Microsoft

Identity in the center of your applications

Tarek Dawoud @CyberTarek
Alex Pavlovsky

Program Managers
Microsoft Identity Division

Agenda

- Problem Statement: It is a lot more likely to expose code secrets today.
- Proposed Solutions: How to prevent developers from accidentally checking in secrets
- Best practices and discussion

Credentials in Code – Why Should You Care?


U.S. House of Representatives
Committee on Oversight and Government Reform

OVERSIGHT
& GOVERNMENT REFORM
EST 1814

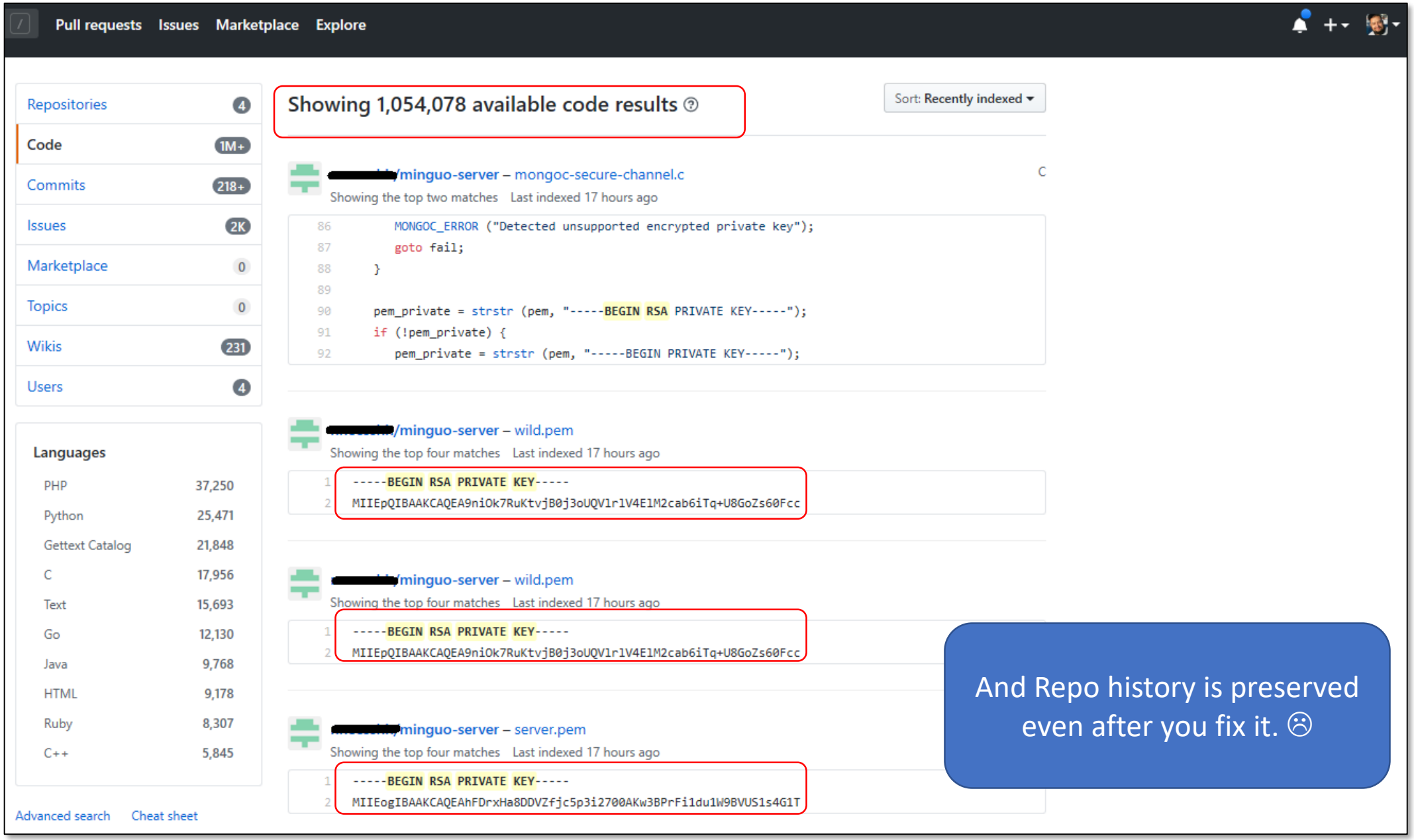
The Equifax Data Breach

Majority Staff Report
115th Congress

On May 13, 2017, attackers began a cyberattack on Equifax. The attack lasted for 76 days. The attackers dropped "web shells" (a web-based backdoor) to obtain remote control over Equifax's network. They found a file containing unencrypted credentials (usernames and passwords), enabling the attackers to access sensitive data outside of the ACIS environment. The attackers were able to use these credentials to access 48 unrelated databases.

https://web.archive.org/web/20181219201616/https:/oversight.house.gov/wp-content/uploads/2018/12/Equifax-Report.pdf

Come on! No one does that!



App Credentials – Needed to Bootstrap Apps

- Service-to-Service Calls (i.e. OAUTH2 Client Credential Grant, OBO Grant)

```
<appSettings>
     <add key="ida:ClientId" value="380e94d8-6b61-405a-bb8c-a316914ee080" />
     <add key="ida:AppKey" value="qkDwDJlDfig2Ip3euUZ" />
</appSettings>
```

- Database Connection Strings

```
<connectionStrings>
    <add name="CustomerDBConnectionString"
         connectionString="Server=tcp:contoso.database.windows.net;Database=customerPII;
         User ID=svcApp1@contoso.database.windows.net;Password=P@ssw0rd1;
         Trusted_Connection=False;Encrypt=True;" />
  </connectionStrings>
```

- Cloud Resource Config Files

```
[S3Profile-BucketA]
aws_access_key_id = b4fbdeaf-b57e-4a7a-9117-4016de5a5ed9
aws_secret_access_key = ZUue3pI2gifDlJDwDkq
```

How to stop the bleeding?

- Prevention:
  https://github.com/awslabs/git-secrets
  (prevents commits and merges that meet a secret pattern)
- Scanning:
  https://azure.microsoft.com/en-us/blog/managing-azure-secrets-on-github-repositories/
  (scans existing repo's for secret patterns)

Providing a way forward for your developers

- Introducing the concept of Managed Secrets by Cloud Services to them.
- Secret configuration is done in the cloud service provider and stored there.
- The Application requests its secret at run time. The secrets are bound to a machine/resource type within your tenancy.
- The secrets are used at run time and never persisted on the machine.
- No certificate to be handled by a trusted Ops person.
- Remember: Developers are like a river… They are never blocked. ☺

Credential Management examples from Azure

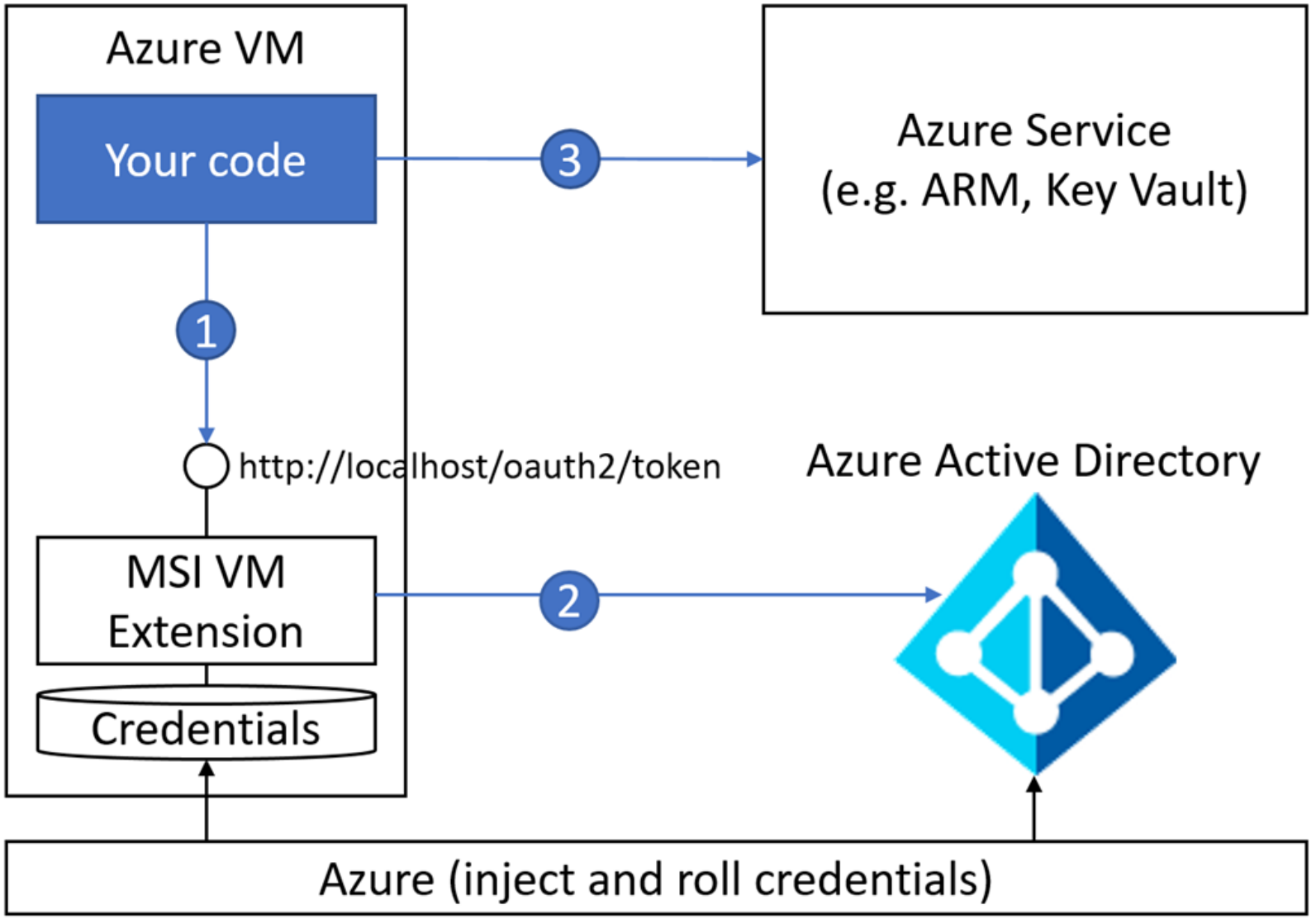
Dealing with secrets in the code

Your friends

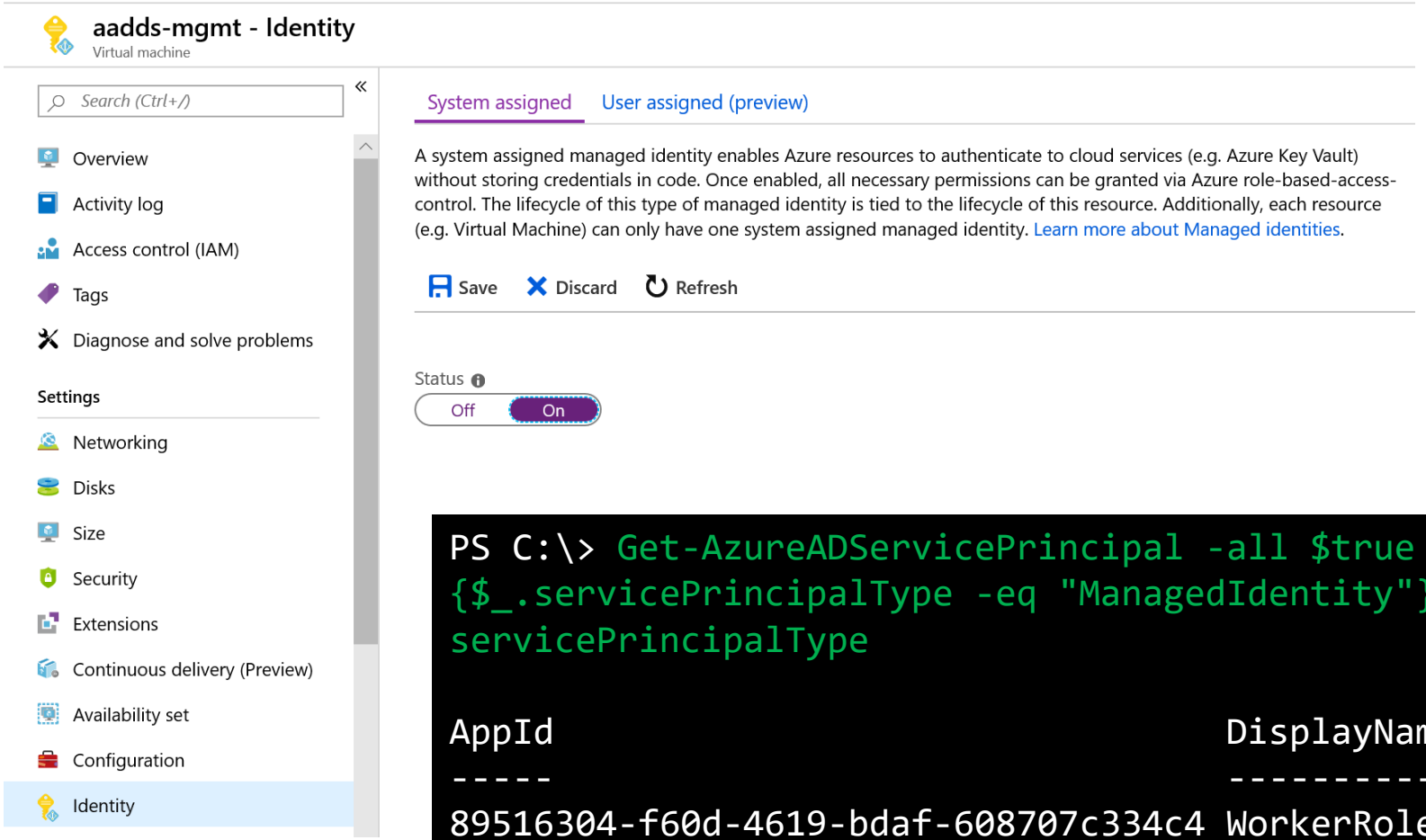
Managed Identities for
Azure Resources
(AWS has Secrets Manager)

Certificate Credentials for
Service Principals

Microsoft Azure Managed Identities



Microsoft Azure Managed Identities



Sample: https://azure.microsoft.com/en-us/resources/samples/app-service-msi-keyvault-dotnet/
(Azure App Service getting a SQL Connection String at runtime from Azure Key Vault)

```
PS C:\> Get-AzureADServicePrincipal -all $true | Where-Object
{$_.servicePrincipalType -eq "ManagedIdentity"} | select AppId, DisplayName,
servicePrincipalType

AppId                                DisplayName           ServicePrincipalType
-----                                -----------           --------------------
89516304-f60d-4619-bdaf-608707c334c4 WorkerRoleAppProcess1 ManagedIdentity
0488aee7-14ce-4e84-a986-ce73e69de2e0 user_MSI_1            ManagedIdentity
d4616abe-e4ed-40e3-825b-976afaea9509 VM-Linux-Ubuntu2      ManagedIdentity
```

End-To-End Scenario Example

- Setup
  - Linux VM in Azure IaaS (VM1)
  - Service running on the VM needs to run a script accessing an API (API1)
  - You enable a Managed Identity for VM1
  - You protect API1 with OAUTH2 Authorization (i.e. register API1 in Azure AD)
  - You grant VM1 managed identity permissions to call API1 (OAUTH Permissions)
- Runtime
  - Service on VM1 makes a local HTTP GET request to http://169.254.169.254/metadata/identity/oauth2/token
  - Returned is an Azure AD access_token with scopes for API1
  - Service on VM1 calls API1 with the access_token for API1
- There is ZERO credentials required to call API1. Not in code, not in Vault.

Managed Identity Assignment Type

- System assigned – single managed identity per resource
- User assigned – one managed identity can be used by multiple services
  - If multiple services access a single resource, makes it easier to manage permissions for the resource

Service Principal with Private Key sample

- If you can’t use a Managed Secret from your cloud service provider, try and use a private certificate instead of a password.

  Example: https://docs.microsoft.com/en-us/azure/active-directory/reports-monitoring/tutorial-access-api-with-certificates
- In Azure AD, Service Principals are application identities. They reduce the attack surface by only working in programmatic interfaces.

Summary

- The way applications manage their secrets has shifted and modernized. Make sure your developers and your security organization is aware of the new way to do things.
- Develop a practice with your Development/Architecture teams to instill the tools to put your organization in a good place:
  - Prevention tools
  - Scanning tools
  - An organizational guide on Secret Management showing common examples for the platforms your org commonly uses.
- Modernize your older apps.

Q&A and Discussion