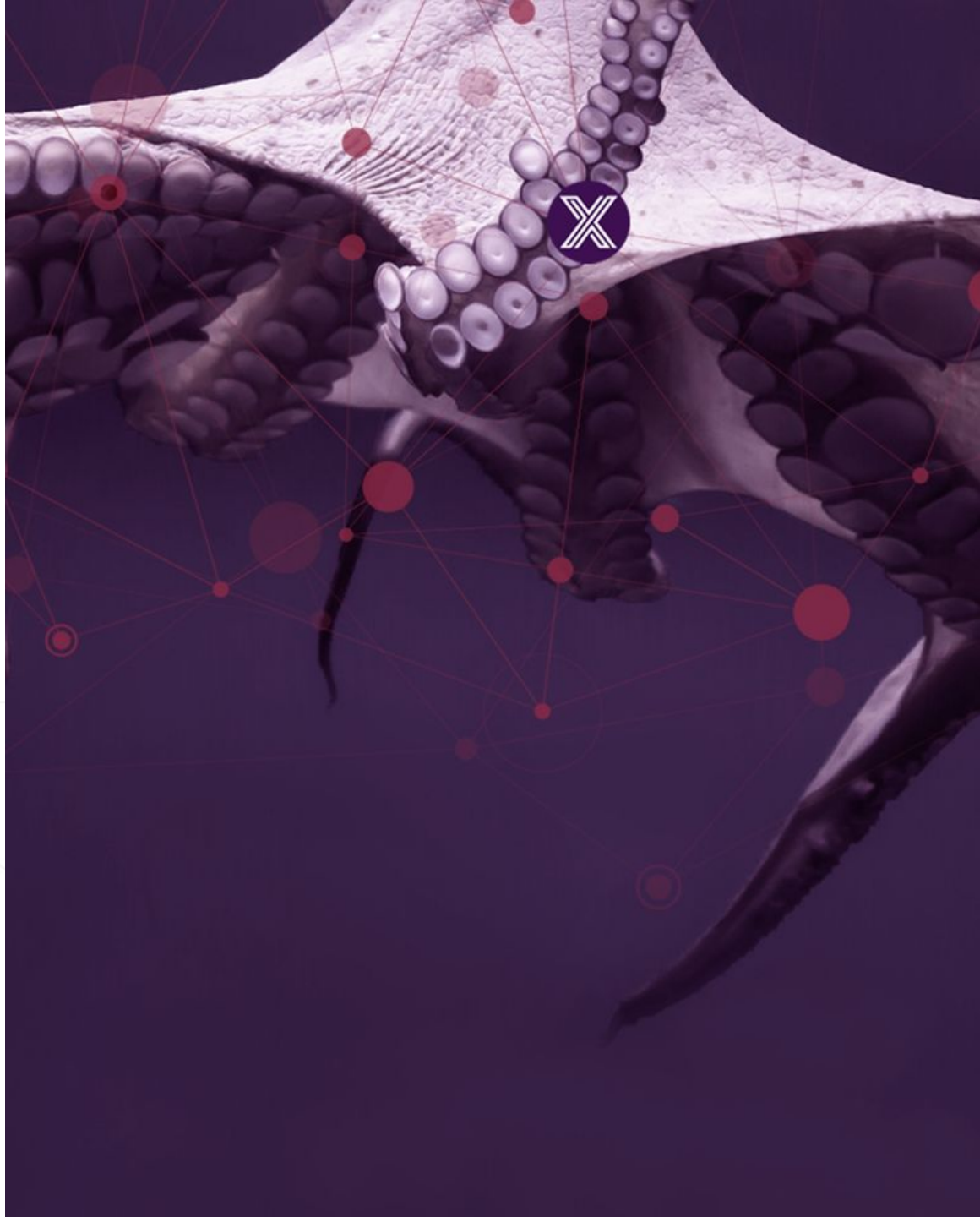
EDGE X FOUNDRY™

EdgeX Microservice Authentication

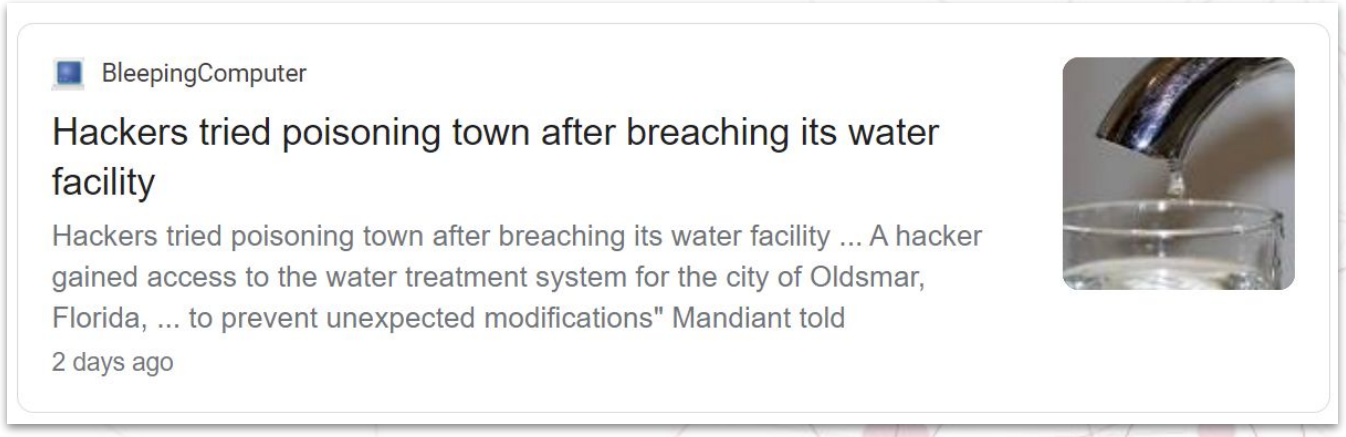


February 16, 2021

Architect's Meeting

Background

- Question of authentication for EdgeX v2 API's
  - Timely discussion -->



- What is at risk? Threat model

Threat Modeling - Assets

- IoT data
  - Read data from attached devices
  - Snoop on AI inferencing outputs
- IoT control
  - Issue device control command to control actuators in the real-world
- System availability
  - Issue transactions that cause the system to crash

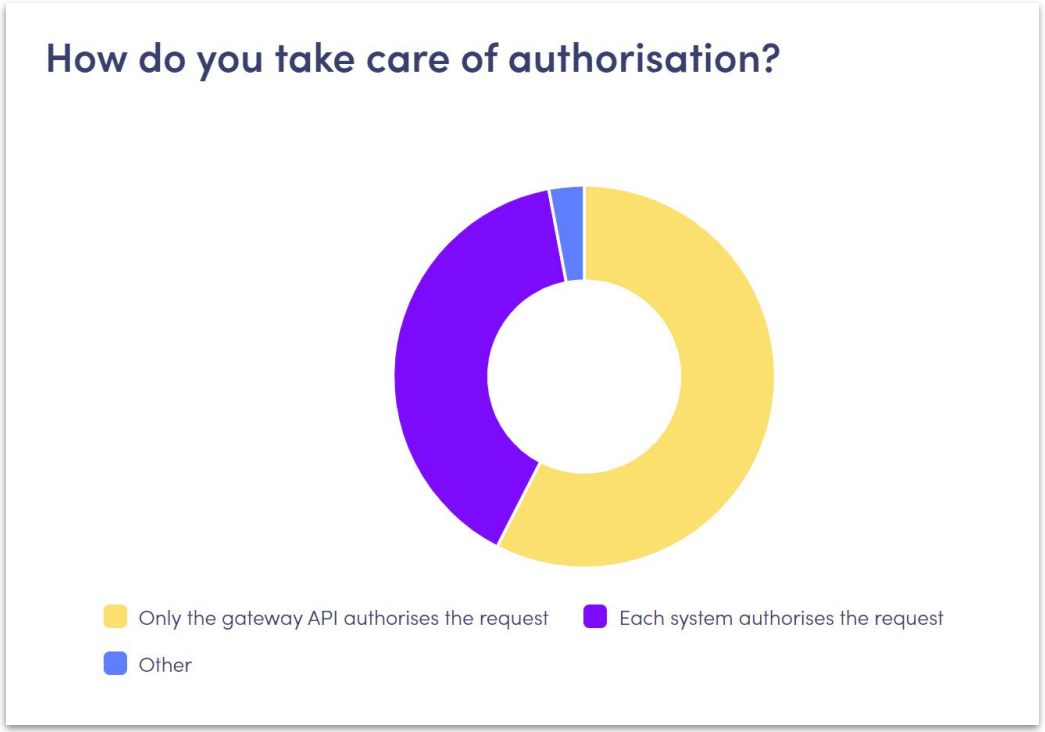
Threat Modeling - Adversaries

- Root-level adversary (or OS, or firmware, or hardware)
  - Can't protect against it - not worth discussing
- Network adversary (attacker coming in from network)
  - Single host: Protect via API gateway and localhost bindings (snaps) or docker bridge networks and being careful with IP routing
  - Distributed EdgeX: Unsafe without 3rd party solution
- **User mode adversary**
  - **Any non-root user-level process running on the host (malware, USB auto-run, Javascript running in a web browser, TeamViewer with weak password, etc.)**

Threat Modeling - Threats

- Malicious user level processes are able to issue arbitrary transactions against EdgeX microservices to collect real-world data or cause actions in the real world.
  - Docker networking offers no protection from compromised host. Compromised microservices also can issue arbitrary transactions.
  - Snaps share host networking stack; not possible to distinguish between EdgeX services vs other processes running on the host.

What are other people doing?



https://tsh.io/state-of-microservices/#varia

- 40% of respondents to survey authorize requests at the microservice level

(General survey; would hope that this number would be higher for IoT scenarios.)

Recommendations

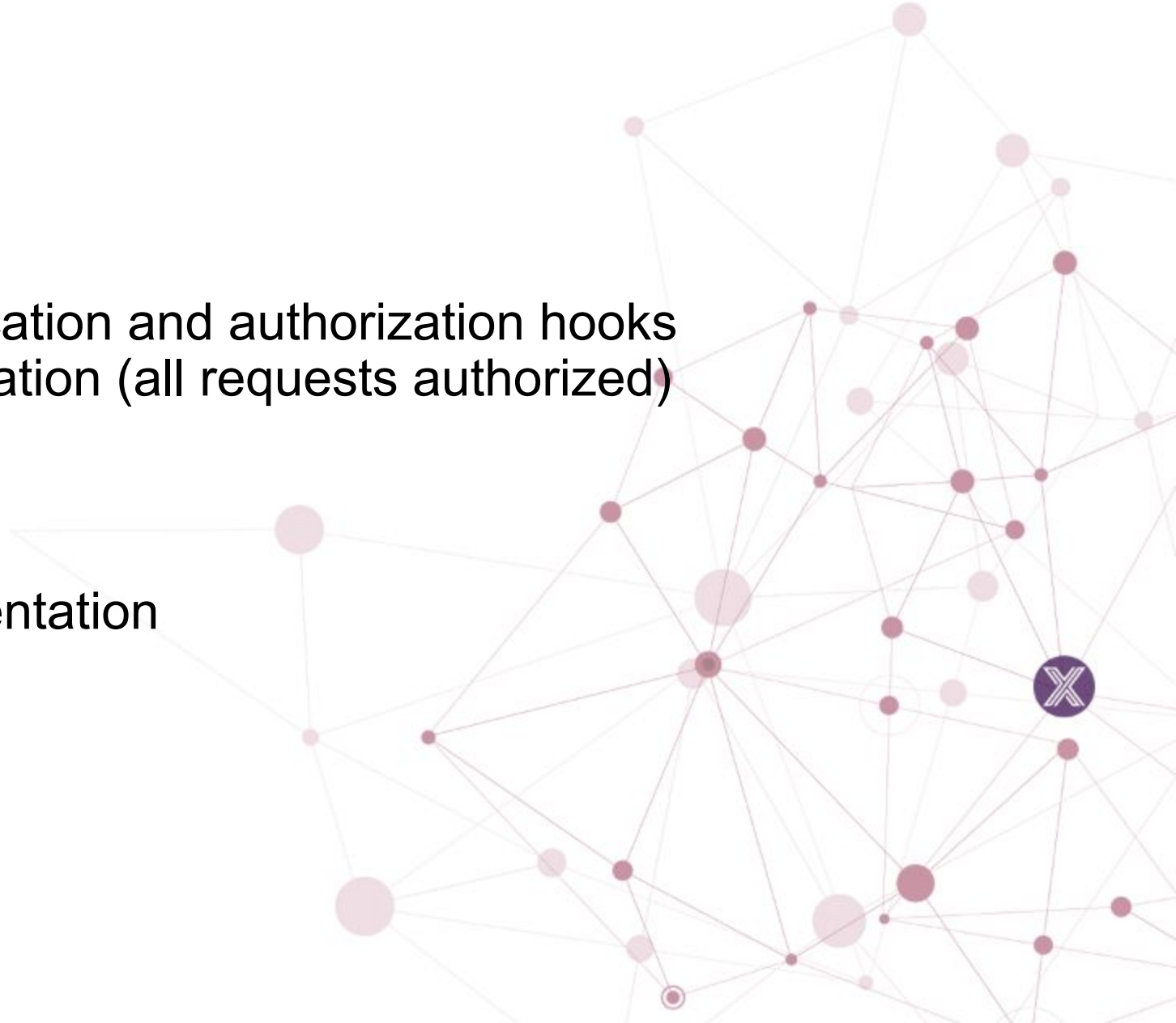
- Enable token-based microservice authentication
  - Use OIDC-compliant Vault Identity secrets engine
    - Vault will share a public key that will validate tokens that it issues
    - Sending microservices use their existing secret store tokens to request OIDC-compliant identity tokens from Vault
    - Receiving microservices use the published public key to validate the JWT
- Define hook in V2 API for implementing authentication and authorization of incoming requests
- Write and approve an ADR
  - Decide how this feature will interact with API gateway
  - Decide granularity of permission enforcement

Alternatives

- Service meshes
  - Would be the “go-to” if EdgeX was a Kubernetes-only framework
  - No easy way to use this technology in Docker and Snaps
- Kong mediates everything
  - Doesn’t protect from the the adversary we are worried about
- Make it “pluggable” instead
  - Write sender and receiver hooks to work with any OIDC-compliant identity provider. Users expected to bring their own IDP.
  - Do above in lieu of Vault-based IDP implementation.

Timeline

- Ireland
  - Implementation too big
  - Can define API v2 authentication and authorization hooks
  - Can provide null implementation (all requests authorized)
  - Can start the ADR
- Jakarta or later
  - Approve the ADR
  - Start and finish the implementation

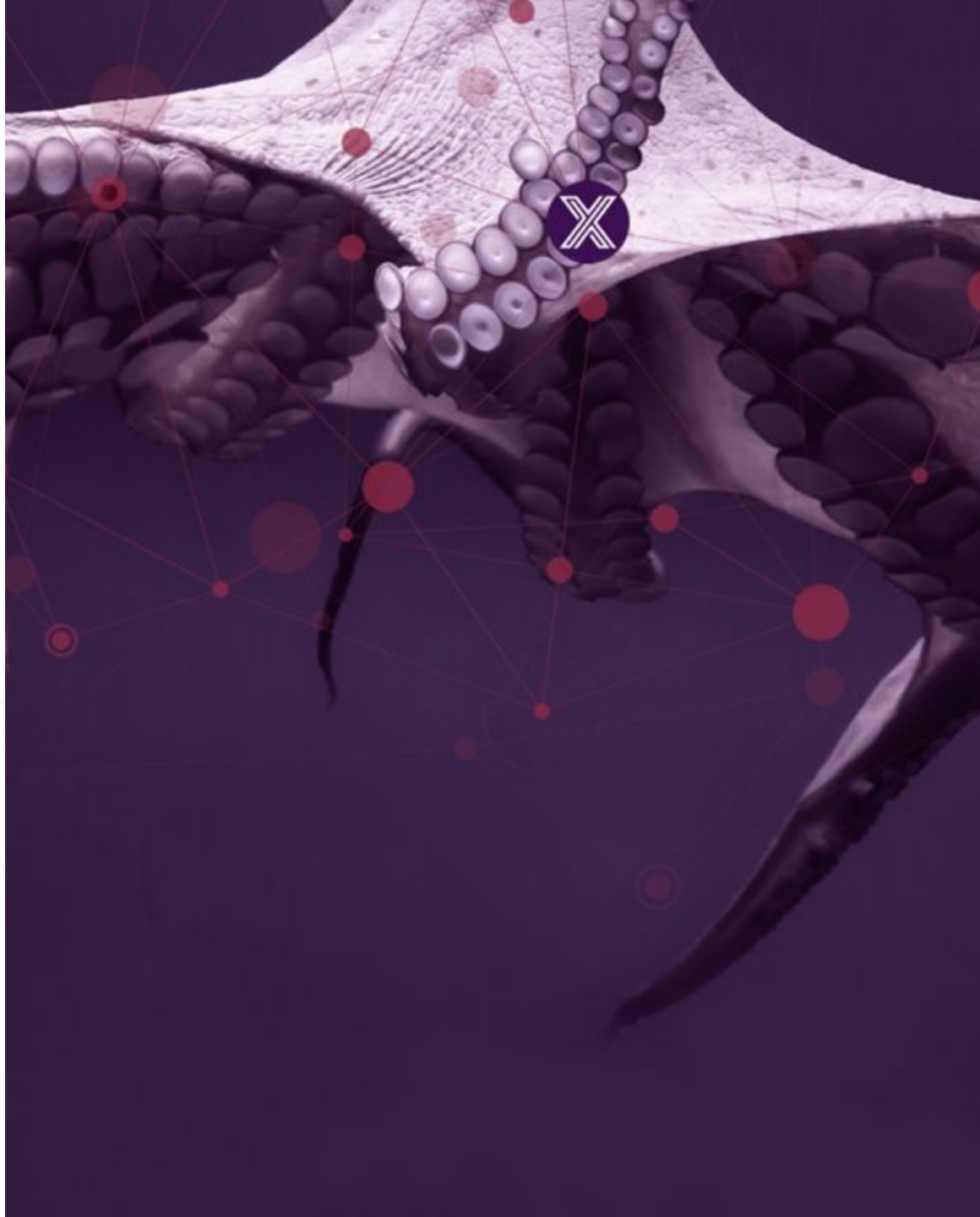


Notes 2/16/2021

- Rodney asserts there is use case for federated identity
- Rodney has info on bridging security domains (e.g. bacnet to http) that play into this discussion (would prefer not to bridge)
  - JimW focus on service-to-service first
- JimW wants to know if can wrap Vault so that it will work with a pluggable solution (also, why not mTLS listed in alternatives?)
- Clarify: this proposal is for service-to-service communication
  - What about pass-through authentication? Simplest is pass-through of token and mitigates confused deputy / escalation attacks.
- Next step: research the above and start an ADR

EDGE X FOUNDRY™

Thank You



edgexfoundry.org | @edgexfoundry