RSAConference2022
San Francisco & Digital | June 6 – 9

TRANSFORM

SESSION ID: **ZT-R03**

Searching for the Grail: Zero Trust Cryptographic Keys & Services

Karen Reinhardt

Principal Engineer, Encryption & PKI
The Home Depot
@Farrside42

Disclaimer

Presentations are intended for educational purposes only and do not replace independent professional judgment. Statements of fact and opinions expressed are those of the presenters individually and, unless expressly stated to the contrary, are not the opinion or position of RSA Conference LLC or any other co-sponsors. RSA Conference does not endorse or approve, and assumes no responsibility for, the content, accuracy or completeness of the information presented.

Attendees should note that sessions may be audio- or video-recorded and may be published in various media, including print, audio and video formats without further notice. The presentation template and any media capture are subject to copyright protection.

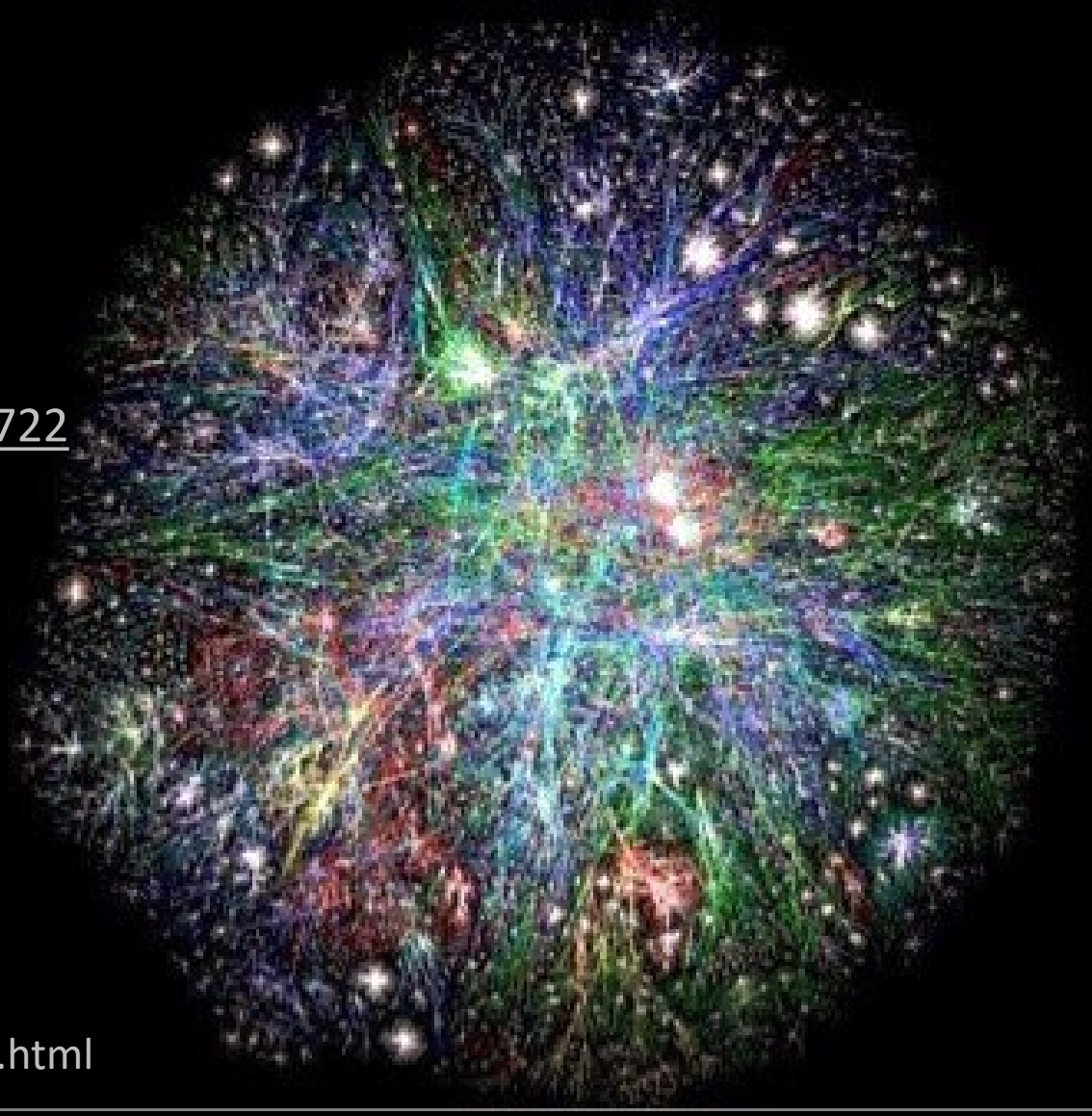
Introduction

Karen Reinhardt

- As per bio

Contacts:

- Karen.Reinhardt@badgersecurity.net
- www.linkedin.com/in/karen-reinhardt-farr-9825722
- Twitter: @Farrside42



http://www.vlib.us/web/worldwideweb3d.html

RSAConference2022

Zero Trust and Cryptography

So How Does Cryptography Work with Zero Trust?

What is Zero Trust?

Perimeterless Security

"Zero trust (ZT) is the term for an evolving set of cybersecurity paradigms that move defenses from static, network-based perimeters to focus on users, assets, and resources."[1]

- No implicit trust, no assumptions
- Trust must be earned, "never trust, always verify"[2]
- Verify (authentication, authorization, policy checking) users and devices before access granted

Core Principles

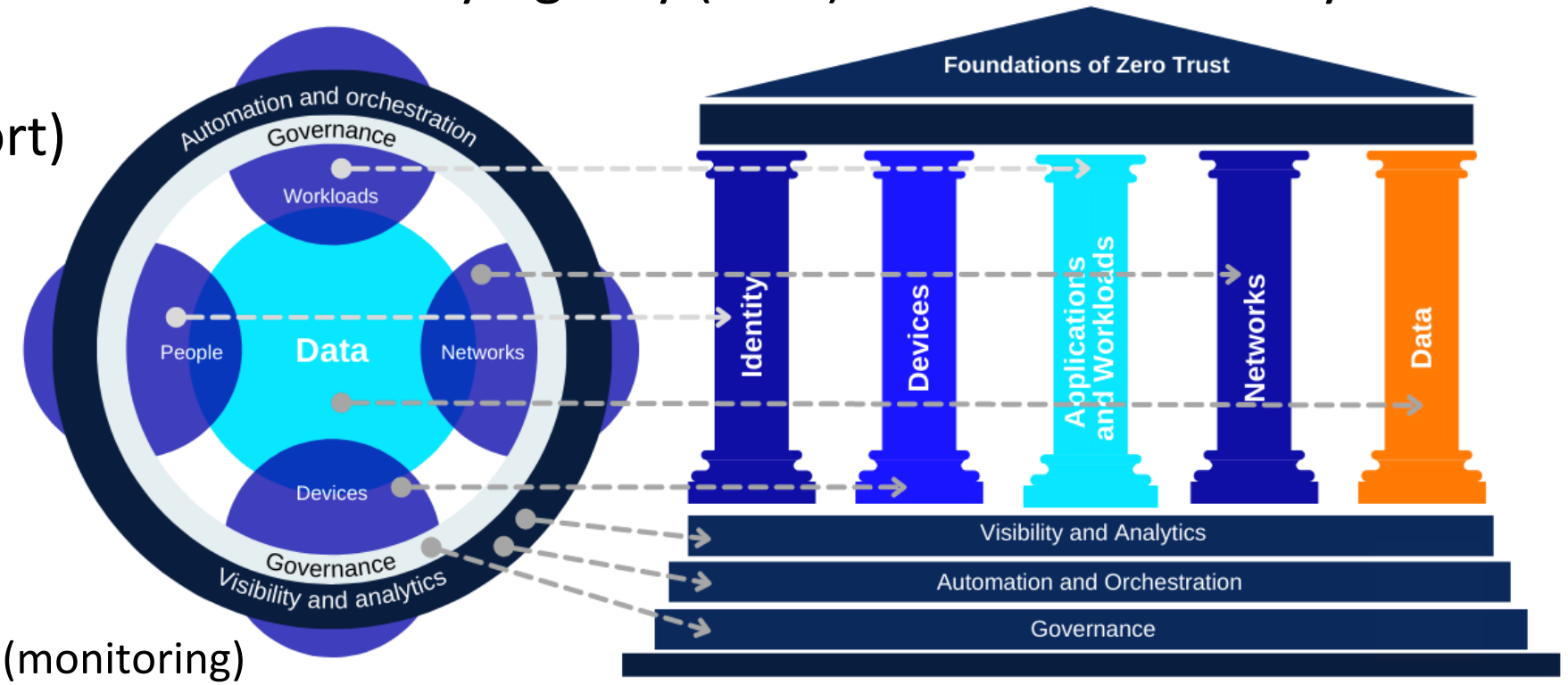
- Authenticate and verify identity
- Centralized Management (Governance)
- Least Privilege Access
- Inspect and Log (Monitor & Audit)

Implementations of ZT[3]

Zero Trust Architecture (ZTA), Zero Trust Network Access (ZTNA) Zero Trust Edge (ZTE).

The Zero Trust Model

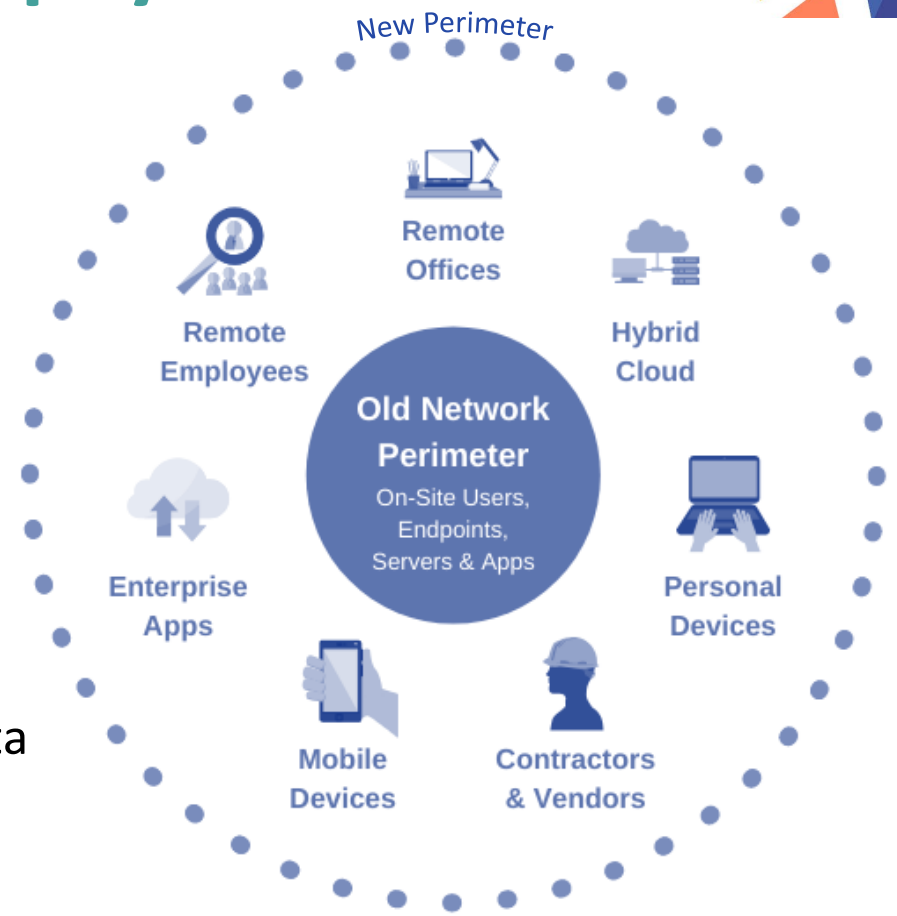
- Cybersecurity and Information Security Agency (CISA) Zero Trust maturity model
- Pillars (Areas of effort)
  - Identity (People)
  - Devices
  - Networks
  - Applications
  - Workloads
- Themes[4]
  - Visibility and Analytics (monitoring)
  - Automation and Orchestration
  - GOVERNANCE (Centralized control)



Mapping Forrester to CISA for modern Zero Trust ©Forrester

Zero Trust, Perimeters and Cryptography

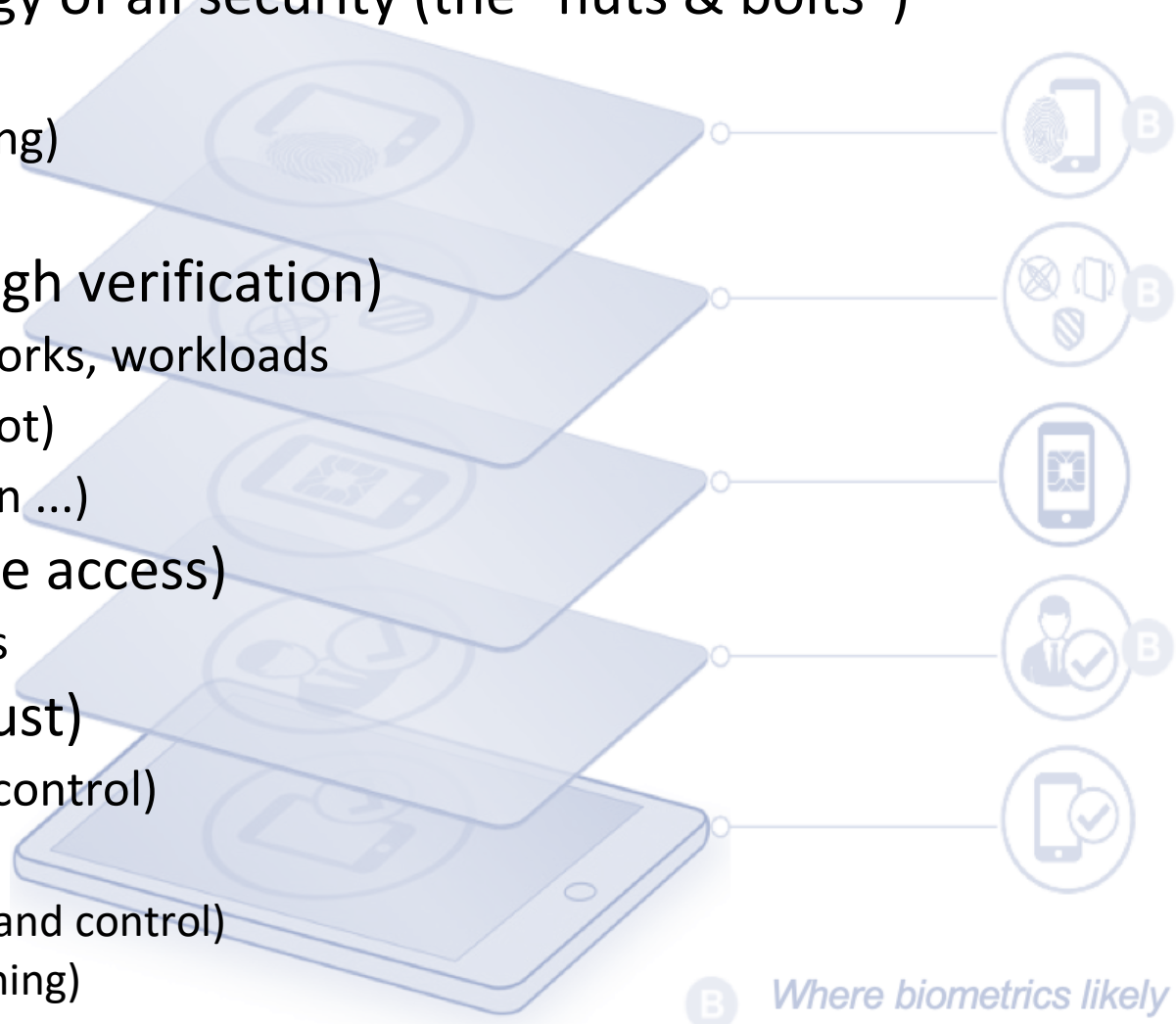
- Traditional Networks: Static perimeters
  - End-user/client devices become part of network
- Zero Trust: Dynamic, ephemeral perimeters
  - Proof of control (authentication, etc., repeated regularly)
  - Establish data access through trusted identity
  - Layered auth for stronger security
- Cryptographic perimeters use encryption
  - Access to data by access to keys rather than proximity to data
  - Still must prove their identity
  - Data in transit: Example TLS Session
  - Client is given access to data rather than being added to the network (TLS* vs VPN)



*Is anonymous SSL/TLS really Zero Trust since it only proves trust one-way? Even if user authenticates?

Cryptography for Zero Trust

- Cryptography is underlying technology of all security (the “nuts & bolts”)
  - Enables stronger identity verification
  - Enables integrity (tamper evidence, signing)
  - Enables confidentiality
- Provides proof of Identity (*trust* though verification)
  - Control of keys for devices, people, networks, workloads
  - Device based “Root of Trust”, (Secure Boot)
  - Layers identities (Device, user, application ...)
- Encrypts data (enforces least privilege access)
  - Access to decrypt through control of keys
- Assures integrity (builds & verifies trust)
  - Endpoint: ”Bottom-up“ integrity (device control)
  - Governance: “top-down” integrity
    - Automation and Orchestration (command and control)
    - Visibility & Analytics (tamper evidence, signing)
    - Governance (Authenticity of policy)



Where biometrics likely have a role

RSAConference2022

Modern Key Management

Chaos of Keys

The Problem with Security? Keys!

In an ideal world doors and treasure chests would open just because they can sense that it is you. In the real world, you need a key.

- Key Usage and Types of keys
  - Encryption –symmetric or asymmetric keys
    - Data Encryption Key (DEK) – encrypts/decrypts data element or store
    - Key Encryption Key (KEK) – encrypts another key; KEK grants access to keys or keystores
  - Identity – Establish identity through signing of data or authentication
    - Credential key(s) “Secrets” – Key(s) associated with a unique identity
    - Establish access to encryption keys

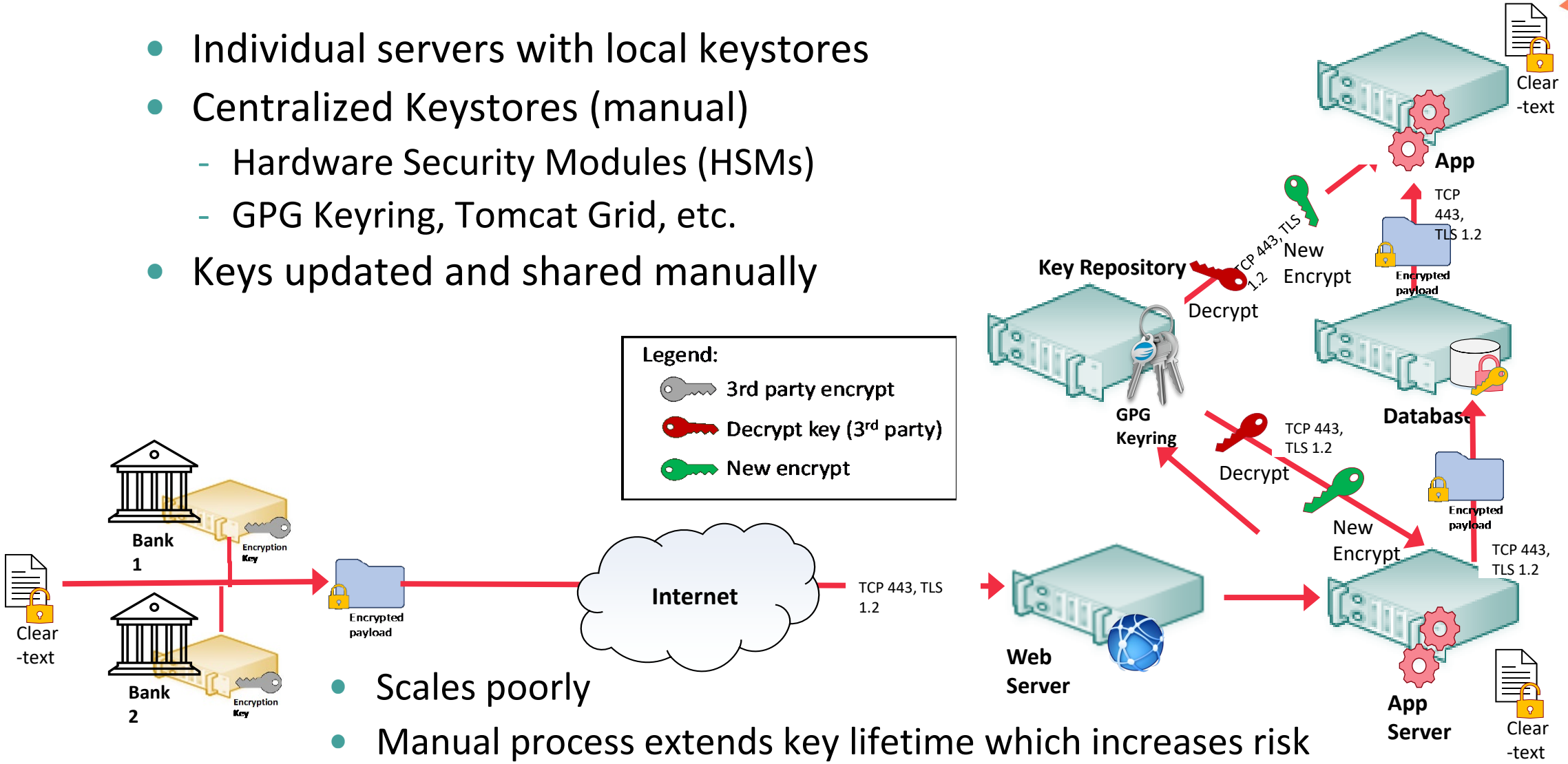
- But what happens when you:
  - lose the key (availability)
  - someone steals a copy
    - They impersonate you and can steal your stuff (confidentiality-encryption, integrity-credential/signing)
    - They can remove your stuff or change it (availability, and integrity)

Trust but verify depends on the assurance of the control of keys

Manual Encryption & Key Management

”Old School”

- Individual servers with local keystores
- Centralized Keystores (manual)
  - Hardware Security Modules (HSMs)
  - GPG Keyring, Tomcat Grid, etc.
- Keys updated and shared manually



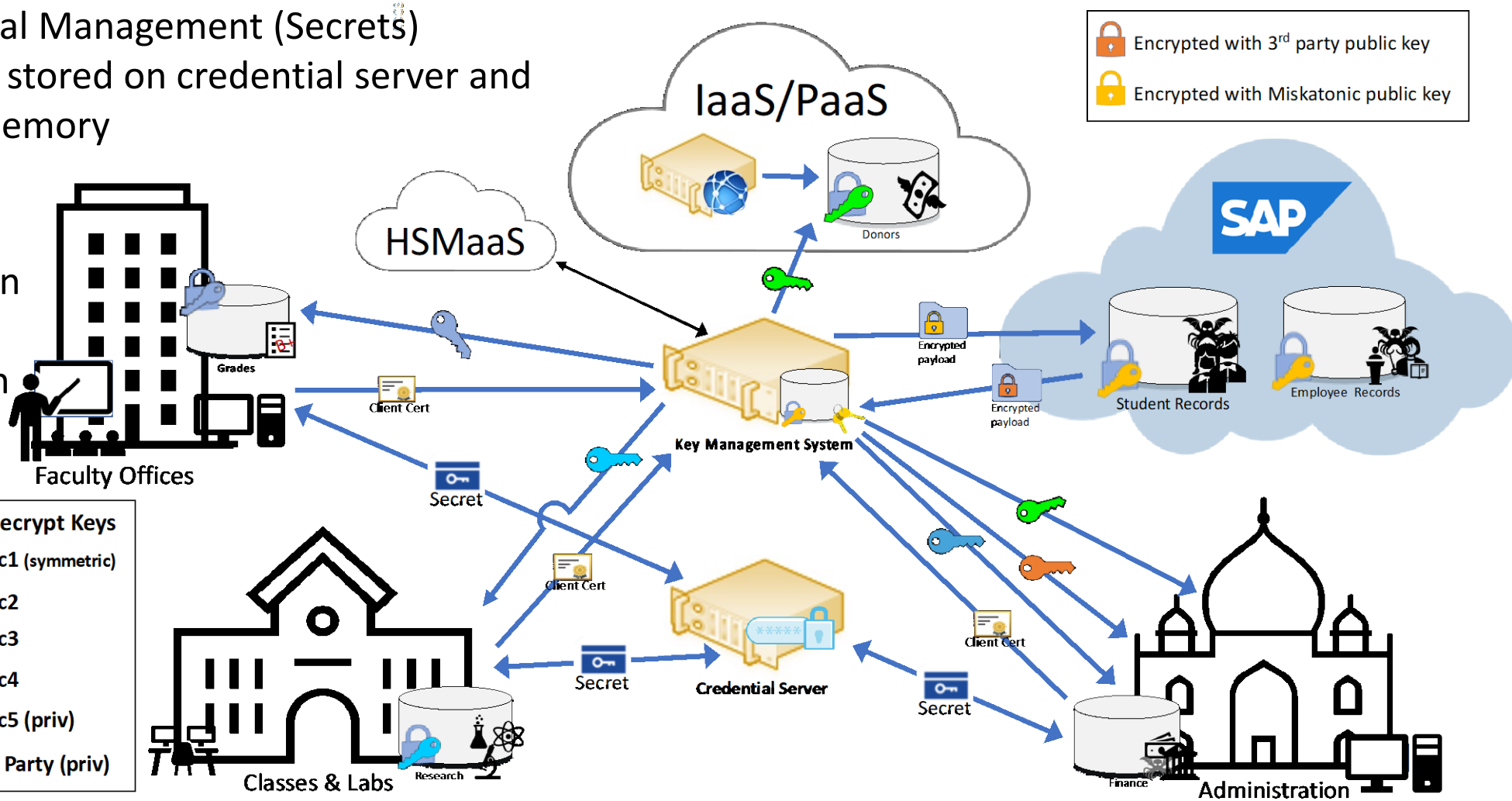
- Scales poorly
- Manual process extends key lifetime which increases risk
- GPG passphrase, “secret”, stored locally

Data Encryption Example

Hybrid Model – Miskatonic U

Today: Marching Towards Zero Trust

- Enhanced Credential Management (Secrets)
- Private key for cert stored on credential server and loaded into local memory
- Client cert used to access KMS
- Separate encryption and identity Keys
- Integration through export/import



Separation for Security:

DEKs from data, Secrets from the data/keys protected, credential signers from credentials

Hybrid Model Challenges

- Keys are segmented by environment
- Integration limited by ability to import or export keys
- Decrypt/re-encrypt required across boundaries
- Variety of authentication methods and credentials
- Security of stored keys vary
  - Local vs centralized
  - Separation of encryption & authentication/identity Keys
  - Separation of Keys and Data
- Lack of key management automation

RSAConference2022

Into the Future

Controlling the Chaos of Keys

Blending Hybrid Environments to Create ZT Cryptographic Environments/Realms and Virtual Perimeters

Centralized Key Management Systems (KMS)

- Facilitates governance and automation
  - Needed at scale for ephemeral keys
  - Key management at scale
- Improved security for keys
  - Better HSM Integration (master key(s), KEKs, less clients)
  - Secure and compliant keys generation (FIPS 140-2, lv2+)
  - Easier key rotation: new keys pushed/pulled
- You can, but should you?
  Combine Key and Secrets Management?
  - Encryption requires key archival to ensure availability
  - Identities must be tightly controlled (non-repudiation-ish)
  - Separation of encryption keys and identity keys is best practice
- Central governance with distributed keys?
  - Central Control of governance (policy enforcement)
  - Distributed key databases, centrally managed

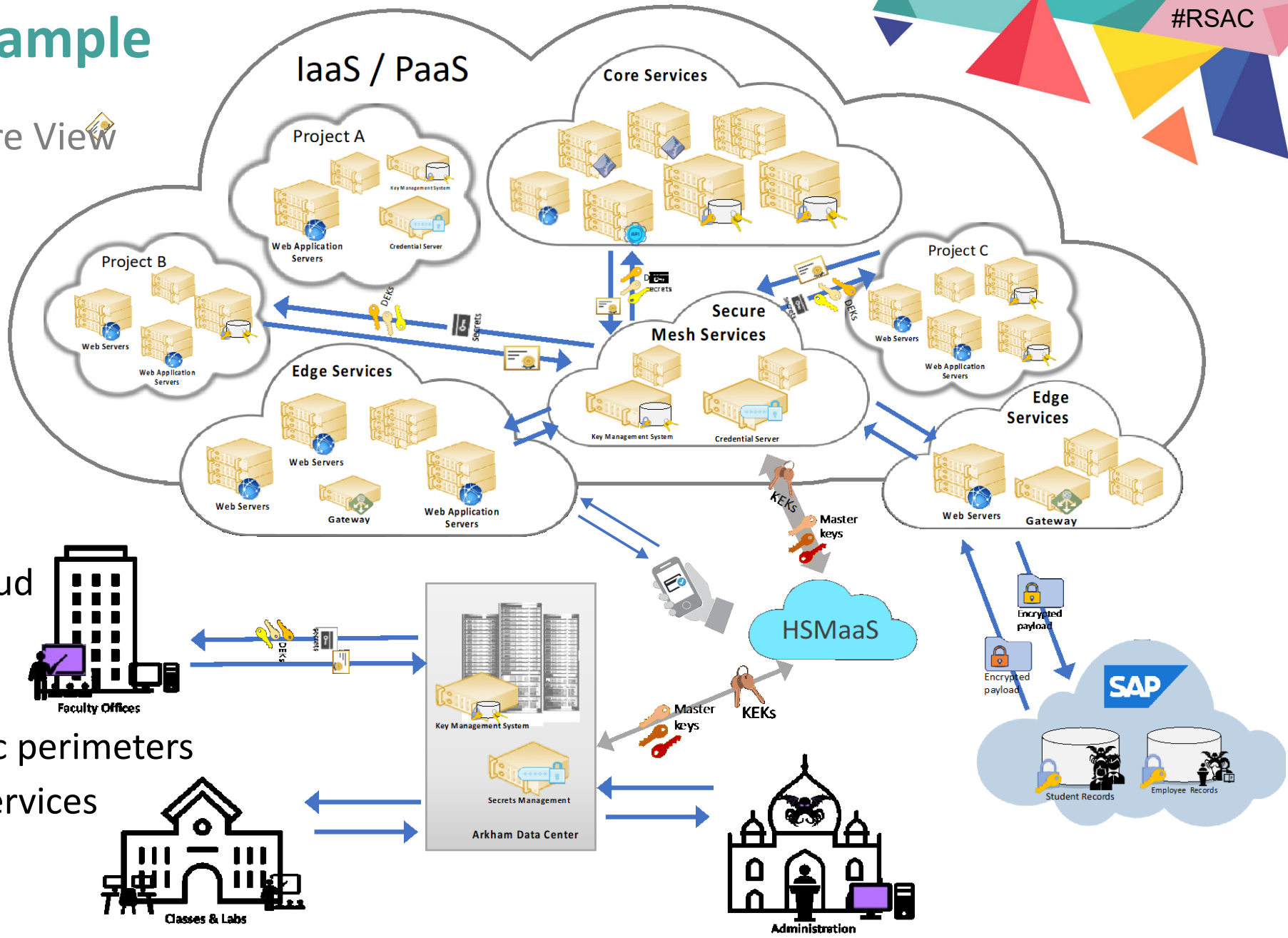
Data Encryption Example

Miskatonic U

A Very Over-Simplified Future View

ZT Crypto

- Continued migration to Cloud
- Cloud native services
- Core vs Edge
- Continued erosion of classic perimeters
- Distributed availability of services
- Access from anywhere
- Centralized governance

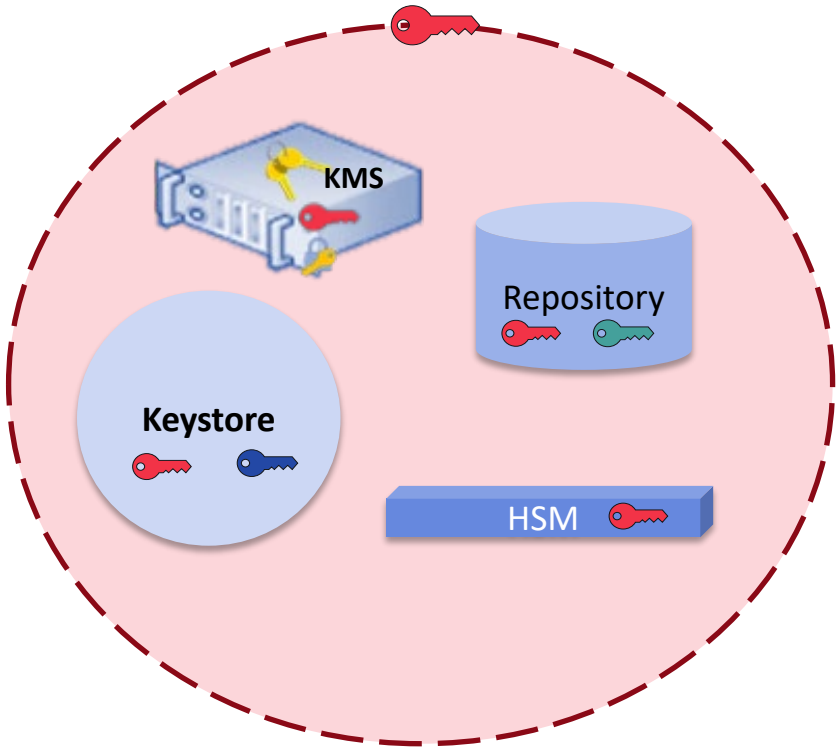


Cryptographic Realms ("Crypto Perimeters")

Security Worlds, Encryption Domains, Zones, etc.

Definition: A Cryptographic realm is a non-network-based environment based on the use of common keys to establish controls through the use of cryptographic operations (encryption, authentication, and/or signing).

- Components of a cryptographic realm:
  - Local Keystores
  - Repositories (Databases, HSMs)
  - Management systems (Key, secrets, credentials)
- Cryptographic realms can be used to:
  - Control what applications/workloads can run (signing)
  - Control access to data (encryption)
  - Layer device, application/workload, end-user controls through authentication and encryption
  - Establish and re-enforce organizational identity

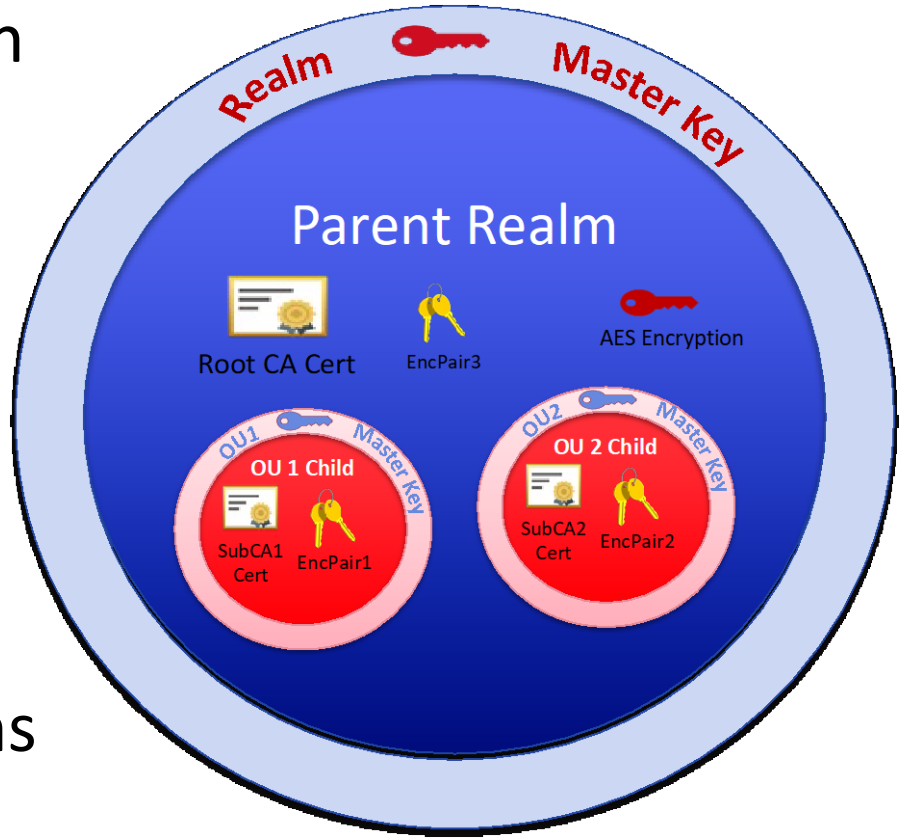


Cryptographic Realm Example
Red Key

Cryptographic Realms ("Crypto Perimeters")

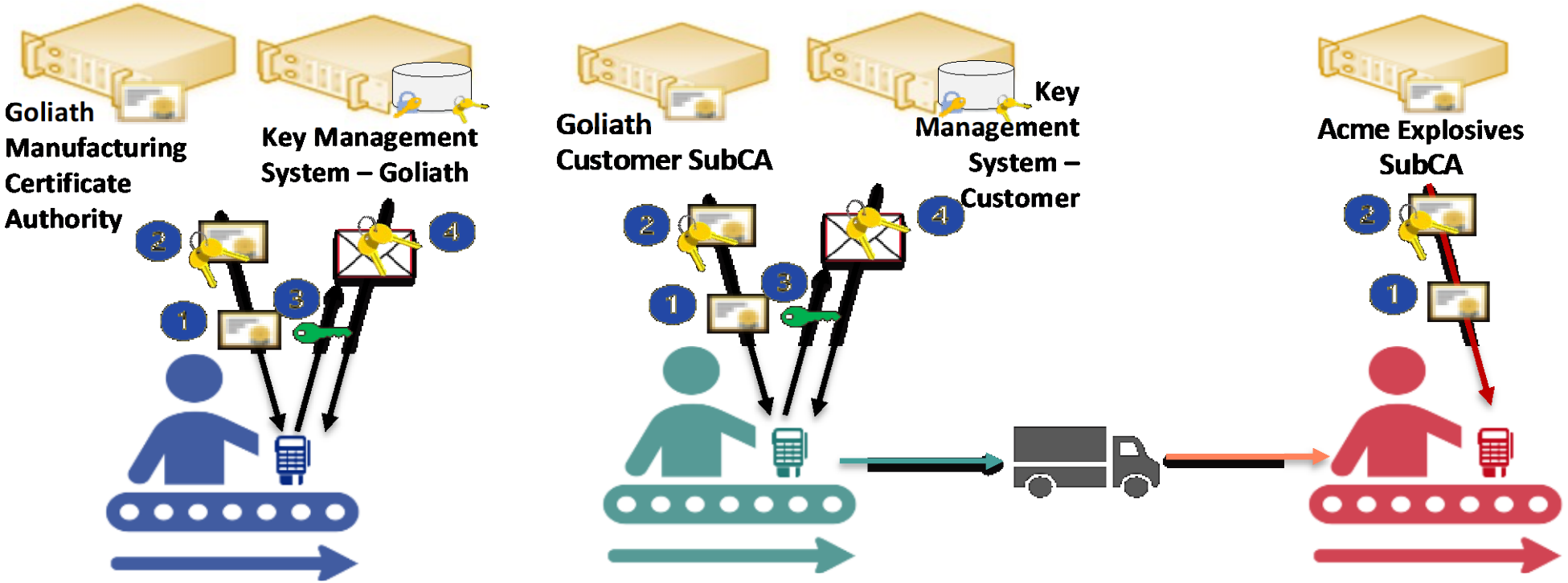
Types of Realms

- **Encryption Realms** control access to data through encryption.
  - Example: HSMs, encryption frameworks/solutions
- **Trust Realms** (aka Trust Domains) create trust of identities
  - Depends on level of assurance (controls)
  - Organizational identity which spans across environments
- **Parent and Child Realms:** Access to keys can be further segregated though the use of Child Realms wrapped within a Parent Realm
  - Can layer identity and Encryption
  - Ex. identity data (secrets, private keys) are protected within an Encryption Realm to create the Trust Realm



Example of Cryptographic Realms

Goliath Credit Card Services



1. Goliath Root Cert is flashed to security chip
2. Device certificate and keys pushed
3. Device public key pulled from device and wraps AES encryption keys
4. Encryption keys sent to device, decrypted, and flashed to chip

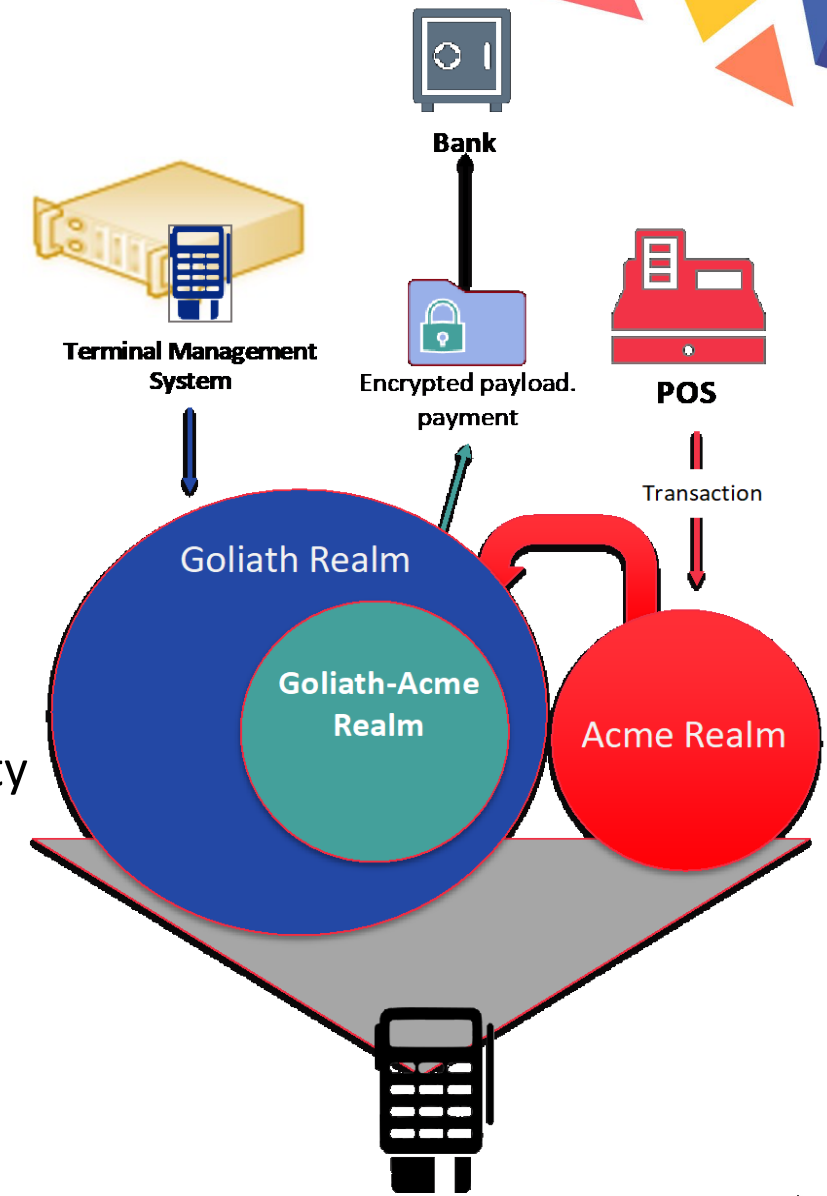
1. Goliath Acme SubCA Cert is pushed
2. Goliath-Acme certificate and keys pushed
3. Acme public key pulled from device and wraps customer's bank keys
4. Encrypted bank keys sent to device, decrypted and installed in chip-protected store

1. Acme CA trust chain is pushed
2. Acme certificate and keys pushed to customer key store (separate from Goliath store)

Example of Cryptographic Realms

Goliath Credit Card Services - Continued

- Keys for one realm do not grant rights to other realms
- Subordinate realms enable segmentation (by key access)
  - Specified access to parent keys
  - No access or defined access to other children
- Parent realm used to administer child
- Parallel realms are separate but can be integrated
- Enablement methods
  - PKI Trust domains (hierarchical trust) with asymmetric identity and/or encryption wrapping keys
  - Asymmetric encryption parents (wrappers) with asymmetric and symmetric keys protected in child
  - Pure encryption with parent-controlled master keys and derived child keys



Expanding Crypto Realms (& Key Management) Across Boundaries

Some Considerations

Making keys available in multiple environments is the trick to creating a realm and integrating realms

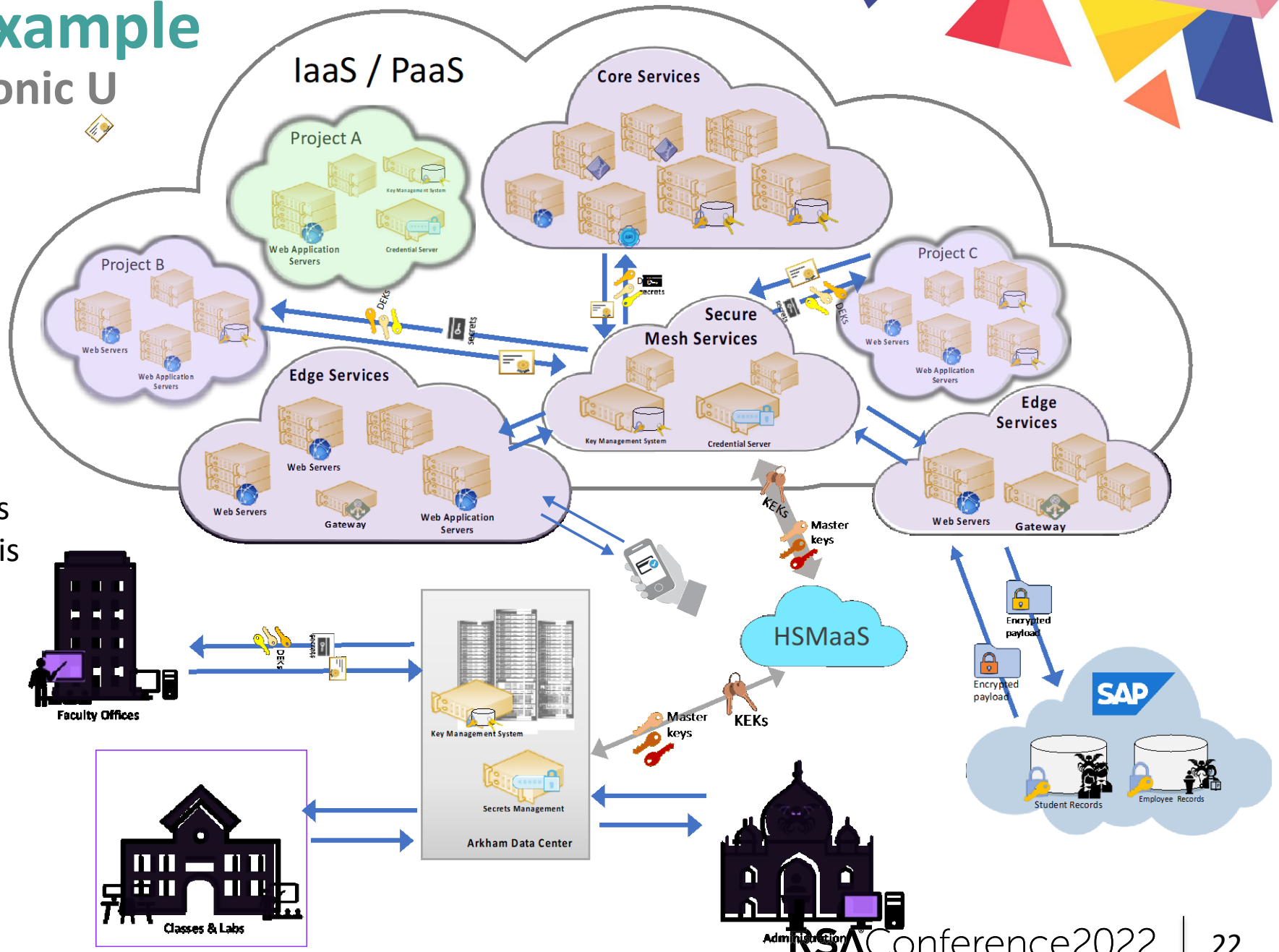
- Access to keys
  - Example: Publicly trusted CAs (Trust Domains)
- Is a dedicated KMS needed for each environment, or can it access a KMS in another environment?
  - Edge vs Core Services?
- Layering secrets management and encryption key management
  - Separation data from the encryption keys, identity keys from encryption keys, credentials from credential signers
- What about digital signing (credential signing, other)?
  - Are signers different from identity credentials? (Should a credential signer be kept separate from a credential?)

Data Encryption Example

Blended Model – Miskatonic U

By Cryptographic Realm

- Keys and Secrets Databases protected by HSM
- Miskatonic Cryptographic Realm (purple)
  - Defined by common keys
  - HSM Security Domain spans across on-prem and Cloud (HSMaaS in this example)
  - Keys replicated across KMS databases
- Standalone realm (green) Separate KMS
  - No Common keys



Through the Crypto Mist

Applying What We Have Learned

For Today

- Centralized Management does not have to mean “all the eggs in one basket”.
- Ensure separation based on use case and risk, not environment
- Ensure Least privilege
- Key Management vs Credential Management vs Secrets Management
  - All keys are not equal

Thinking about the future

- Maintaining integrity (trust) to assure confidentiality
  - Identity as main control for key access (as apposed to proximity)
  - Weaving Integrity and Confidentiality rather than combining (2 streams)
- Distributing confidentiality and trust In the future
  - Hierarchy, Mesh. Blockchain ledgers, ...
- Establishing Control
  - Proof of control without hardware root?
  - Combined control of virtual environments (split key, Shamir’s secret sharing)?

Thank You

Citations and Sources

Citations:

1. Rose, Bochert, Mitchell, Connelly, “Zero Trust Architecture”, NIST Special Publication 800-207, August 2020), https://doi.org/10.6028/NIST.SP.800-207
2. Turner, “Zero Trust Is Not A Security Solution; It’s A Strategy”, Forrester, February 18, 2021, https://www.forrester.com/blogs/zero-trust-is-not-a-security-solution-it-is-a-strategy/
3. Akamai, “What is Zero Trust? Zero Trust Security Model”, 2022, https://www.akamai.com/our-thinking/zero-trust/zero-trust-security-model
4. Chris Beinecke, “Zero Trust M-22-09: What you need to know”, February 25, 2022, https://swishdata.com/zero-trust-m-22-09-what-you-need-to-know/

Sources:

Holmes, Burns, Mellen, Pollard, Cerrato, Cser, “OMB’s Zero Trust Strategy: Government Gets Good“, Forrestter, Feb 1, 2022, https://www.forrester.com/blogs/ombs-zero-trust-strategy-government-gets-good/

Prime Factors (2022), “Encryption Key Management”, https://www.primefactors.com/data-protection/encryption-key-management/

Chris Beinecke, “Zero Trust M-22-09: What you need to know”, Swift Data, February 25, 2022, https://swishdata.com/zero-trust-m-22-09-what-you-need-to-know/

Cybersecurity & Infrastructure Security Agency, “ZERO TRUST MATURITY MODEL”, 2022, https://www.cisa.gov/zero-trust-maturity-model