Securing Your Cloud Applications with Identity and Private Networking Best Practices

Preventing the next “blizzard”!

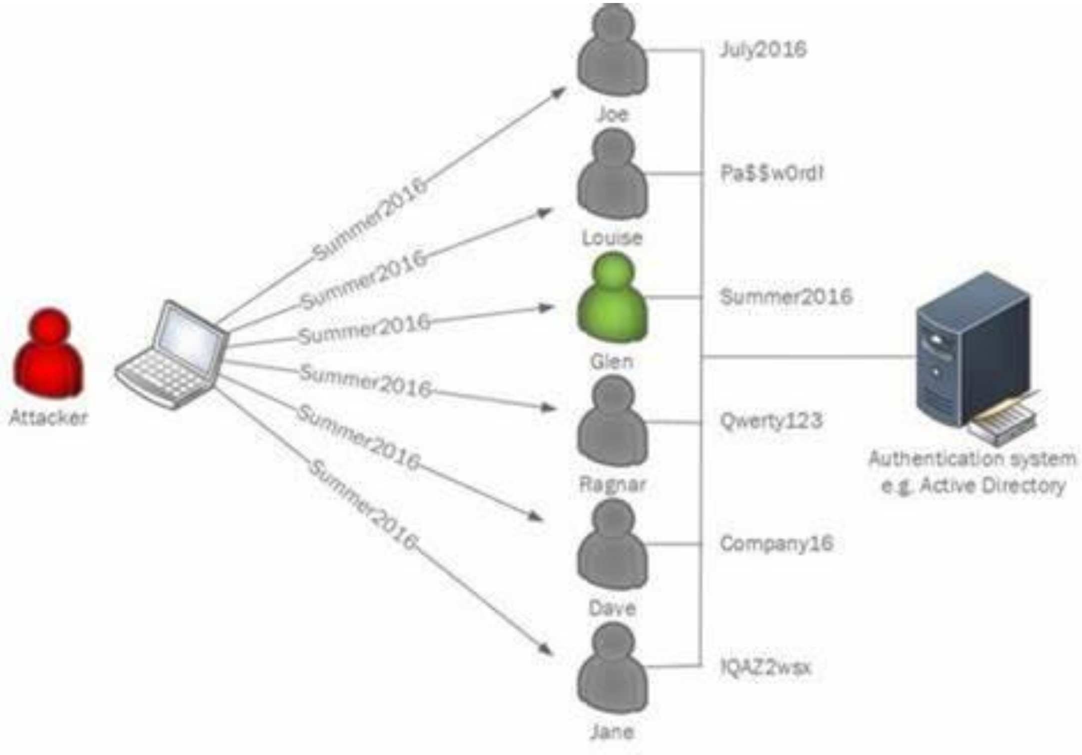
Paul Yuknewicz (@paulyuki99)

In Today’s Session

- Attack examples
- Basic prevention in infrastructure
- Passwordless deep dive using managed identity
- Network isolation deep dive
- Tips n tricks
- Resources

Midnight Blizzard Attack

- Legacy & dev test servers, leverage passwords and service principals
- Buying/finding account names online
- Password spraying
- OAuth app and admin misuse
- Reading emails with more passwords <repeat>



Midnight Blizzard: Guidance for responders on nation-state attack | Microsoft Security Blog

Protecting your organization against password spray attacks | Microsoft Security Blog

Basic Strategies – Part 0 - Infrastructure

- Use modern, PATCHED servers!
- Use latest secure SDKs, runtimes, etc
  - PaaS/managed services help
  - Secure supply chain / registries
- Install only what you need
- Limit access to absolute minimum
- Separate servers by concern, e.g. dev, test, prod, docs, and *never mix workloads/assets (e.g. prod workload users on test, test scripts on prod)

I DON'T ALWAYS THINK ABOUT CYBERSECURITY...
BUT WHEN I DO, IT'S USUALLY TOO LATE.
imgflip.com

Basic Strategies – Part 1 - Passwordless

- Go Passwordless!
  - Identity based connections: Entra ID, **Managed Identity**, OIDC, more..
  - PassKey & FIDO keys
  - Turn off passwords
  - Delete service principals
  - Delete tokens/secrets from disk, code, env vars, everywhere..
- Use MFA!
- Never accept unrecognized request
- If you must use password/token/secret, use Key Vault

Basic strategies – Part 2 – Access Control

- This is about RBAC or roles based access control
- Restrict access levels to roles, and only grant the minimum
- Review all access lists now, prune, refresh regularly
- Avoid using any Full Control, Full Admin type roles
- Beware of service principals – they still store the secret (identity and/or OIDC with managed identity always preferred)

Basic Strategies – Part 3 – Network Isolation

- This is about fencing networks from threats
- Apps & dependencies belong to VNET
- App outbound traffic to dependencies uses private endpoint, optionally NAT
- App inbound traffic uses an option to limit traffic (auth, vnet, allow lists)

new Azure Functions quick start samples

These samples are secure & scalable, they

- demonstrate best practices in creating secure apps
- include VNET integration
- use Flex Consumption
- use the Azure Developer CLI for easy deployment

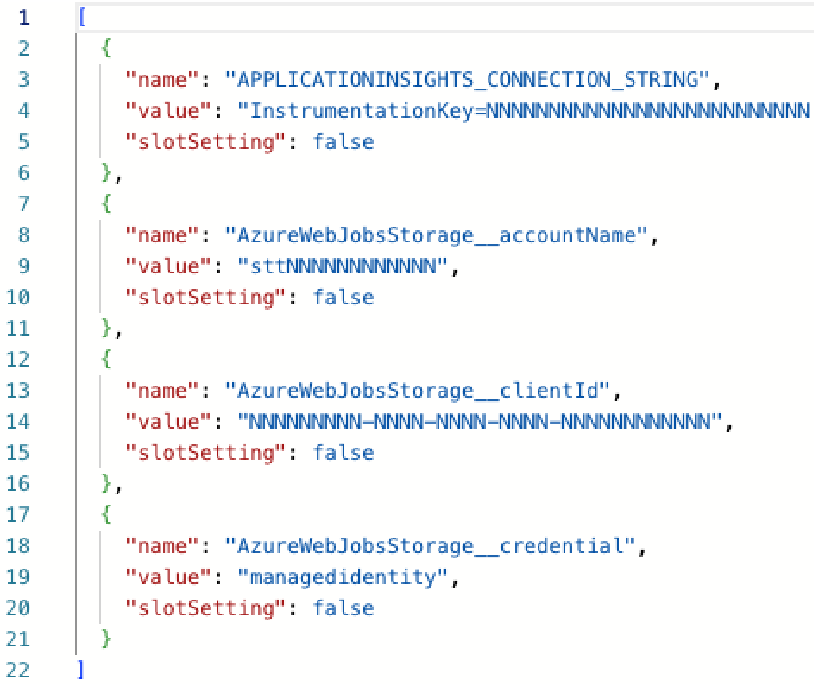
https://aka.ms/functions-secure-samples

Identity Deeper Dive

- Managed identity
  - System assigned or SAMI (quick)
  - User assigned or UAMI (durable, recommended for prod)
- Default identity connection uses SAMI
- **ClientId**, **Credential**, & **URI/name** (3 settings) always needed for UAMI
- “conn__property” syntax for SDKs and bindings
- IAM (RBAC) roles are then required to grant least access
  - Consider your app’s managed identity
  - Consider your own login identity

[Example – identity based connection: Azure Service Bus trigger for Azure Functions | Microsoft Learn]

- Top places to use identity
  - Deployment package load (functions.zip)
  - AzureWebJobsStorage state management
  - SDKs/Triggers/bindings connections



```
[
  {
    "name": "APPLICATIONINSIGHTS_CONNECTION_STRING",
    "value": "InstrumentationKey=NNNNNNNNNNNNNNNNNNNNNNNNNNN
    "slotSetting": false
  },
  {
    "name": "AzureWebJobsStorage__accountName",
    "value": "sttNNNNNNNNNNNN",
    "slotSetting": false
  },
  {
    "name": "AzureWebJobsStorage__clientId",
    "value": "NNNNNNNNN-NNNN-NNNN-NNNN-NNNNNNNNNNNN",
    "slotSetting": false
  },
  {
    "name": "AzureWebJobsStorage__credential",
    "value": "managedidentity",
    "slotSetting": false
  }
]
```

Want to Follow Along?

https://aka.ms/functions-secure-samples

```
azd init --template functions-quickstart-dotnet-azd

azd up
```

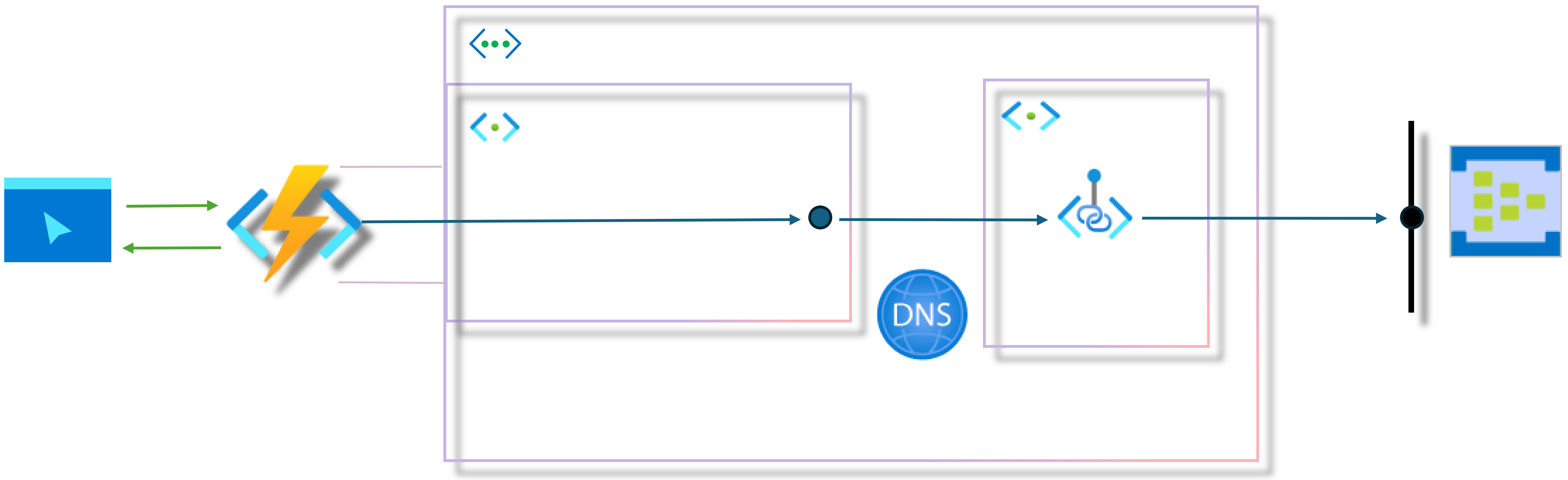
Demo

Implementing identity and network isolation (new app)
Implementing … (existing app)

Network Isolation Overview

- Inbound networking controls access to your app
  - Public network access can be either enabled or disabled
  - IP access restrictions
  - Service endpoints
  - Private endpoints

- Outbound networking controls how connections are made
  - Virtual network integration
  - NSG Rules
  - UDRs
    NAT Gateway


DNS

Network Isolation Deeper Dive

- App inbound traffic uses an option:
  - VNET (most secure, but limiting public access)
  - Trusted IPs
  - Auth method (e.g. Easy Auth and authorization list)
  - Recommend turning off public access by default, opt in if really desired
  - Use Azure Front Door and Defender to protect and managed inbound traffic

- App outbound
  - Joins vnet
  - Private service endpoints
  - NAT
  - Service tag to identity customer/tenant

Demo

Network isolation part 2

Resources

Create functions in Azure using the Azure Developer CLI | Microsoft Learn – secure by design quickstarts

Security - Azure App Service | Microsoft Learn – app service

Securing Azure Functions | Microsoft Learn – functions

@paulyuki99 when all else fails