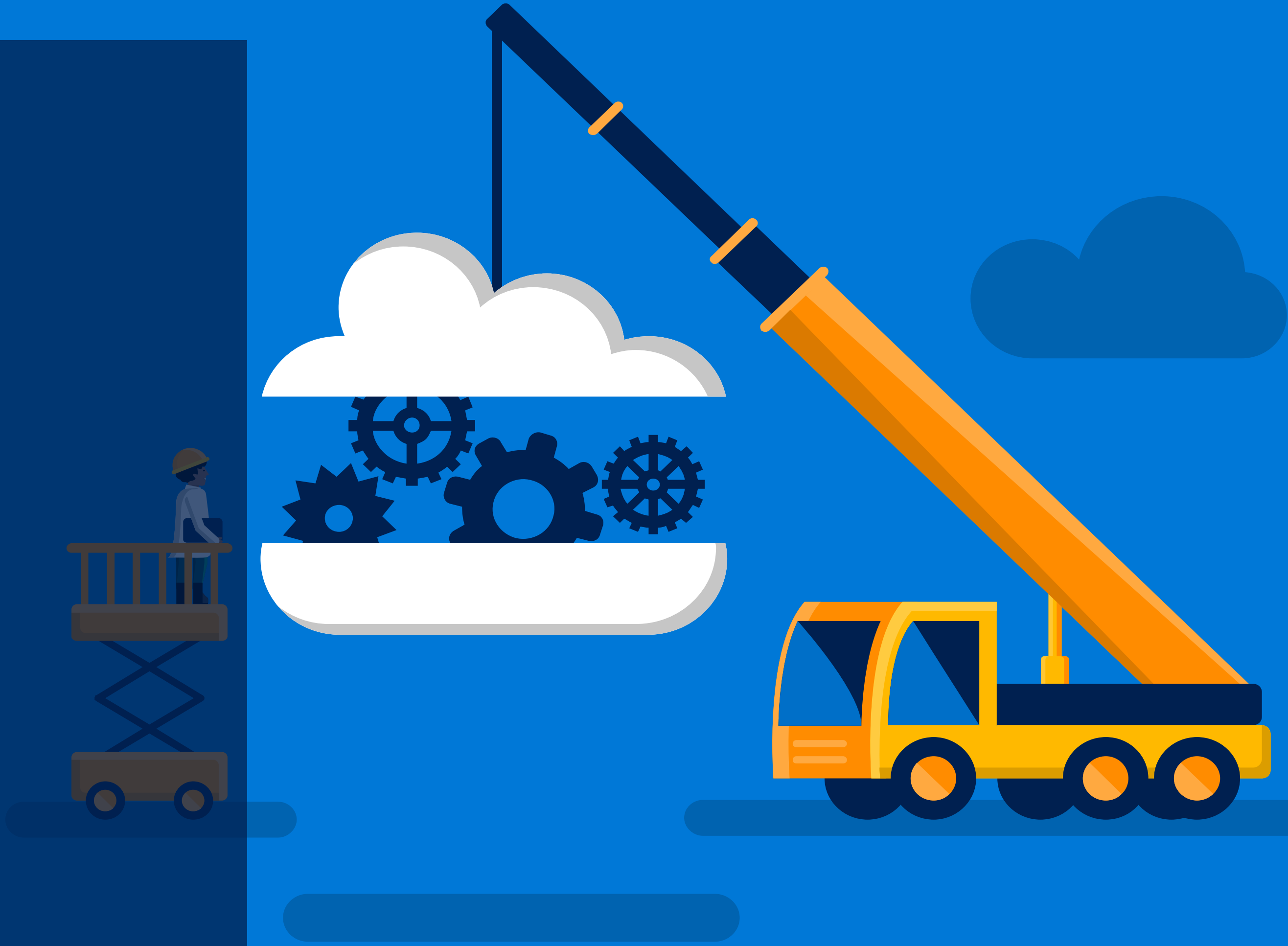
Today’s Agenda

10:00 am: Introduction and Housekeeping – Will Eastbury
10:15 am: The threat ecosystem – Phil Winstanley
10:45 am: Cloud Security Patterns and Practices – Will Eastbury
11:00 am: Coffee Break
11:15 am: Directory Security - IDaaS with Azure AD and OpenID Connect – Will Eastbury
11:30 pm: Trust and Blockchain technologies – Mike Ormond
12:00 pm: Azure Active Directory Application Security in action - Securing a multi-tier app using AAD – Ben Roscorla
12:20 pm: ID/PaaS - Azure App Service EasyAuth – Will Eastbury
12:30 pm: Lunch
1:15 pm: Cloud Security Principles – Phil Winstanley
1:30 pm: Azure Perimeter Network Security, WAFs, Vnets + NSGs
Azure Subscription Security, Azure AD, ARM, Resource Groups, RBAC, Resource Locks – Will Eastbury
2:15 pm: Expert Panel QA
2:45 pm: Expert-Led Group Discussion
3:15 pm: Close

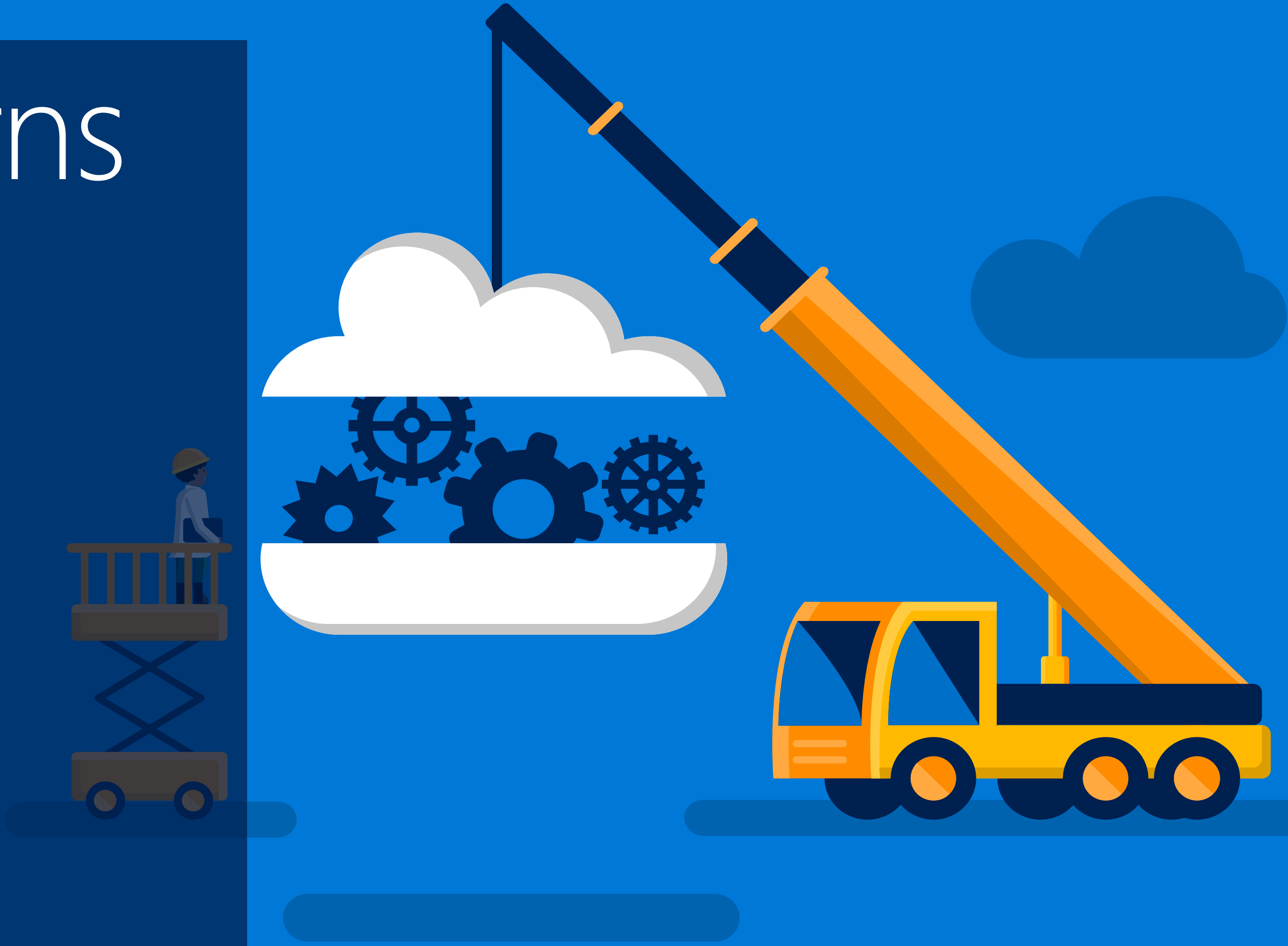
Microsoft

The Threat ecosystem

Phil Winstanley
Microsoft
Premier Developer



Cloud Security Patterns and Learnings



Will Eastbury
Microsoft
Developer Experience

Federated Identity Pattern

Simply delegate *authentication* to an external identity provider, such as Azure AD.

This pattern can simplify development, minimize the requirement for user administration, and improve the user experience of the application.

Why roll your own authentication services when there are highly secure, well tested and stable authentication services available for free*.

Note that this pattern is also sometimes used to refer to identity delegation *between providers* where there is a trusted identity that resides in an external domain, such as and on-premise directory accessed over ADFS or Azure AD B2B.

Microsoft

Gatekeeper Pattern

Protects applications and services by using a *dedicated* host instance that acts as a broker between clients and the application or service, validates and sanitizes requests and passes requests and data between them.

This pattern can provide an additional layer of security, and limit the attack surface of the system.

This is usually implemented as a PaaS service (such as Azure Application Gateway’s WAF tier) or a VM / IaaS Web application firewall – such as a Barracuda or F5 Firewall appliance.

Microsoft

Valet Key Pattern

Use a token or key that provides clients with restricted direct access to a specific resource or service in order to offload data transfer operations from the application code.

This pattern is useful in applications that use cloud-hosted storage systems or queues, and can minimize cost and maximize scalability and performance, without compromising security by opening up the entire system publicly for writes by anyone.

A common use of this pattern is with Azure Service Bus or Azure Storage, to issue a short-lived access key to write a message to a queue or topic, or to directly read part of a table – this style of usage is known as a “Shared Access Signature”.

Microsoft

Links and Downloads

Free Cloud Design Patterns ebook
http://www.microsoft.com/en-us/download/details.aspx?id=42026

Cloud Design Patterns Poster
https://azure.microsoft.com/en-us/documentation/infographics/cloud-design-patterns/

This session - Demos and Presentation Repository
https://github.com/dxuk/AzureWorkshops/

Coffee Break

Directory Security - IDaaS with Azure AD and OpenID Connect

Will Eastbury
Microsoft
Developer Experience

Key Identity Terms

Authentication (AuthN) – Determining if a (user or app) identity is as claimed by the principal.

Authorization (AuthZ) – Determining if a known and identified user has access to a resource.

Identity – a set of attributes related to an **entity** (human, machine, service)

Entity - might have many identities with a particular identity being shown depending on who or what we are talking to.

Identity Provider (IdP) – the service validating the identity of a user. Token issuer.

Relying Party (RP) – the Client application the user wants to access. The application relies on the IdP

User-Agent – the user's means of communicating with the RP e.g. a browser

Claim – a key-value pair returned in a token. Pertains to data related to the authentication ***or*** authorisation of the user e.g. a permission or session related information.

Access Token – a cryptographically signed set of Claims *with an expiry set in minutes*

Refresh Token – a token used to refresh an Access Token and whose *expiry is typically measured in days.*

ID Token – a kind of Access Token but the data is *specific to the **authentication** of a user* (only used by Open ID Connect flows)

Scope – indicates which resources and permissions/actions an app is requesting access to

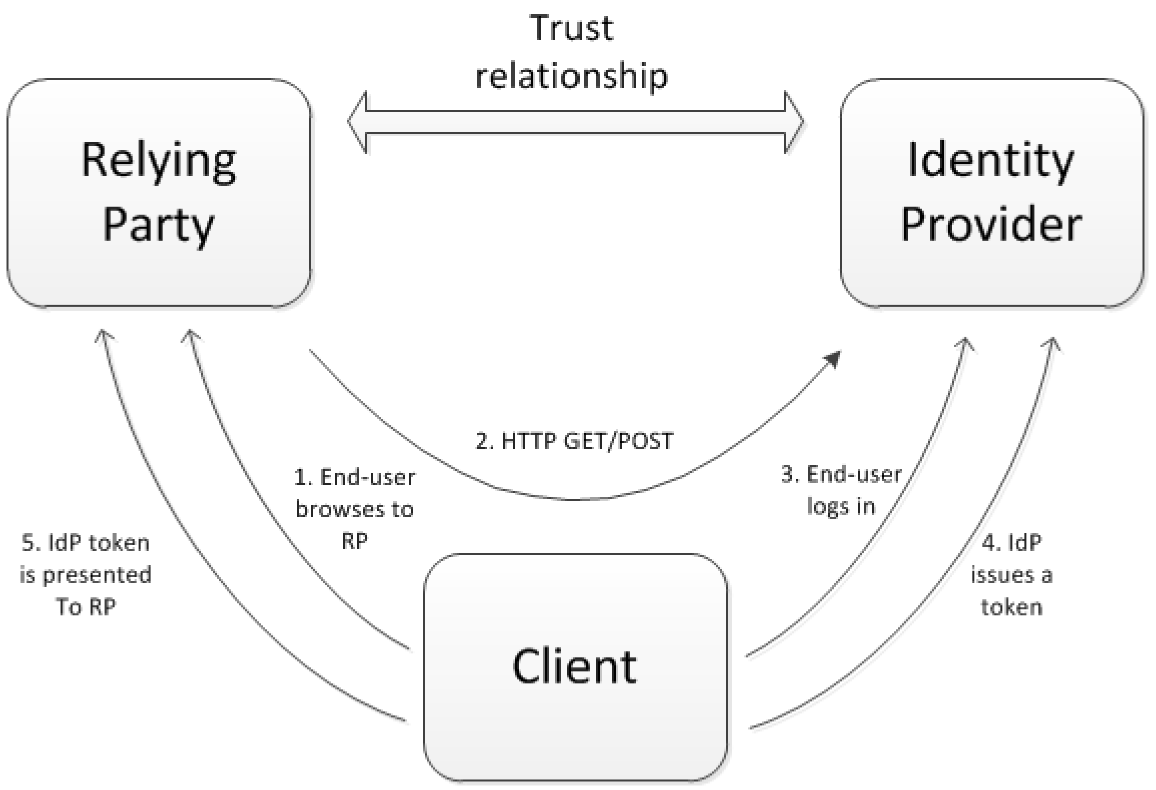
SAML

Security Assertion Markup Language is widely used for Single Sign On

XML based, open standard data format for exchanging authorization and authentication data between parties.

Permissions (**Claims**) are stored in a SAML token which is issued by a **Security Token Service** (aka **Identity Provider**).

SAML is widely used in **Windows Identity Foundation** which can be used to secure MVC applications and Web Services.



OAuth 1.0 & 2.0

Open Standard for ***Authorization*** created around 2006 for use by Twitter.

It specifies a process for resource owners to authorize 3rd party access to their services without sharing credentials.

Whereas SAML, WS-Federation and passwords are aimed at establishing the identity of the user (i.e. Authentication), OAuth is chief concern is determining **whether the caller is authorised to perform the operation (ie. Authorisation)**.

Introduces the concept of **access tokens** that are issued by third party clients by an authorization server.

OAuth 2.0 is not backwards compatible with 1.0. To simplify the developer experience message signing was removed and it relies completely on TLS for confidentiality and server authentication.

OpenID Connect

Simply an identity (Authentication) layer **on top of OAuth 2.0** protocol

Defines a new token - **ID Token** – a security token (JSON Web Token JWT) that contains Claims about the authentication of an End User.

ID Tokens must be signed using JSON Web Signature.

Use of this extension requires clients to specify “openid” as a scope in the authorization request.

An Identity Layer provides

WHO	is the user that got authenticated
WHERE	were they authenticated
WHEN	were they authenticated
HOW	were they authenticated
WHAT	attributes they can give you
WHY	they are providing them

To be interoperable…

Standard Scopes	• openid, profile, email, address, phone
Methods for requesting more granular claims	• Using the JSON Request Object
ID Token	• Info about the authenticated user
UserInfo endpoint	• Get attributes about the user • Translate the tokens

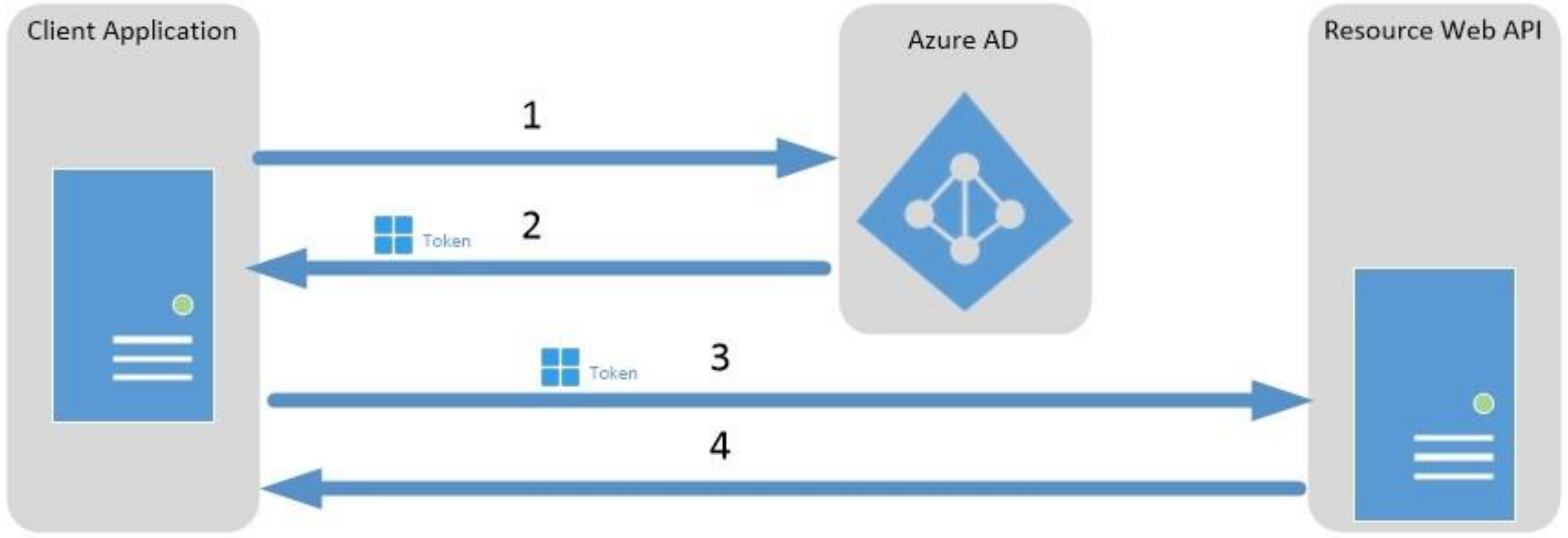
OAuth Flows (AKA Grants)

How access tokens are returned to the Client (Relying Party application)

Authorization Code	for applications running on a web server
Implicit	browser-based (JavaScript) or mobile apps
Client Credentials	server applications within a trusted domain
Password	username and password

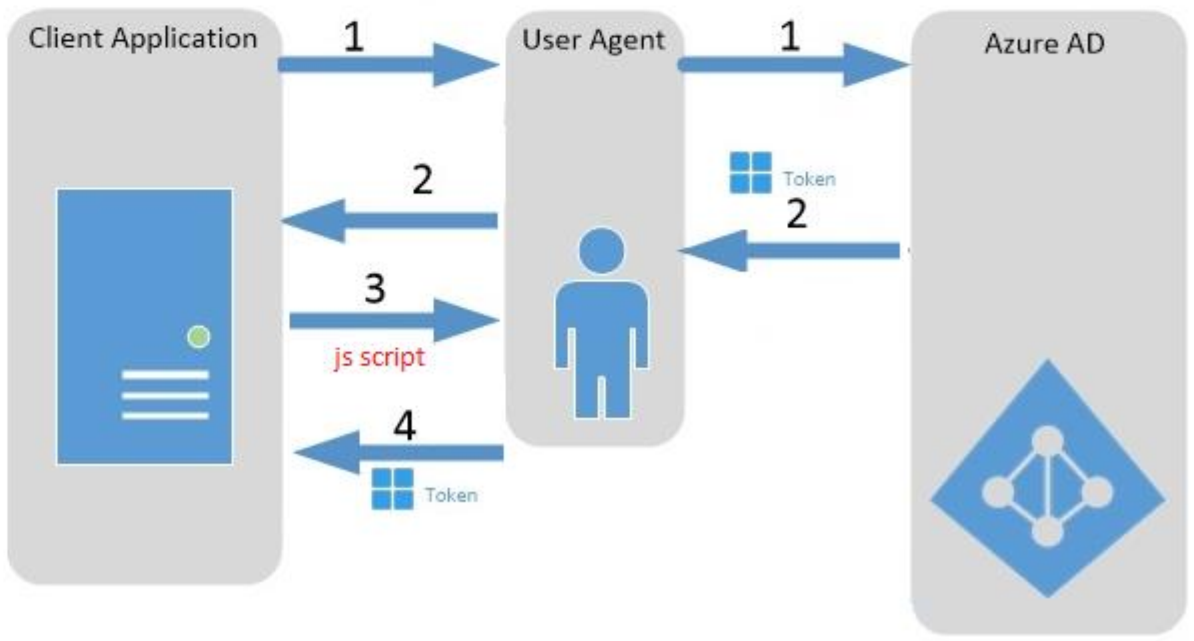
OAuth Flows – Client Credentials Flow

Access Token sent back to the client in exchange for id and secret therefore Client Application must be trusted



OAuth Flows – Implicit Flow

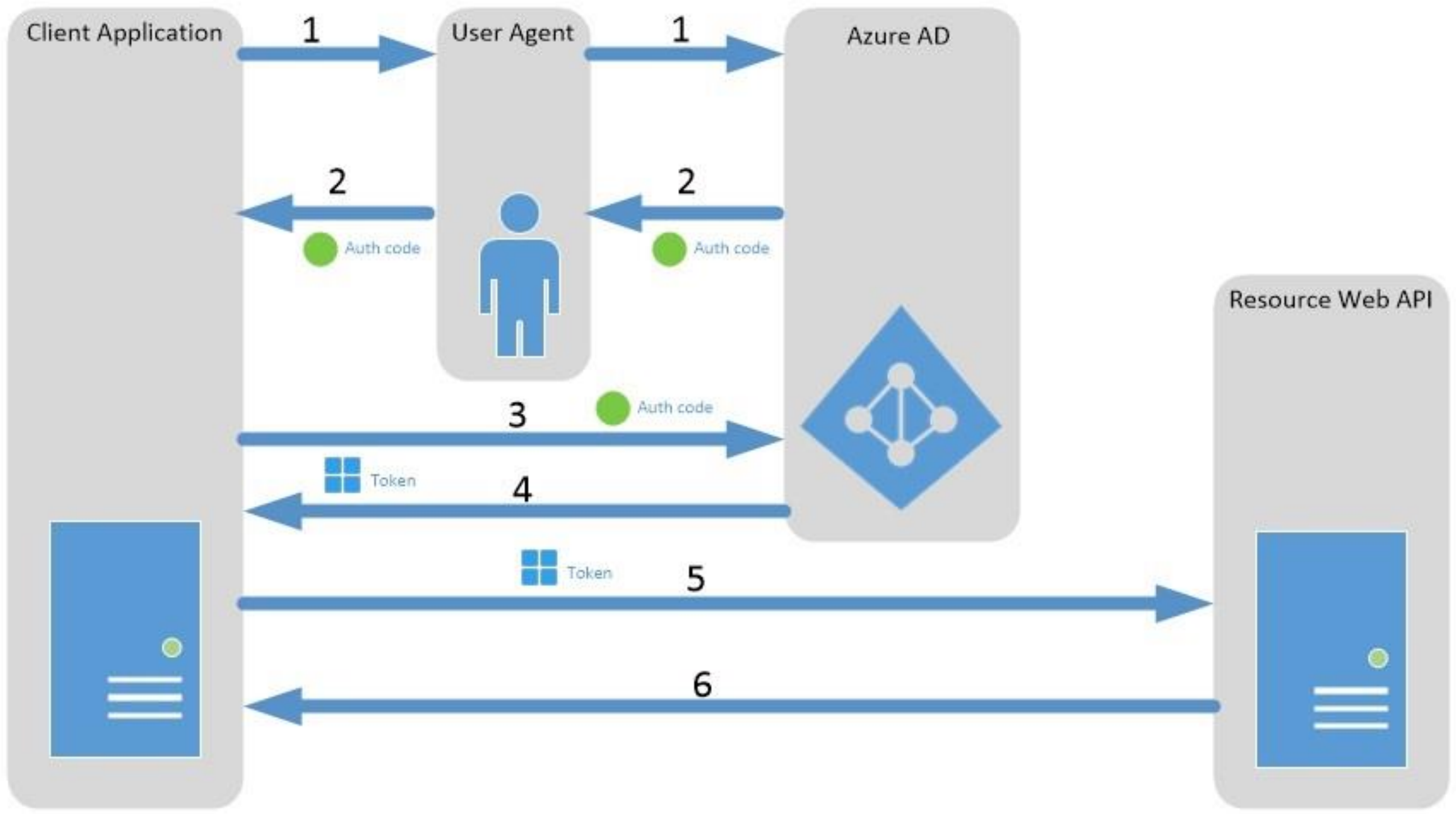
Access Token sent back to the client via the user agent, but the client application never sees the credentials.
Use this for insecure client apps, such as JavaScript SPA applications.



Taken from http://oauthlib.readthedocs.io/en/latest/oauth2/grants/implicit.html

OAuth Flows – Authorization Code Flow

Access Token sent back to the Client without being sent to the User Agent (e.g. a browser)



Open ID Connect Flows

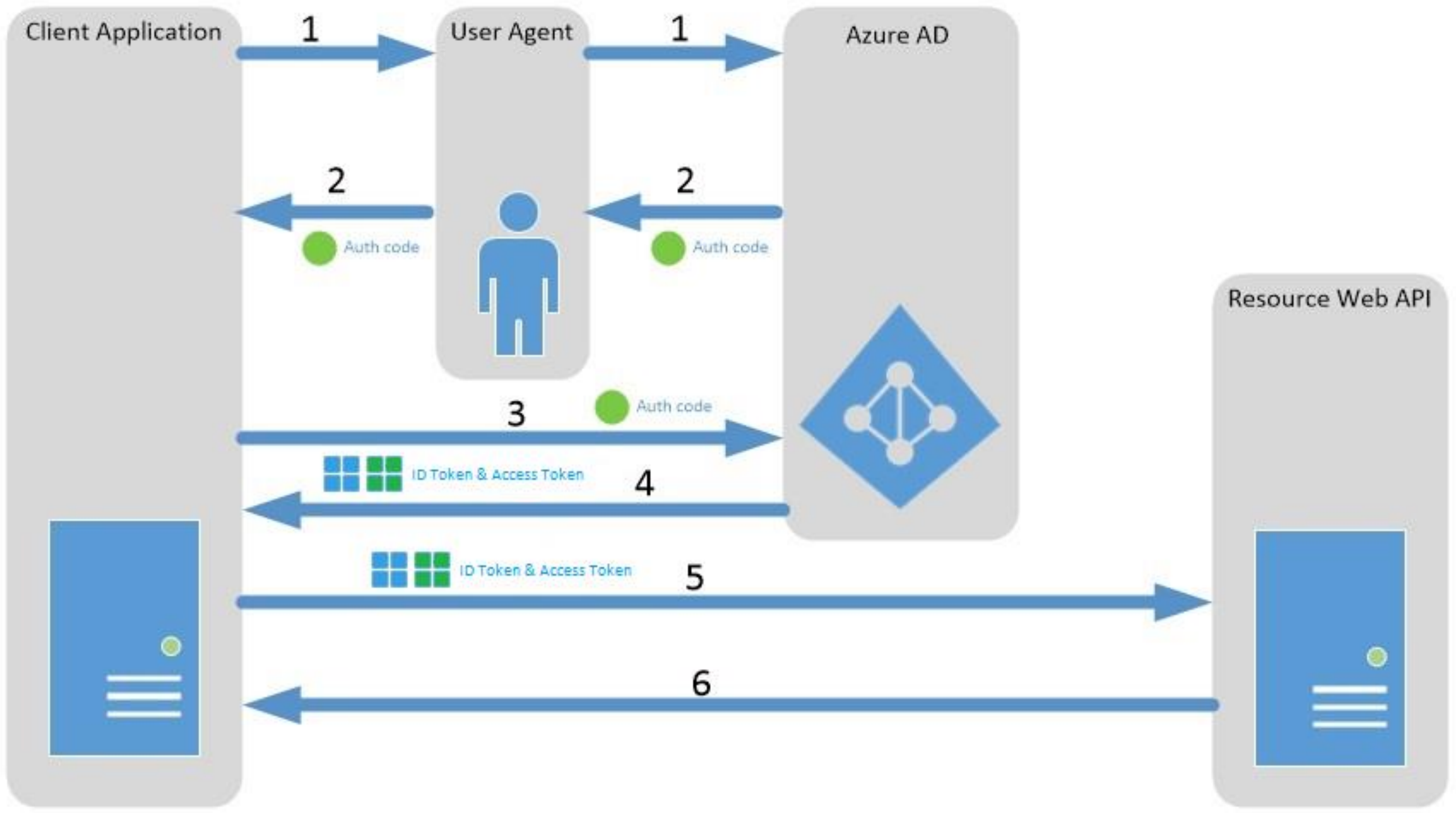
How ID Tokens & Access Tokens are returned to the Client (Relying Party application)

Authorization Code	response_type=code
Implicit	response_type=id_token token
Hybrid	response_type=code id_token

In the Hybrid flow the authentication response contains both the ID Token and Access Token.

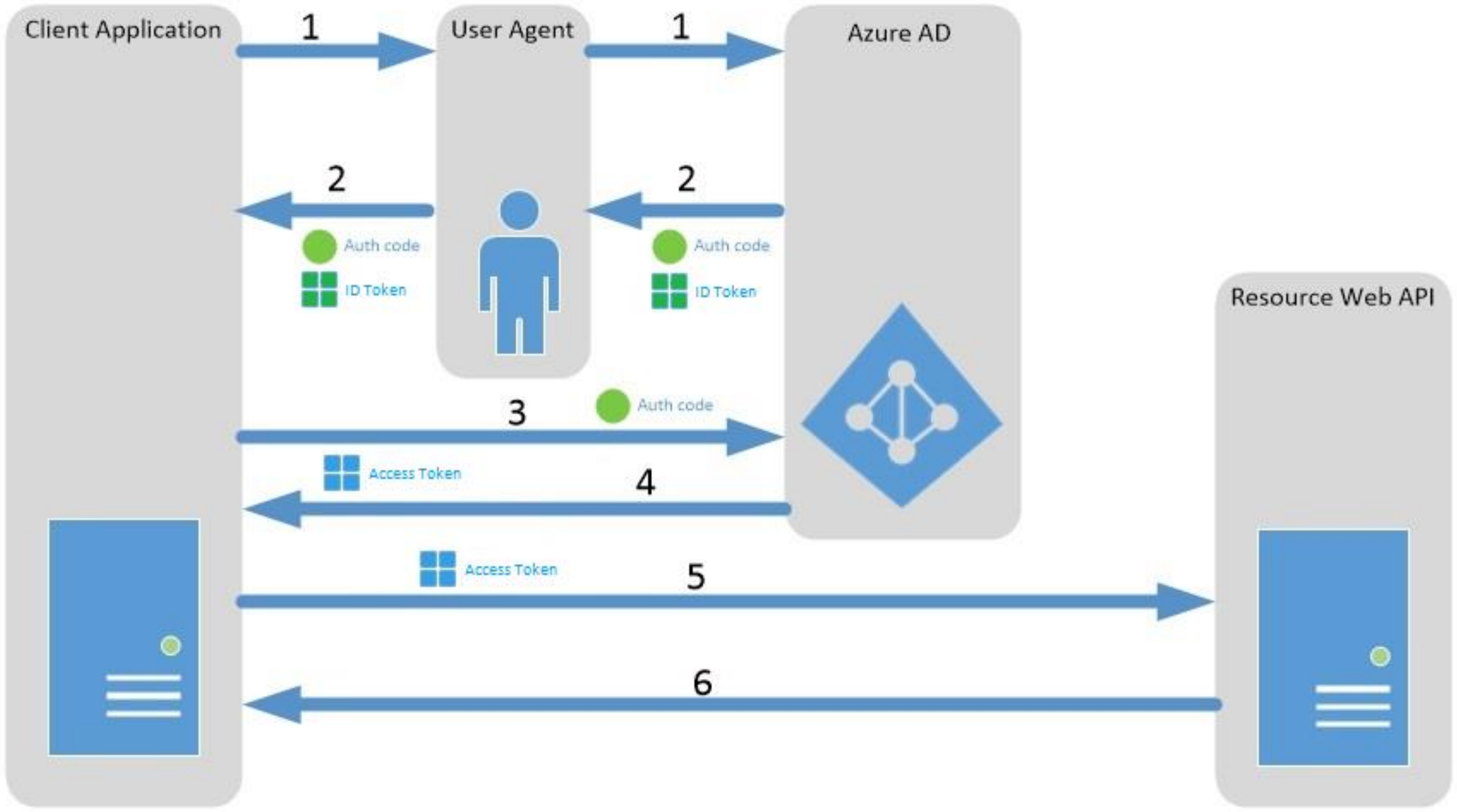
OpenID Connect – Authorization Code Flow

ID Token & Access Token sent back to the Client without being sent to the User Agent (e.g. a browser)



OpenID Connect – Hybrid Flow

ID Token & Access Token sent back to the Client via the User Agent (e.g. a browser)



Azure Active Directory

- Microsoft's "Identity Management as a Service (IDaaS)" for organizations.
- Millions of **independent** identity systems **controlled** by enterprise and government "tenants."
- Information is **owned and used by the controlling organization**—not by Microsoft.
- Born-as-a-cloud directory for Office 365. Extended to manage across many clouds.
- Evolved to manage an organization's relationships with its customers/citizens and partners (B2C and B2B).

>85%
of Fortune 500 companies use Microsoft Cloud (Azure, O365, CRM Online, and PowerBI)

Azure AD Directories
>10 M

More than
750 M
user accounts on Azure AD

1 trillion
Azure AD authentications since the release of the service

>110k
third-party applications used with Azure AD each month

>1.3 billion
authentications every day on Azure AD

Every Office 365 and Microsoft Azure customer uses Azure Active Directory

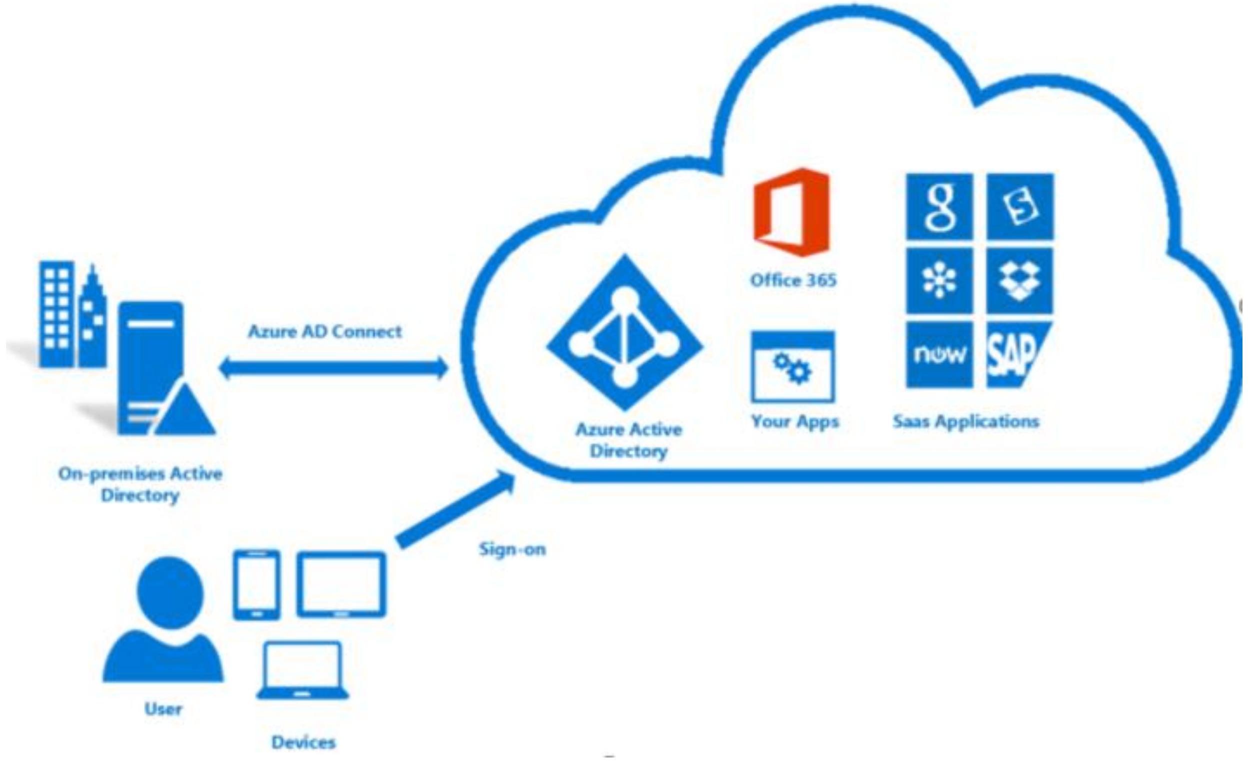
Single Sign On

Azure AD could be cloud-only or it could be a projection of an on-premises AD

Why? This allows users to access both Cloud and on-premise applications with a single identity plus Azure AD makes integration with Internet based applications much easier.

Identities can be synced from on-premises to Azure AD using Azure AD Connect

Password Hash Sync is optional and Azure AD Pass-through authentication can be used instead..



Microsoft Identity Libraries

OWIN (Open Web Interface for .NET) Middleware

- Protocol middleware that abstracts tedious protocol details from application and extract incoming tokens into *ClaimsPrincipal* object.
- Developers focus on business logic in a protocol agnostic way
- Available as Nuget Packages Microsoft.Owin.Security.*

ADAL (Active Directory Authentication Library)

- Enable Client to obtain access token against Azure AD or ADFS
- Provide asynchronous support, a configurable token cache and automatic token refresh
- Supports multiple platforms including .NET , iOS/OS X, Java, Android, JavaScript, Node.js

MSAL (Microsoft Authentication Library) – Preview

- Works with Azure AD, Microsoft Accounts, Azure AD B2C accounts
- Lesson learned from ADAL (e.g. improved cache for multi-tenant app)
- Support .NET today (and iOS and Android via Xamarin). More platforms in future.

Consent framework

Oauth2Permissions*

What actions client applications can do with the registered application.

Only comes into play if the application is a web API.

requiredResourceAccess*

Lists all the resources and permissions the application needs access to.

*These are partnered together. The permission id for the requiredResourceAccess value relates to the id of the oauth2Permission entry

prompt=admin_consent

By appending to the request to the authorisation endpoint, you can login as an admin to an application and consent for all users of a tenant.

Not necessary if a global admin registered the application in AAD.

Multi-tenant apps

Azure AD Common endpoint

https://login.microsoftonline.com/common

Used when the tenant id is not known

Care must be taken to validate tenants are allowed to your application via the IssuerValidator

Check the “application is multi-tenant” check box and your app will then show up in other tenants for pre-registration in Azure AD.

Azure AD B2B

Splits responsibility for Identity, Authorisation and Authentication between a partner and the customer.

Grant access to your applications, in your own tenant, where identity and Authentication resides in the partner tenant.

API or spreadsheet import process + invitation process to add users

Azure AD Graph API

How do I implement RBAC?

Provides programmatic access to Azure AD through OData REST API endpoints.

@ https://graph.windows.net

You can use the Graph API to :-

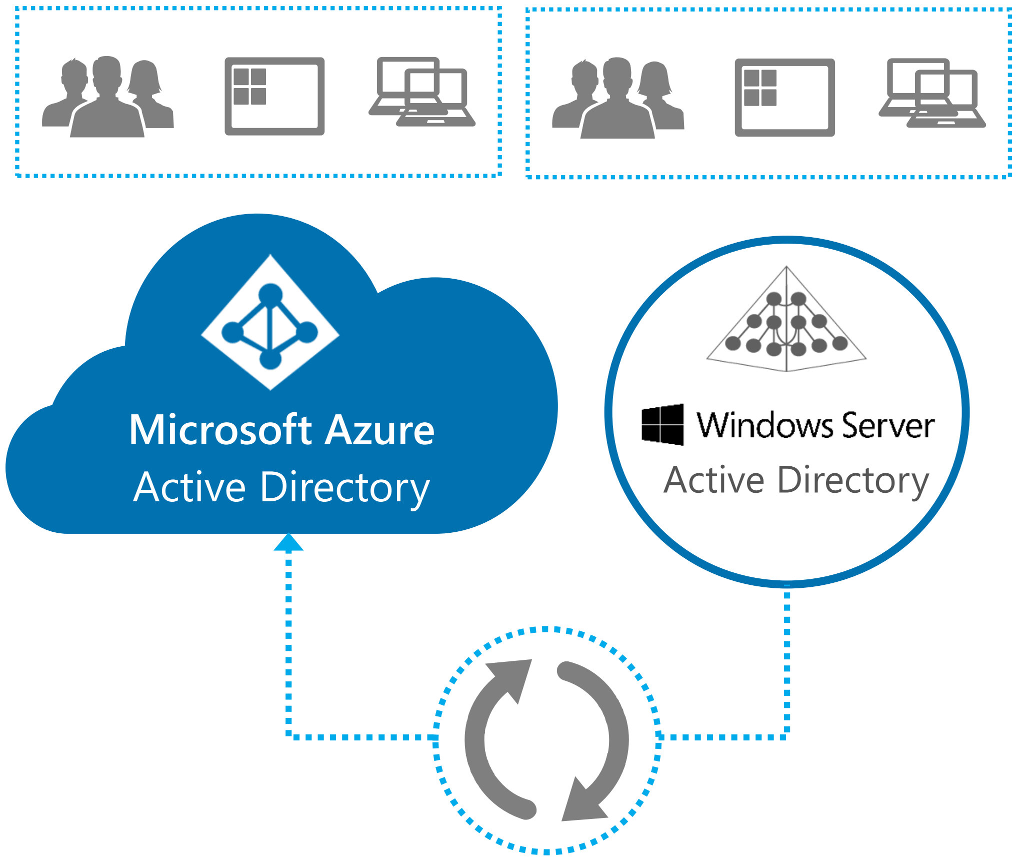
Create a new user
View or update user's properties
Change user's password
Check group membership for role-based access
Disable (or delete) a user.

Also Consider

Microsoft Graph API – Integrate Azure AD and Office 365 as one service call @ https://graph.microsoft.com

https://msdn.microsoft.com/Library/Azure/Ad/Graph/howto/azure-ad-graph-api-supported-queries-filters-and-paging-options#CommonQueries
https://graph.microsoft.io/en-us/graph-explorer

Identity & Access: Multi Factor Authentication



- ✓ Protect sensitive data and applications both on-premises and in the cloud with Multi Factor Authentication
- ✓ Can use Active Directory (on-premises) with Azure Active Directory (in cloud) to enable single sign-on, a single directory, and centralized identity management
- ✓ Multi Factor Authentication can be implemented with Phone Factor or with AD on-premises

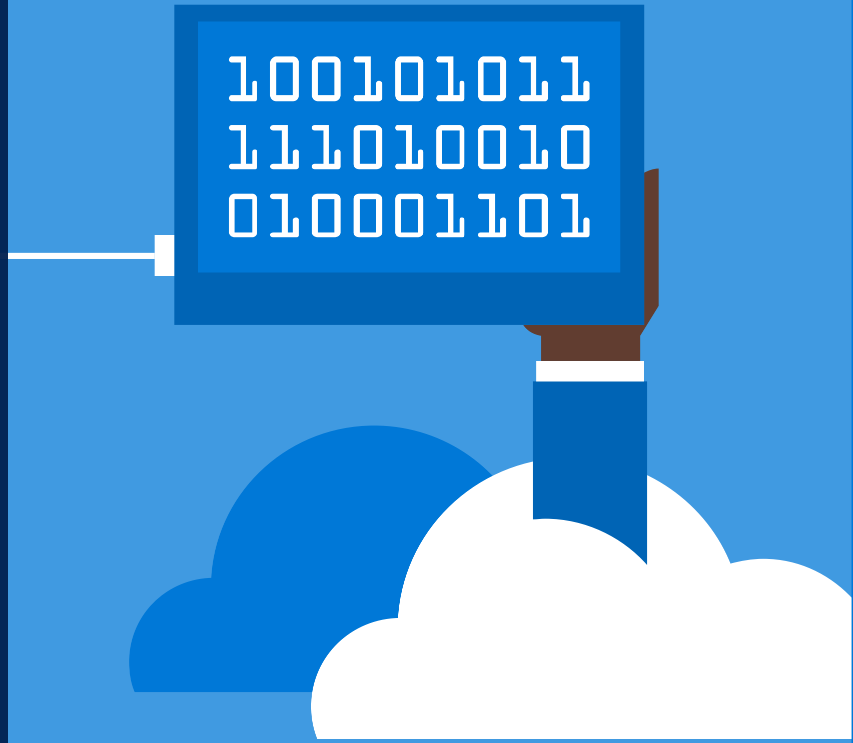
Trust and Blockchain

Mike Ormond
Microsoft
Developer Experience

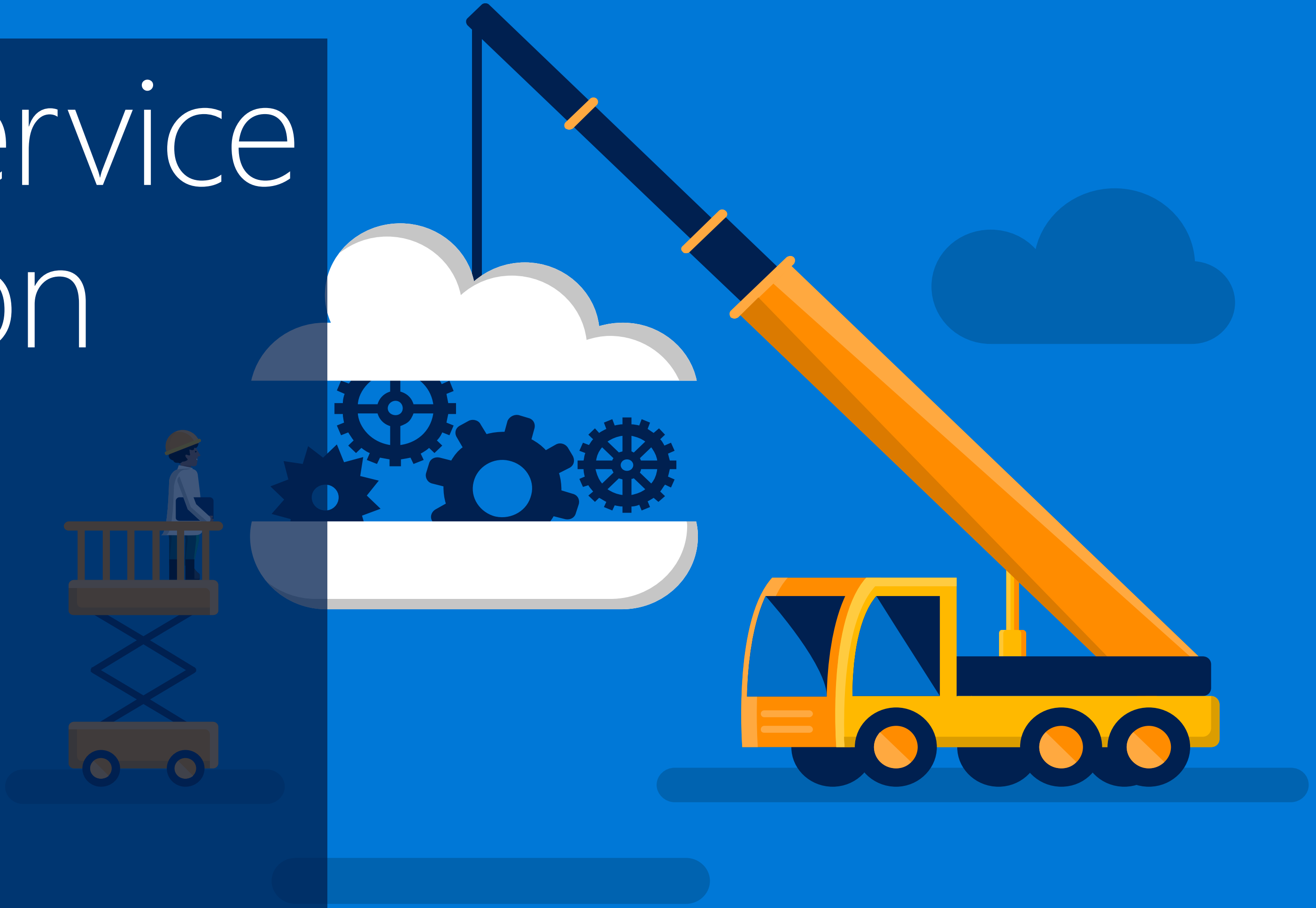
Azure Active Directory Application Security in action

Securing a multi-tier app using AAD

Ben Roscorla
Microsoft
Developer Experience



ID/PaaS - App Service EasyAuth In Action



Will Eastbury
Microsoft
Developer Experience

https://docs.microsoft.com/en-us/azure/app-service/app-service-authentication-overview

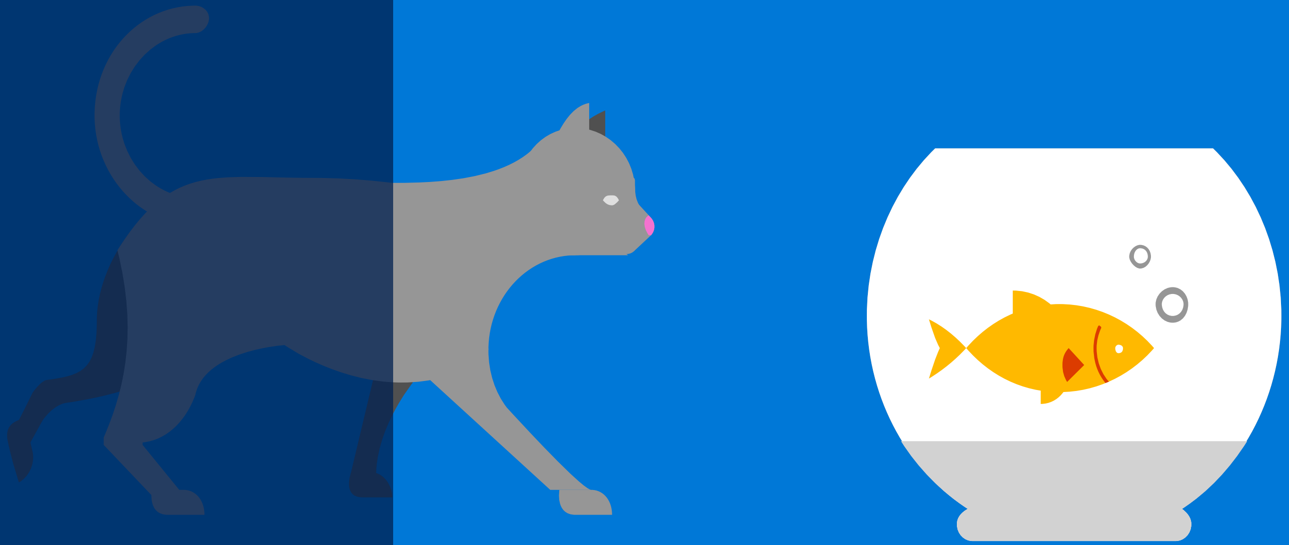
Evaluation Forms

We hope you have enjoyed the morning, we have one ask of you.

Please out your evaluation forms to give us feedback on what we did right and how we can improve the sessions next time.

Your feedback is critical to making more of these sessions a success !

Lunch



Cloud Security Principles

Phil Winstanley
Microsoft
Premier Developer

https://www.ncsc.gov.uk/blog-post/debunking-cloud-security-myths

Azure Specific Security features and practices

Will Eastbury
Microsoft
Developer Experience

Azure Compliance

The largest compliance portfolio in the industry

ISO 27001

SOC 1 Type 2

SOC 2 Type 2

PCI DSS Level 1

Cloud Controls Matrix

ISO 27018

Content Delivery and Security Association

Shared Assessments

FedRAMP JAB P-ATO

HIPAA / HITECH

FIPS 140-2

21 CFR Part 11

FERPA

DISA Level 2

CJIS

IRS 1075

ITAR-ready

Section 508 VPAT

European Union Model Clauses

EU Safe Harbor

United Kingdom G-Cloud

China Multi Layer Protection Scheme

China GB 18030

China CCCPPF

Singapore MTCS Level 3

Australian Signals Directorate

New Zealand GCIO

Japan Financial Services

ENISA IAF

<Redacted>

+ Standard deck includes some NDA only slides available internally

Replaced with some additional Ad-hoc chat from Will

"What about the things to avoid, i.e. what <u>*not*</u> to do."

General Guidance – Security and DevOps

ROTATE KEYS AND SECRETS

Most customers do not rotate their storage account keys or secret credentials.

Customers should Investigate Azure Key Vault and rotate keys in line with their usual procedures

IMPLEMENT A SECURE DEVELOPMENT LIFECYCLE PROCESS

Customers believe that managing their perimeter is enough to remain secure.

Customers should be aware that this is not enough, and develop with SDLC in mind.

UNDERSTAND YOUR ATTACKER

Some customers believe that managing their perimeter is enough to remain secure, this is not true – on-premises or in the cloud.

Customers should be aware of the threats, both internal and external.

NEVER EVER PUT SHARED SECRETS IN SOURCE CONTROL

Developers should not generally be able to see these secrets and connect to production databases .

Customers should be aware of the risk to production data from staff leaving the organisation.

CLEAR PROCESS TO IDENTIFY AND ACTION ON A BREACH

Some customers do not baseline usual activity of their system, so are unable to detect unusual activity.

Customers should consider operating as though they have been breached.

General Guidance – Subscriptions and Admin

Don't Ignore admin best-practice

Most customers should have;

- Minimum number of admin accounts in production
- Multi Factor Auth. for admins
- Enabled Azure AD accts, NOT using MSA accounts

Customers should be aware of the potential impact of a compromise of an admin account.

Don't Make devs admins on production subscriptions

We see customers with a large number of admins that were developers on their production subscription.

Customers should be aware of the risk here, that a developer could accidentally deploy into live, causing potential business impact.

Lock resources against accidental deletion

Many customers have resources in their environment that, if deleted by an attacker, could bring down the solution.

Customers should be aware of the risk of malicious deletion by an employee. Consider using Azure Resource locks as a mitigation step.

Partition subscriptions and 'classic' (non ARM) environments

Many customers are running multiple environments (Dev, Test and Production in one subscription.

Customers should be aware that ASM services do not have Role Based access in place, so can't be granularly controlled,

Panel Q&A

Will Eastbury
Ben Roscorla
Phil Winstanley

Group Discussion

Evaluation Forms

We hope you have enjoyed the day, we have one ask of you.

Please out your evaluation forms to give us feedback on what we did right and how we can improve the sessions next time.

Your feedback is critical to making more of these sessions a success !

Microsoft