MAKING A MESH OF THE INFRASTRUCTURE

PETE BROWN

NETWORK ENGINEER, AUTOZONE

@JPBROWN1

Pete has supported various functional areas of infrastructure for around 20 years. He is a proponent of democratizing access to infrastructure data in order to promote learning, facilitate better decision making and reduce reliance on tribal knowledge.

AGENDA

- The Challenge
- Infrastructure Sources
- Traditional Approaches
- Mesh Integration
- Demo
- Q&A

THE CHALLENGE

- We are often asked to collect and analyze data from disparate infrastructure source types. These efforts are hindered by constant technological and environmental changes.

- How can we mitigate the impact of these changes to reduce time spent on development and maintenance?

THE SOLUTION

Apply data mesh* principles! Infrastructure data sources should be:

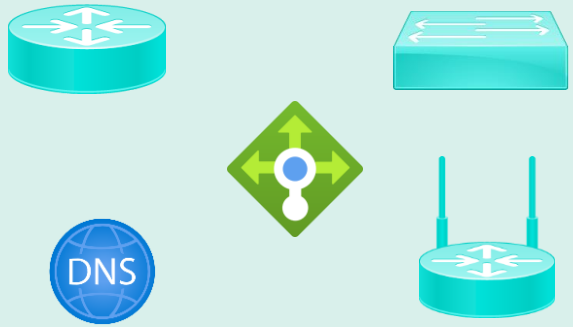
- Discoverable
- Addressable
- Trustworthy
- Self-Describing
- Interoperable
- Secure

https://martinfowler.com/articles/data-monolith-to-mesh.html

COMMON INFRASTRUCTURE SOURCES

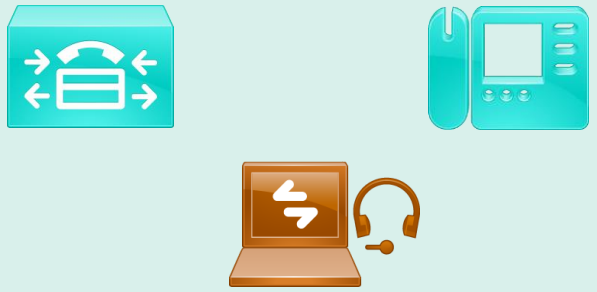
Directory Services

Network

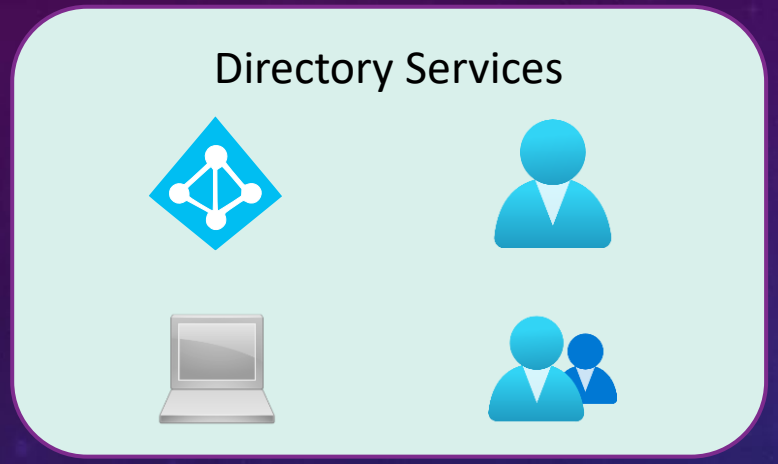


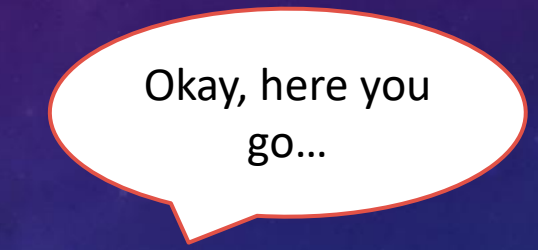
InfoSec

Telephony

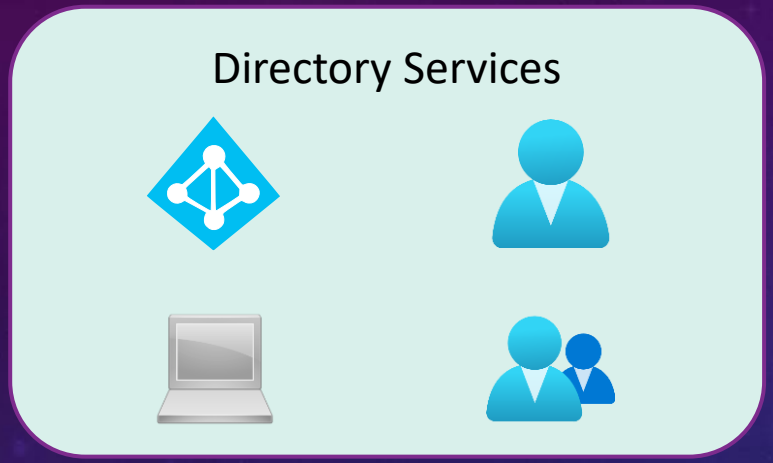


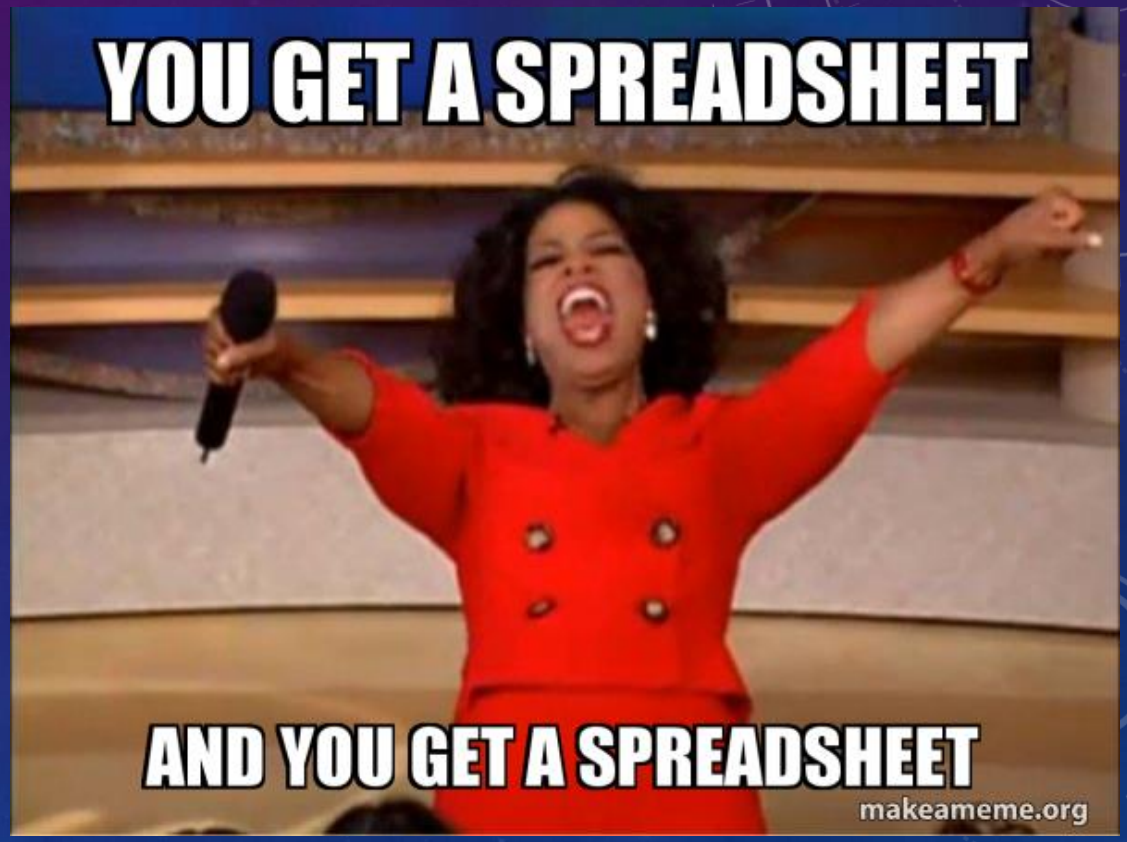
COMMON INFRASTRUCTURE SOURCES



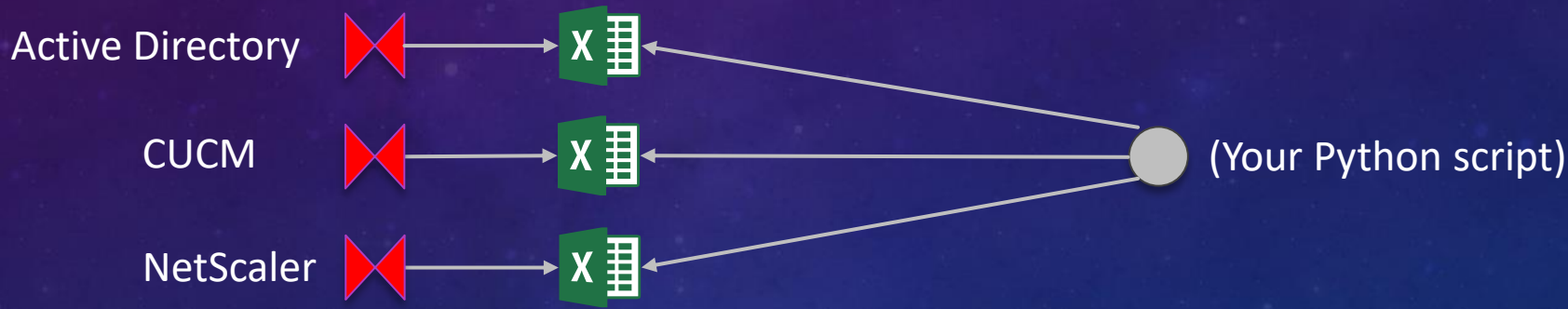


COMMON INFRASTRUCTURE SOURCES





ACCESSING A SOURCE – NO API



Consumer

Native

SAMPLE SOURCES

Source Type	Protocol	Data Format	Spec Format	Spec Source	Client Language
Meraki	REST	XML/JSON	OpenAPI	Application	any
CUCM – AXL/RIS	SOAP	XML	PDF/HTML	Application	any
CUCM - JTAPI	JTAPI	binary	PDF/HTML	Application	Java
CUC - CUPI	REST	XML	PDF/HTML	Internet	any
NetScaler	REST	JSON	PDF/HTML	Internet	any
Active Directory	LDAP	binary	PDF/HTML	Internet	any

ACCESSING A SOURCE – DIRECT API

Study Integration Interface(s)

Could be an API or native protocol (SQL, LDAP, etc)

Implement Interface Libraries

Acquire or create language specific interface libraries

Discover Instances

What instances are running in the environment? What versions are they running?

Acquire Credentials & Roles

Work with service owners to acquire credentials & roles for each instance

Address Routing & Firewall Issues

Work with network and security groups to resolve any communication issues

ACCESSING A SOURCE – DIRECT API

Consumer

Native

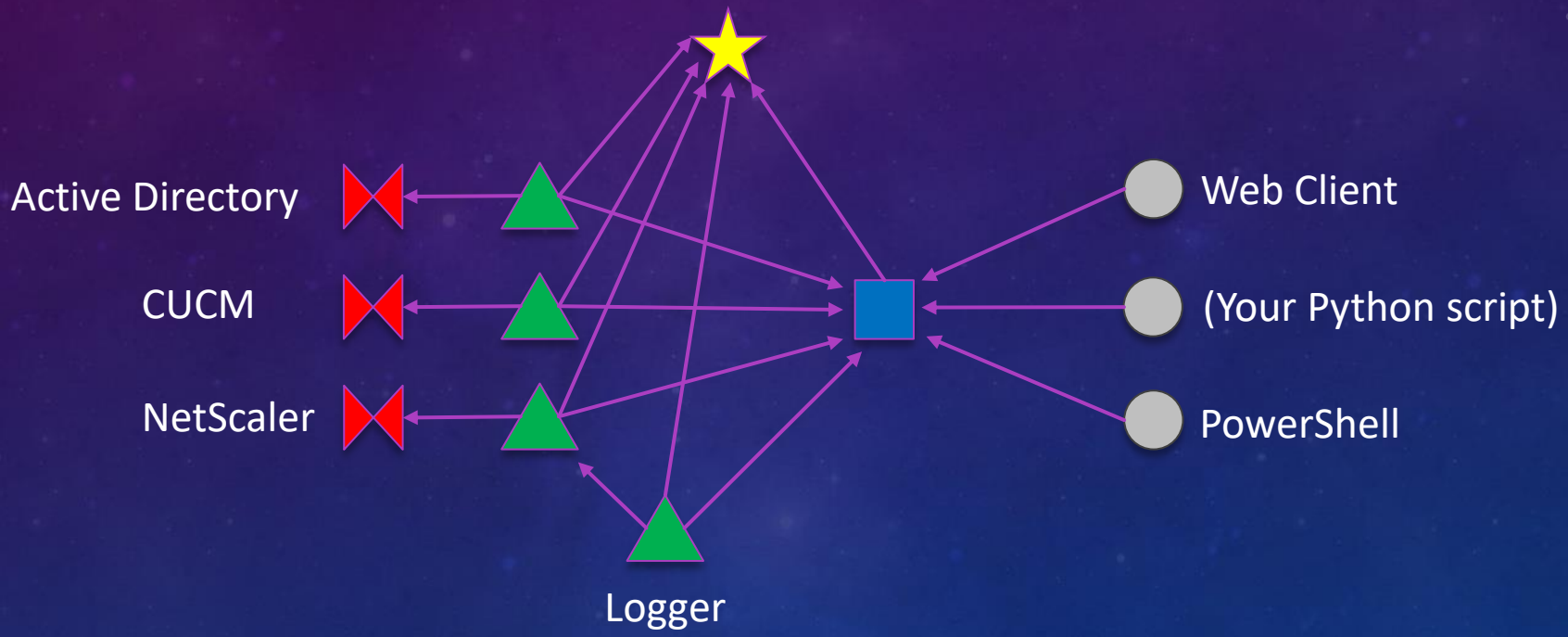
DRP – DECLARATIVE RESOURCE PROTOCOL

- A JSON based WebSocket subprotocol for declaring and consuming resources
- Provides a relatively easy way to create a service mesh for the infrastructure
- Allows consumers to focus on data analysis functions by reducing time spent on discovery and connectivity
- The Registry and Broker nodes are meant to run as part of the network (containers on core switches)

DRP – DECLARATIVE RESOURCE PROTOCOL

- Sources are declared, not discovered
- Sources use a common format for declarations
- Consumers use a single logical endpoint to access all sources
- Consumers use a standardized RPC & pub/sub mechanism to access all sources
- Promotes an integrate once, use many approach

ACCESSING A SOURCE - MESH



ACCESSING A SOURCE - MESH

1. ~~Study Integration Interface(s)~~

   ~~Could be an API or native protocol (SQL, LDAP, etc)~~

2. ~~Implement Interface Libraries~~

   ~~Acquire or create language specific interface libraries~~

3. ~~Discover Instances~~

   ~~What instances are running in the environment? What versions are they running?~~

4. Acquire ~~Credentials &~~ Roles

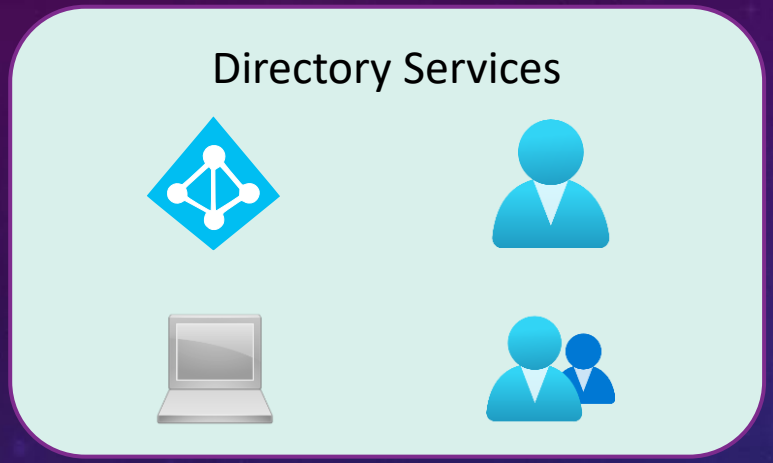
   Work with service owners to acquire ~~credentials &~~ roles for each instance

5. ~~Address Routing & Firewall Issues~~

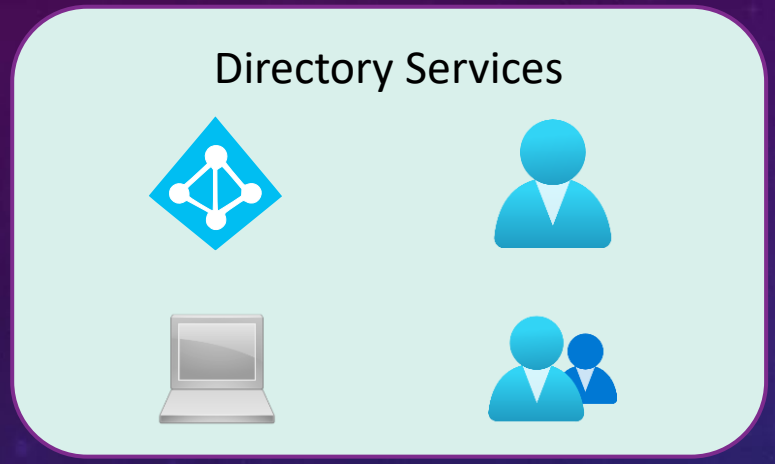
   ~~Work with network and security groups to resolve any communication issues~~

COMMON INFRASTRUCTURE SOURCES

Directory Services

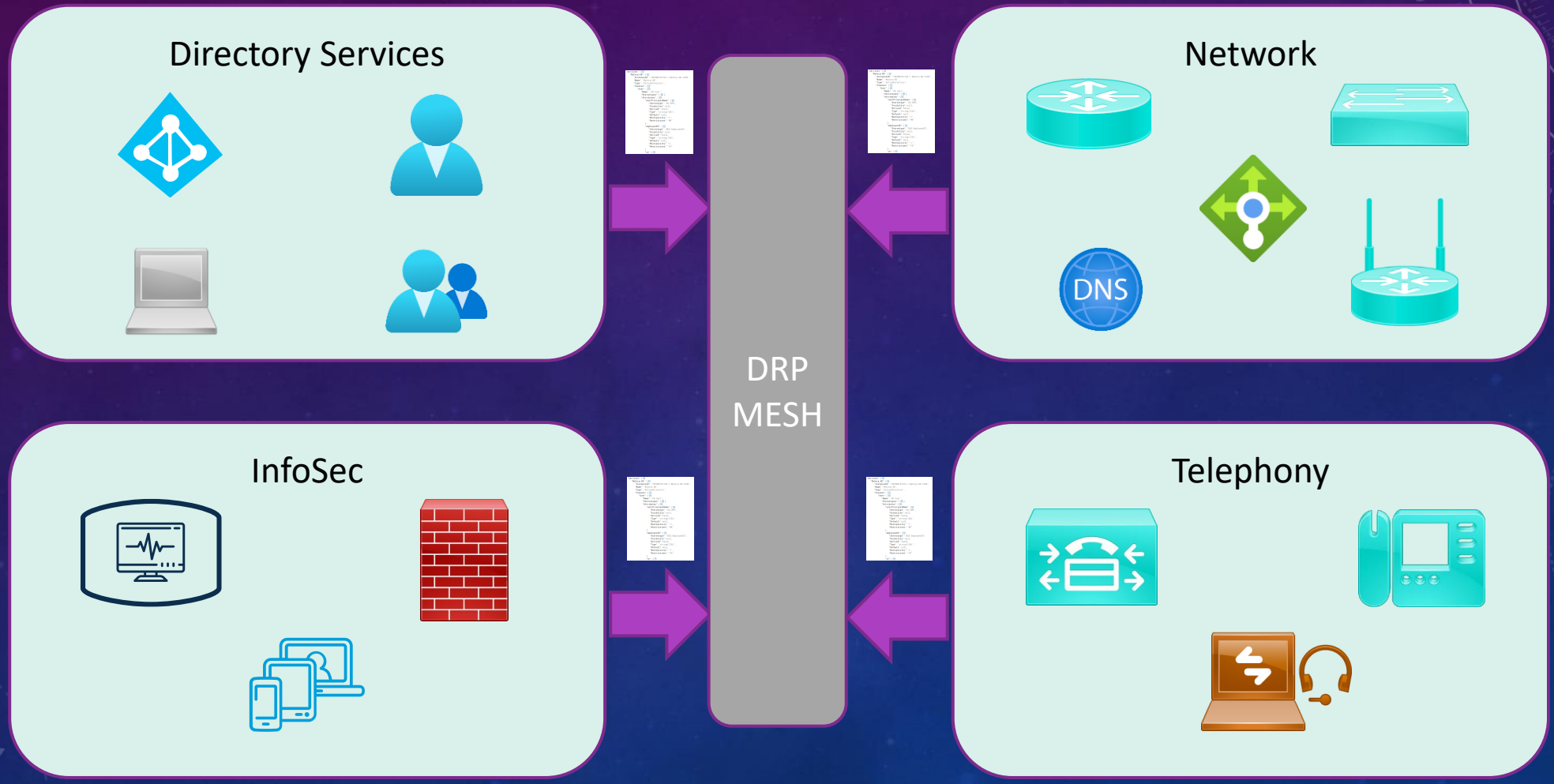


COMMON INFRASTRUCTURE SOURCES



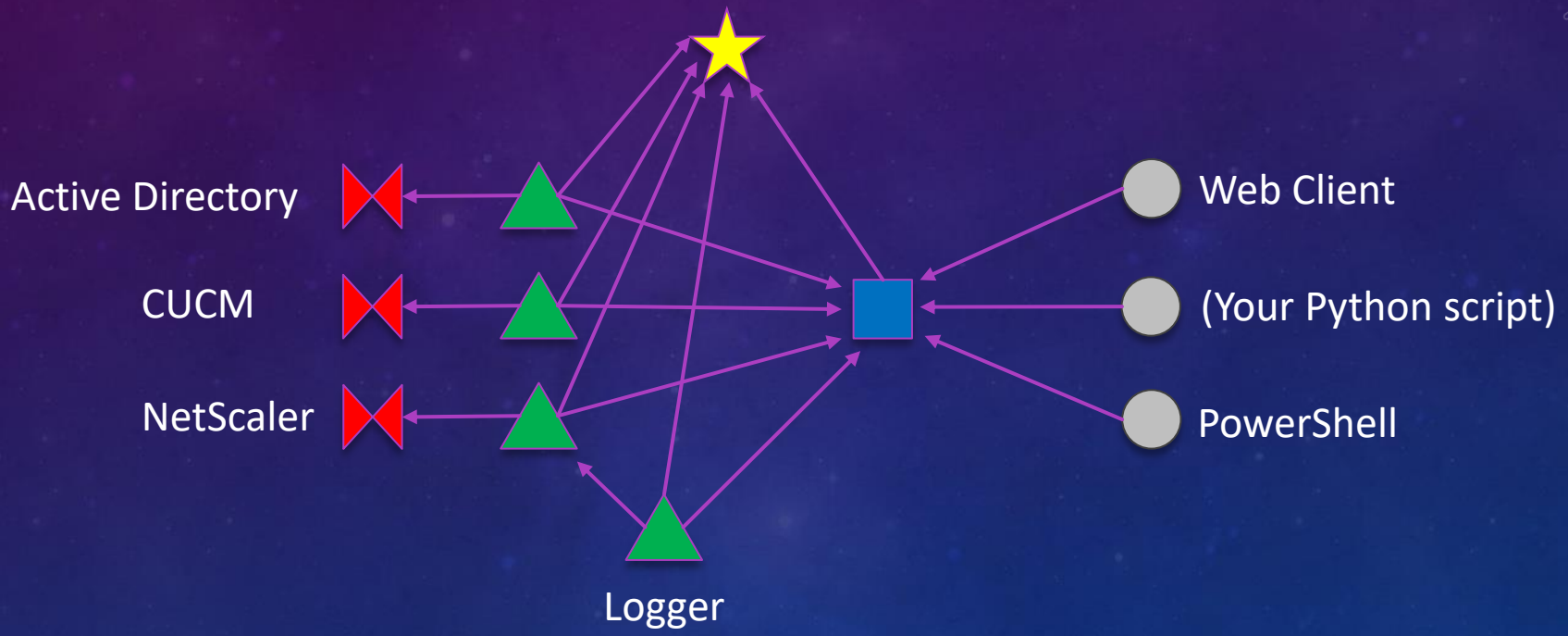

```
"Services":{
   "MyCorp-AD":{
      "InstanceID":"70f40b7671b5-1-MyCorp-AD-7240",
      "Name":"MyCorp-AD",
      "Type":"ActiveDirectory",
      "Classes":{
         "User":{
            "Name":"AD.User",
            "Stereotypes":[ ],
            "Attributes":{
               "userPrincipalName":{
                  "Stereotype":"AD.UPN",
                  "Visibility":null,
                  "Derived":false,
                  "Type":"string(128)",
                  "Default":null,
                  "Multiplicity":"1",
                  "Restrictions":"MK"
               },
               "employeeID":{
                  "Stereotype":"BUS.EmployeeID",
                  "Visibility":null,
                  "Derived":false,
                  "Type":"string(128)",
                  "Default":null,
                  "Multiplicity":"1",
                  "Restrictions":"FK"
               },
               "sn":{
```

COMMON INFRASTRUCTURE SOURCES

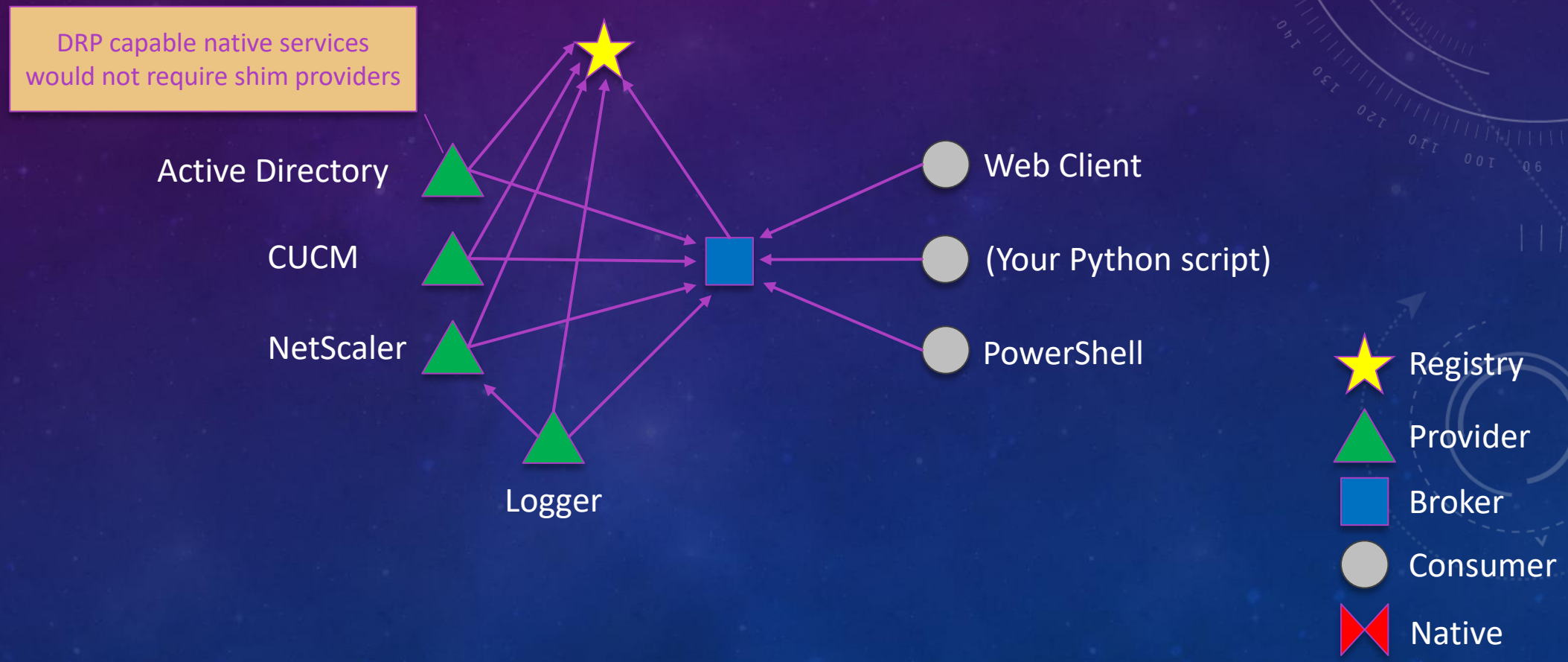


DEMO TIME

DRP MESH COMPONENTS



DRP MESH COMPONENTS - FUTURE



IS IT FOR YOU?

Pros

- Simplifies source access
- Promotes integrate once, use many approach
- Promotes learning
- Allows the use of non-listening services
- Can inject services into insecure zones
- Potentially consolidates access to all aspects of an object
- Reduces perception of “black boxes”
- Can enhance HA/DR capabilities
- Optional caching mechanism
- May lead to using graph queries with disparate infrastructure sources

Cons

- Loss of resolution in source system logs (single API user) for pre-existing sources
- Elimination of shortest route between consumer and source
- Dynamic nature of connections may cause confusion when troubleshooting
- Small development team (1)

Q & A

Project used in demo:

https://github.com/adhdtech/DRP