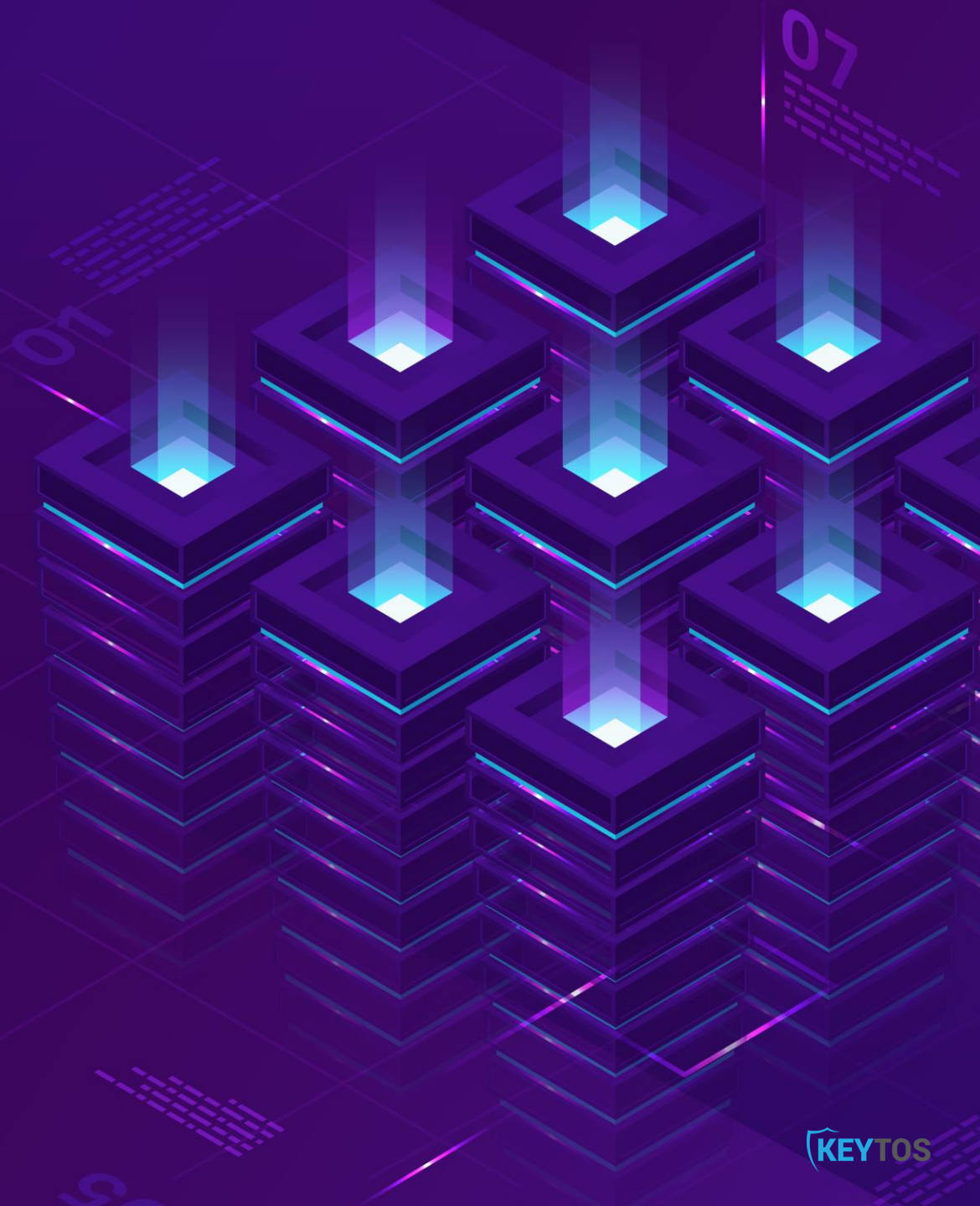
KEYTOS

EZSSH

SSH Made Easy!



KEYTOS

PROBLEM OVERVIEW

Cloud adoption is making companies move to a zero-trust networks.

99% of compromises involve a stolen credential[1]

Stolen SSH credential attacks are on the rise.

Companies are spending millions of dollars on improving their corporate identity.

Linux servers do not use Active Directory Accounts.

Thousands of keys are leaked on GitHub each day.[2]

KEYTOS

IN THE NEWS

KEYTOS

WHY GO PASSWORD LESS?

Passwords are no longer secure due to brute force attacks.

72% of individuals reuse passwords in their personal life while nearly half (49%) of employees simply change or add a digit or character to their password when updating their company password every 90 days.

Compromised passwords are responsible for 81% of hacking-related breaches, according to the Verizon Data Breach Investigations Report.

Microsoft recently announced that a staggering 44 million accounts were vulnerable to account takeover due to compromised or stolen passwords.

KEYTOS

ARE SSH KEYS THE SOLUTION?

- Linux servers in large corporations have between 50 and 200 SSH keys.
  - 90% of those keys are not used.
  - SSH keys never expire.
  - 50-200 keys per server.
- Keys must be manually life cycled.
- No Advance Identity Protection
  - Conditional Access
  - Smart Alerting
  - Just in Time Access

- Hard to keep inventory of which key gives access to who.
- Engineers don't follow best practices to protect the keys.
- Current Linux Systems are protected in two ways:
  - Creating an account for production and sharing the credentials among engineers.
  - Creating accounts for each engineer in each of the servers.

KEYTOS

CURRENT WORKFLOW

Each User Gets an Account

Engineer goes to a site and learns how to create an SSH key.

Engineer creates the key.

Sends it to security team or server admin to be added to the server.

KEYTOS

CURRENT WORKFLOW

Each User Gets an Account

Security team adds it to the server.

Engineer gets access to the server and now can start their work.

When engineer no longer needs access, is the account removed?

KEYTOS

EACH ENGINEER CREATES AN ACCOUNT PROBLEMS

Engineer goes to a site and learns how to create an SSH key.

Poor key hygiene, no key clean up since it is hard to keep track of who still needs access.

Long and tedious access reviews.

High price to onboard new team member

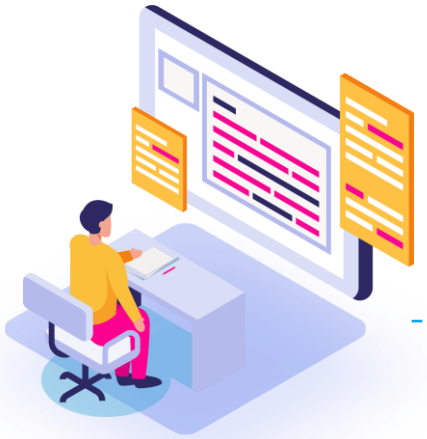
Key reuse over different scopes.

Keys are not properly protected by users.

KEYTOS

CURRENT WORKFLOW

Shared Accounts



Engineer goes to team wiki to get key locations

Many of these keys are reused between test and production.

Engineer gets the key from team shared location

Engineer saves the key in their system

Engineer accesses server and now can start their work

KEYTOS

ONE ACCOUNT FOR ALL ENGINEERS

Usually, keys are shared in unsecure ways such as: email, file shares, wikis, git.

Hard to rotate since all engineers would have to get the new key.

When an employee leaves, they can maintain access to servers.

Big insider threat opportunity (61% of CISOs worry about insider threats).

→ Not possible to know who did which change since all server logs show being done by the same account.

No approval flows to get access to server

Reuse of “team keys” for many services.

KEYTOS

YOUR CURRENT COST

SSH key inventory and access reviews for all your servers

Added risk for life cycling accounts out of the environment when no longer needed

Added risk for having engineers manage key and access

Security team time adding and removing keys

Engineering time creating and passing the keys

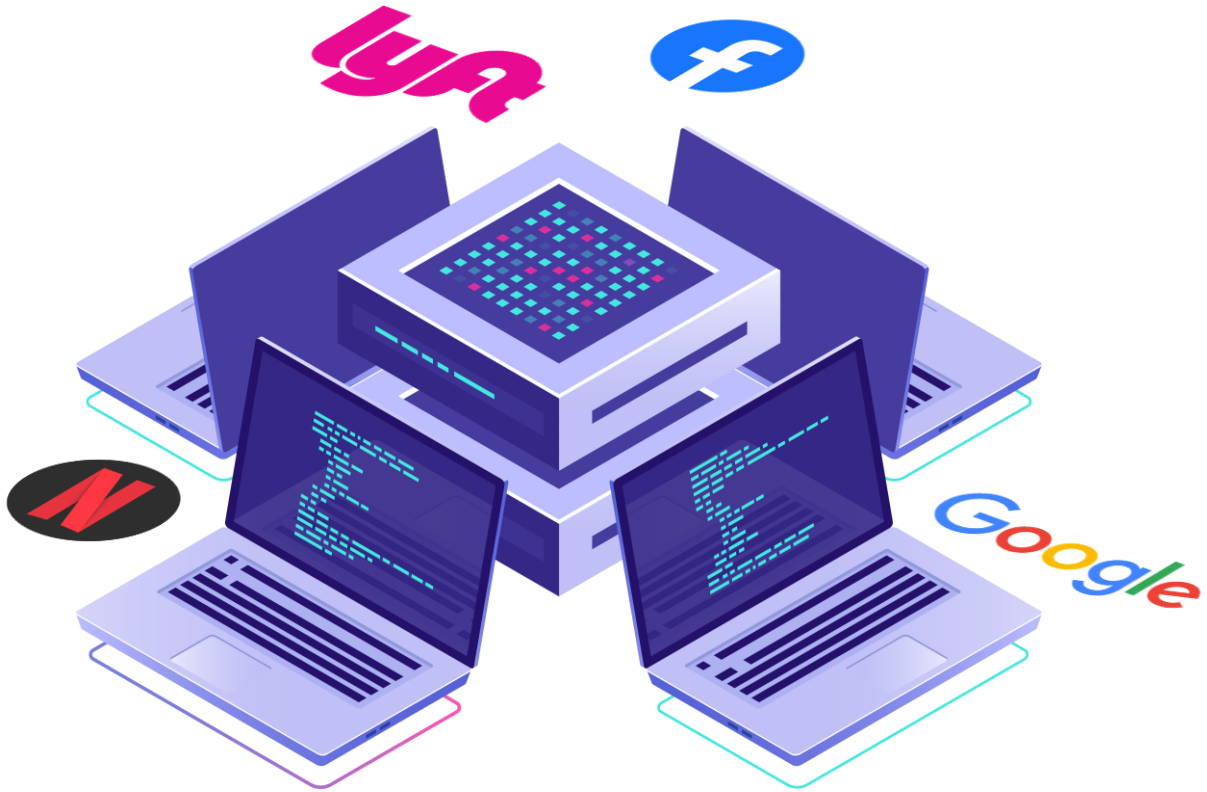
KEYTOS

SSH CERTIFICATES

Poorly documented

Need cryptographic knowledge

Manual setup and management



No automatic provisioning

No approval workflow

Most of the large companies use it (with custom built tools)

KEYTOS

HOW CERTIFICATES WORK

1. Request Access

2. Approval Workflow

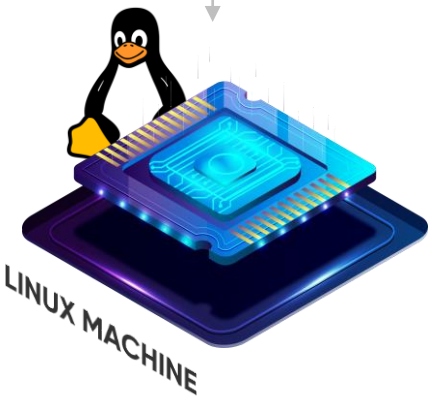
EZSSH

3. Request Signed Certificate

4. Return Signed Certificate

5. Return Signed Certificate

6. Access machine with Signed Certificate



KEYTOS

OUR SOLUTION

Seamlessly integrates with Azure

- Works with Azure Security tools such as Azure JIT and Azure PIM.
- Integrates with Azure RBAC for automatic access management.
- Automatically adds Azure Servers to your policies.

Works with hybrid and multi-cloud.

Easy setup for all your servers.

Uses your secure corporate account to create time bound certificates.

Makes security transparent to the user

Automatically onboards new team members

Approval workflow for critical environments

Automatically removes access when no longer needed

KEYTOS

EZSSH ADVANTAGES

Designed for Zero Trust networks

Reduce Onboarding time and cost by removing need to manage SSH keys

Remove key management overhead from engineers.

Reduce insider threat by having Just In Time Access with appropriate approval workflows.

Reduce audit costs with easy to Audit access logs

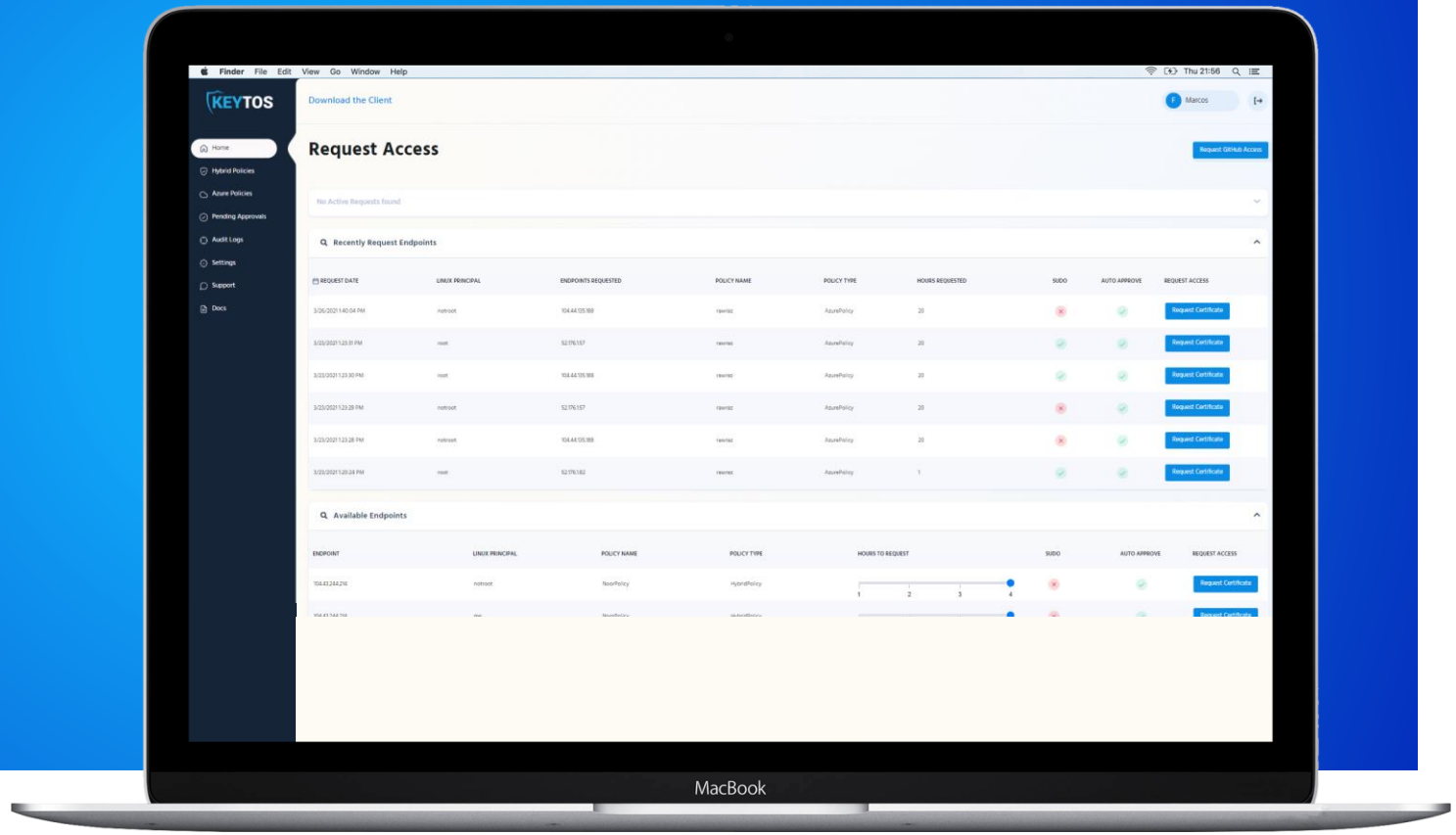
Reduce offboarding time and risk.

Native Linux Authentication no custom PAM module or code runs on your servers.

Bring your own Certificate Authority support

KEYTOS

DEMO


KEYTOS
Download the Client
Request Access
Home
Hybrid Policies
Azure Policies
Pending Approvals
Audit Logs
Settings
Support
Docs
Recently Request Endpoints
Available Endpoints
MacBook

KEYTOS

KEYTOS

EZGIT
Protecting GitHub with SSH Certificates

PROBLEM OVERVIEW

- Hackers are targeting developer credentials to steal code.
- SSH Keys are not properly managed by users.
- SSH Certificates are supported but there is no infrastructure to issue them.
- Need Secure infrastructure to run your own Certificate Authority.
- Conditional Access does not apply to the most critical operations

GitHub Breaches

GitHub leaks exposed up to 200,000 medical records: 4 details

threatpost Cloud Security / Malware / Vulnerabilities / InfoSec Insiders / Po

Podcast: Shifting Cloud Security Left With Infrastructure-as-Code

Report: Microsoft's GitHub Account Gets Hacked

Home > News > Security > Source code from dozens of companies leaked online

Source code from dozens of companies leaked online

Compromised SSH keys used to access popular GitHub repositories

June 3, 2015 By Pierluigi Paganini

Security experts Ben Cox explained that the official Github repositories of the UK Government, Spotify, and Python were accessed using compromised SSH keys.

KEYTOS

OPPORTUNITY

- GitHub is forcing you to go password-less in 2021.
  - Gives you an opportunity to modernize your development security stack.

Reduce surface area with short-term SSH Certificates

Make audits easier with easy to audit logs

Reduce engineer onboarding time

Make security transparent for your users.

KEYTOS

OUR SOLUTION

Uses your Secure Azure AD Identity for Authentication of your developers.

Seamlessly integrates with our VM offering

Easy setup with any Git offering.

Uses your secure corporate account to create time bound certificates.

Makes security transparent to the user

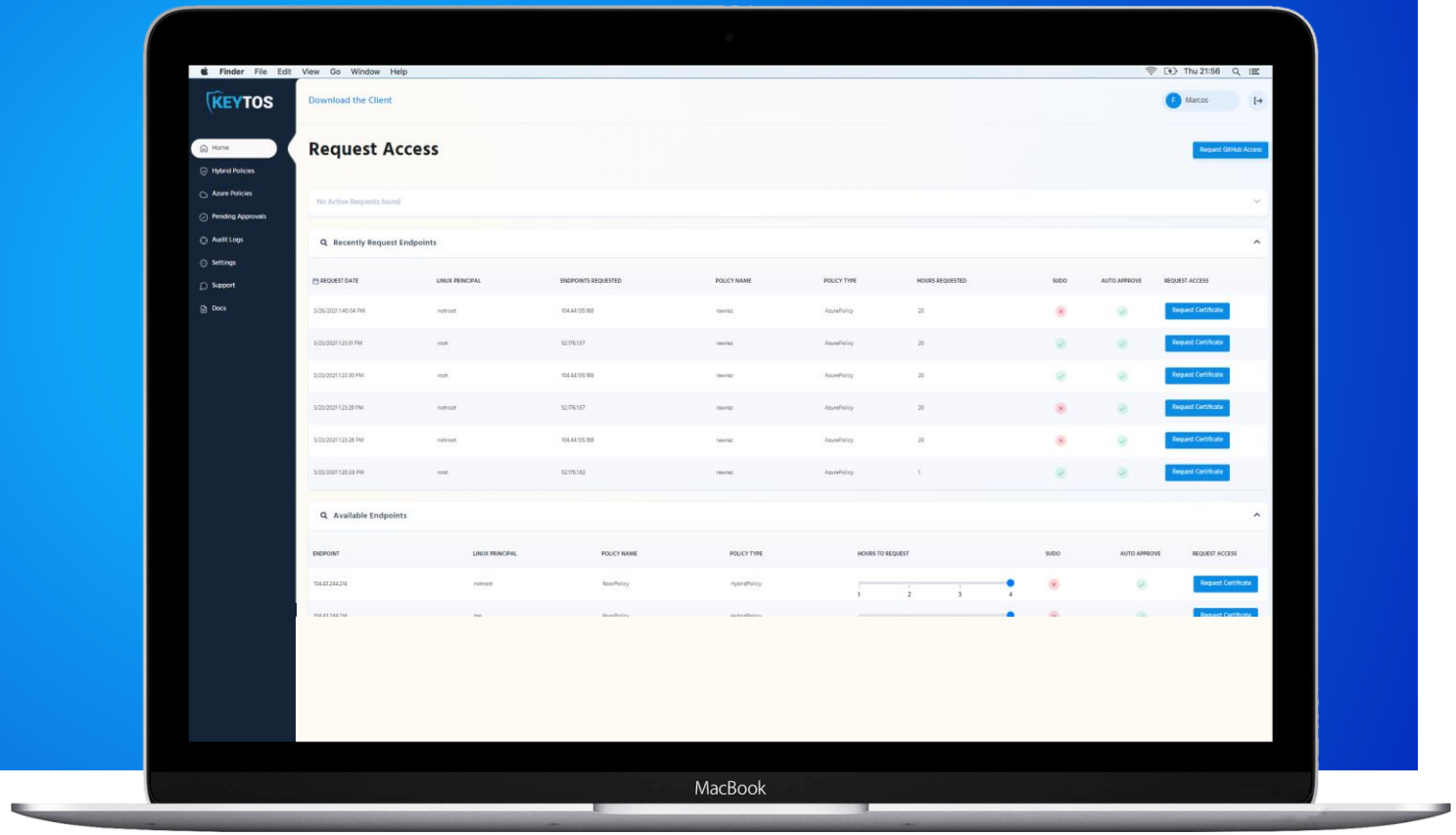
Automatically onboards new team members

Integrates with any git tool that uses ssh-agent as authentication method.

Automatically removes access when no longer needed

KEYTOS

DEMO


KEYTOS
Download the Client
Request Access
Home
Hybrid Policies
Azure Policies
Pending Approvals
Audit Logs
Settings
Support
Docs
Recently Request Endpoints
Available Endpoints
MacBook

KEYTOS


KEYTOS
EZCA
PKI Management
Made Easy!

KEYTOS

Running PKI In-House is Complex & Expensive

- Expensive Infrastructure
  - Unlike other services, PKI requires far more than a server. From HSMs in secure locations, to external services. PKI costs can add up quickly
- Keeping a secure and compliant PKI requires skill, knowledge, and discipline.
- Having a geo-redundant PKI deployment requires multiple secure locations across the world.
- Current solutions do not scale to current cloud needs.

KEYTOS

Advantages of Adopting EZCA

- Lower up to 30% of PKI cost.
- **Free Up Your Security Team:** Focus your security team on more pressing issues.
- **Secure and Compliant:** Run a PKI meeting and exceeding the highest levels of security and compliance.
- **Highly available and scalable infrastructure:** Take advantage of our geo-redundant infrastructure to run a highly available and scalable PKI
- **Full visibility into your organization:** maintain full visibility of the health of your organization's certificates with our dashboards.

KEYTOS

Automate Certificate Operations

- **Monitor your organization's certificate lifecycle:** Our advance dashboards and logs give you full visibility to all the certificates issued by EZCA.

- **Automate Certificate Issuance and Lifecycle:** Leverage our integrations with Key Management systems to automate the creation and lifecycle of your certificates.

- **Coding Best Practices:** EZCA offers sample code for authenticating using certificates.

KEYTOS

IoT Ready

- EZCA was design with the growing IoT field in mind
- Stop Hardcoding Credentials
- IoT ready samples for creating and rotating certificates.
- Manufacturing integration for Root of Trust establishment.
- Free design assessments with industry leading architects.

Our Solution

Seamlessly integrates with Azure Key Vault

Makes security transparent to the user

Works with hybrid and multi-cloud

Automates certificate lifecycle.

Supports Bring your own CA and bring your own HSM.

Gives you visibility into your Organization's SSL Health

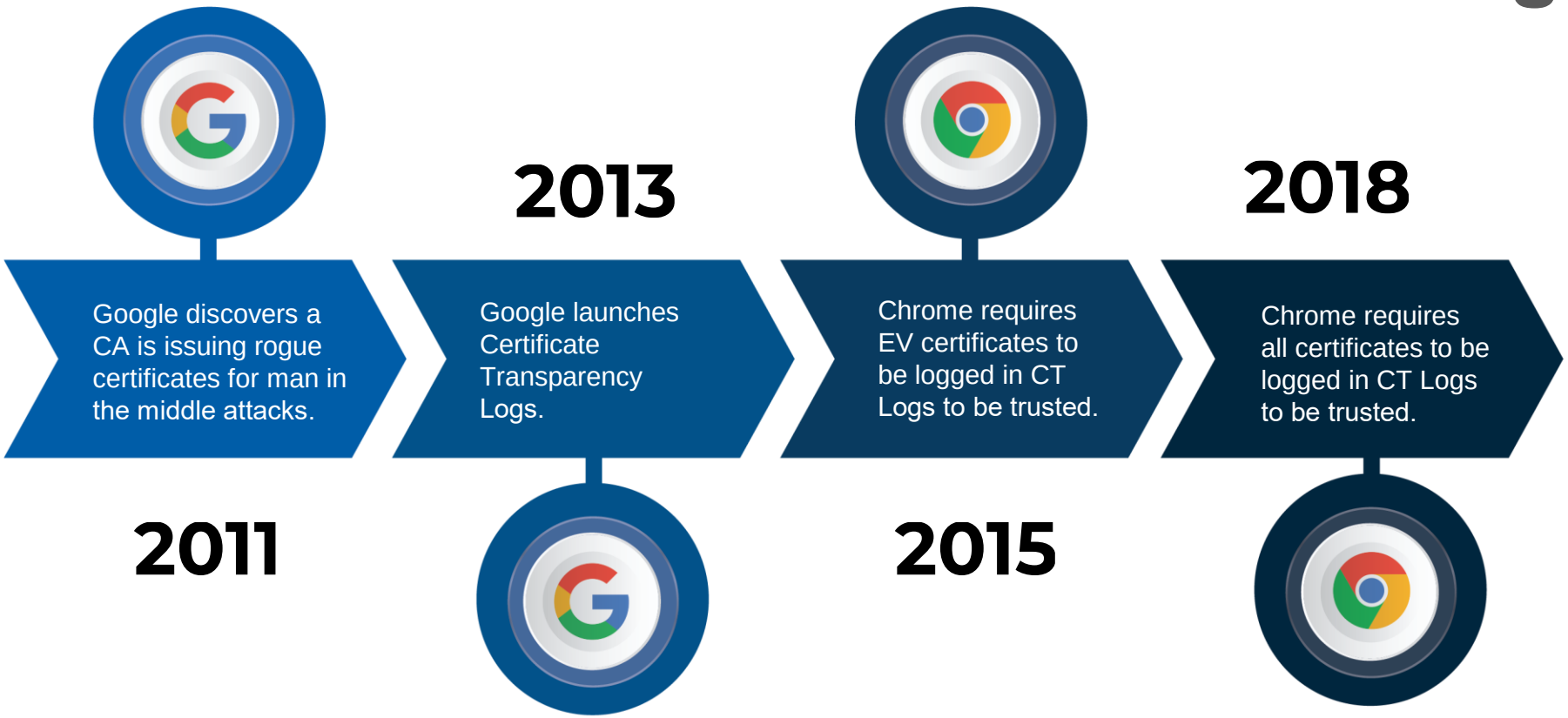

KEYTOS
Home
Certificates
Domains
Certificate Authorities
Pending Requests
Audit Logs
Settings
Support
Docs
Flamingo
Dashboard
Active Certificates
17
Revoked Certificates
5
Registered Domains
6
Certificates
16
CAs By Status
Active
CAs By Key Type
RSA 4096
ECDSA P256
Certificate Creation Type
Auto Renew Enabled


KEYTOS


KEYTOS
EZMONITOR

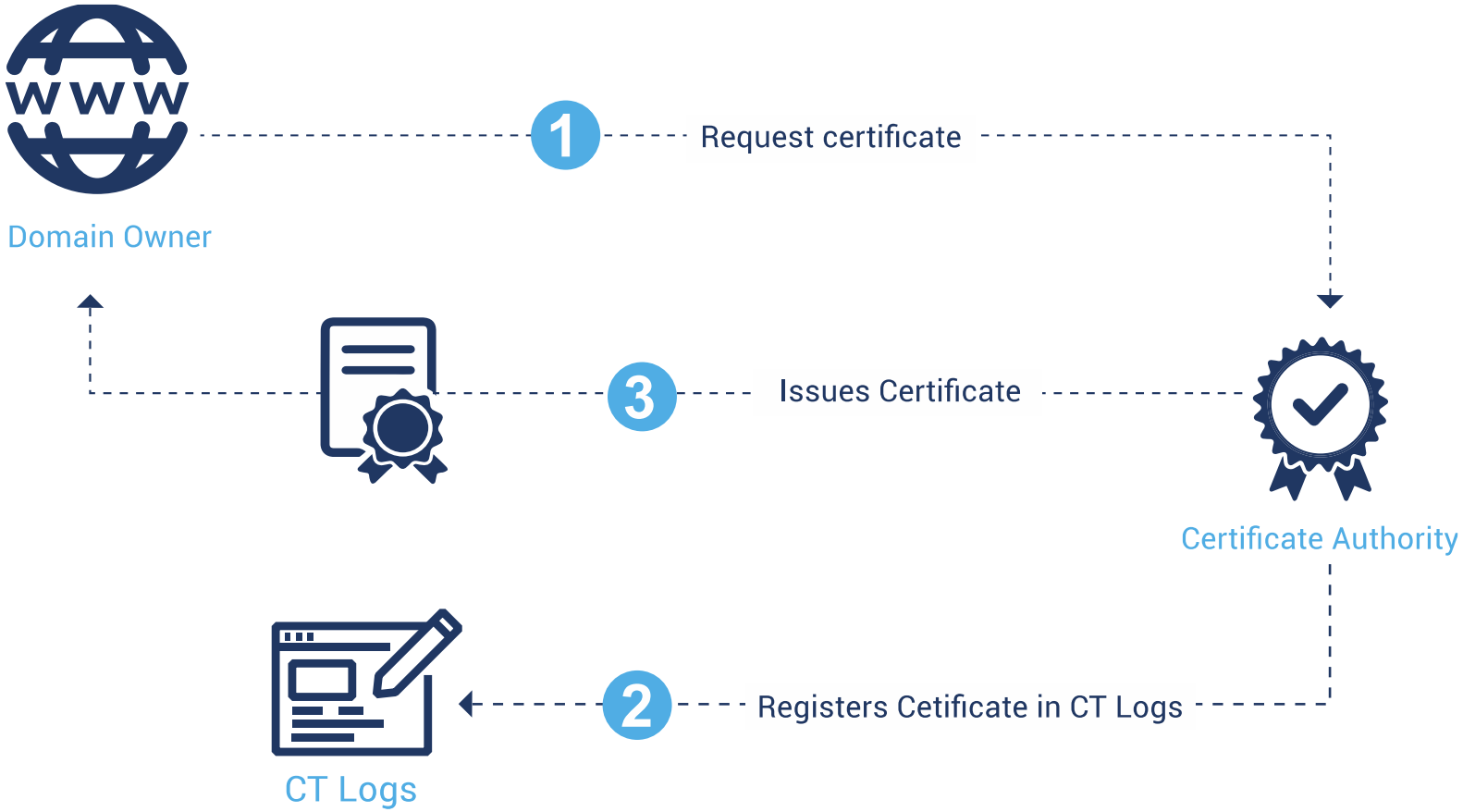
KEYTOS

History of CT Logs



KEYTOS

How CT Logs Work



KEYTOS

Domain Owner

CT Logs

Alert domain owner if a change is detected

EZMONITOR

Monitors Domains

KEYTOS

EZMonitor Alerts

- Monitor unauthorized certificate issuance.
- Alert on not renewed expiring certificates.
- Alert on renewed certificates that were not installed.
- Monitor new subdomains.
- Alert on subdomains containing your domain.
- Alert on certificates issued by new CA.
- New vulnerabilities monitoring.


KEYTOS

KEYTOS

ADCS Current Needs (Per Geo)

- Secure Rack
- HSM
- HSM Management Cards
- HSM Management Cards Management System
- Windows PKI Server
- Multiple CRL Web Servers
- Monitoring Solution for Certificate Uptime
- Monitoring Solution for CRL Uptime

KEYTOS

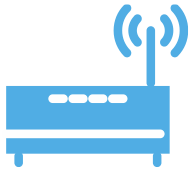
2021 had the highest average cost in 17 years

Data breach costs rose from USD 3.86 million to USD 4.24 million, the highest average total cost in the 17-year history of this report.

Security AI had the biggest cost-mitigating effect

Automation and security artificial intelligence (AI), when fully deployed, provided the biggest cost mitigation, up to USD 3.81 million less than organizations without it.

Remote work due to COVID-19 increased cost



The average cost was USD 1.07 million higher in breaches where remote work was a factor in causing. the breach, compared to those where remote work was not a factor.

A zero trust approach helped reduce cost

The average cost of a breach was USD 1.76 million less at organizations with a mature zero trust approach, compared to organizations without zero trust.

KEYTOS

Compromised credentials caused the most breaches

The most common initial attack vector, compromised credentials, was responsible for 20% of breaches at an average breach cost of USD4.37 million.

Cloud migration impacted costs and containment

Organizations further along in their cloud modernization strategy contained the breach on average 77 days faster than those in the early stage of their modernization journey.

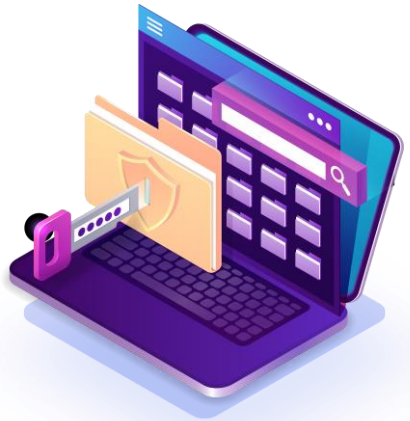
KEYTOS

Problem Overview

1. Current PKI solutions were designed 20 years ago and does not scale to current cloud scale needs.
2. On-premises Certificate Issuing and Management tools cannot scale to Cloud Scale Management
3. Running ADCS securely requires a very advance set of skills.
4. Having a geo-redundant PKI deployment requires a lot of upfront investmet
5. Automate Certificate Rotation
6. HSMs are expensive and hard to set up

HISTORY OF SSH AUTHENTICATION

1995 SSH Created

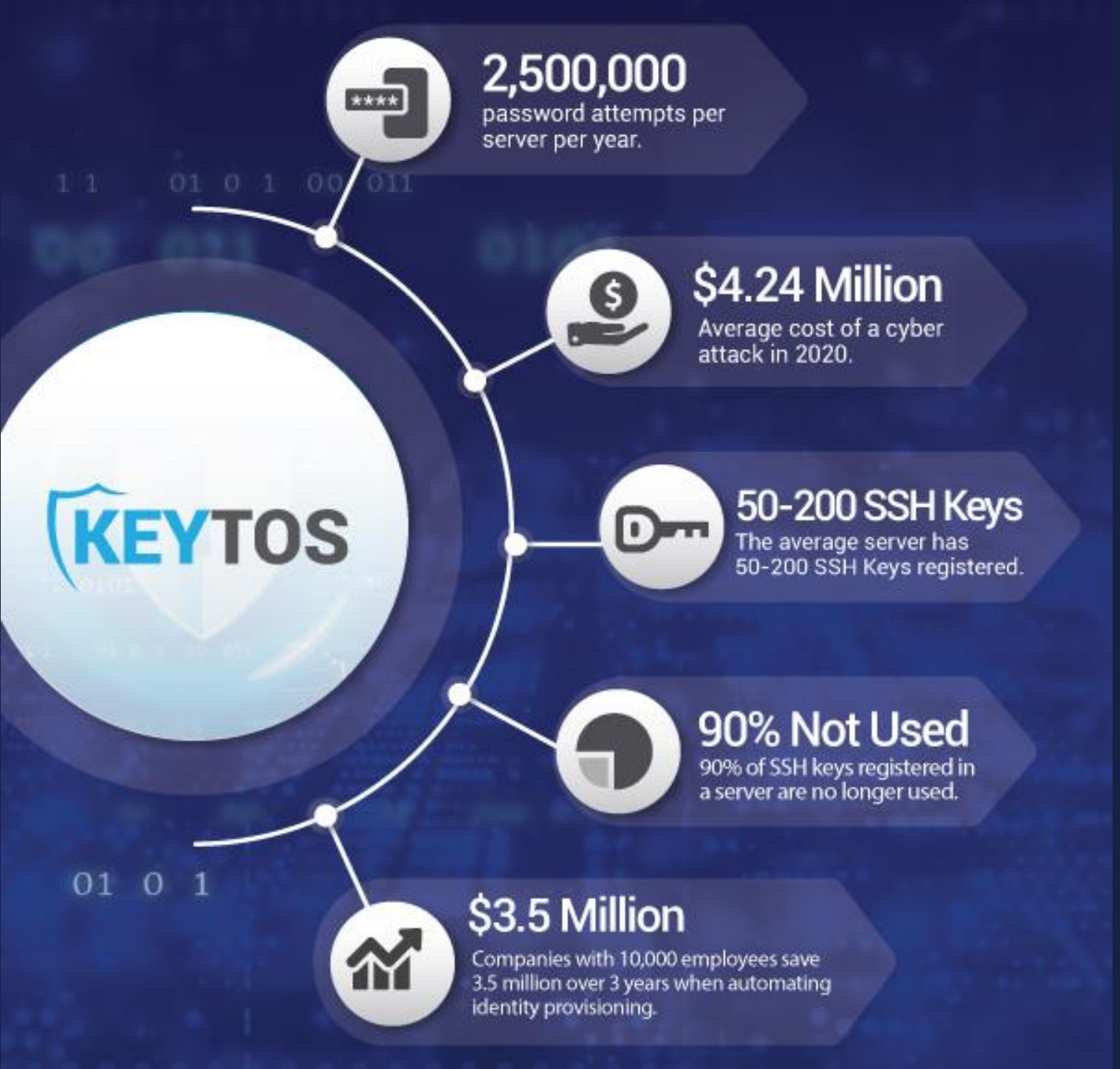


2010 SSH Certificates introduced

EZSSH

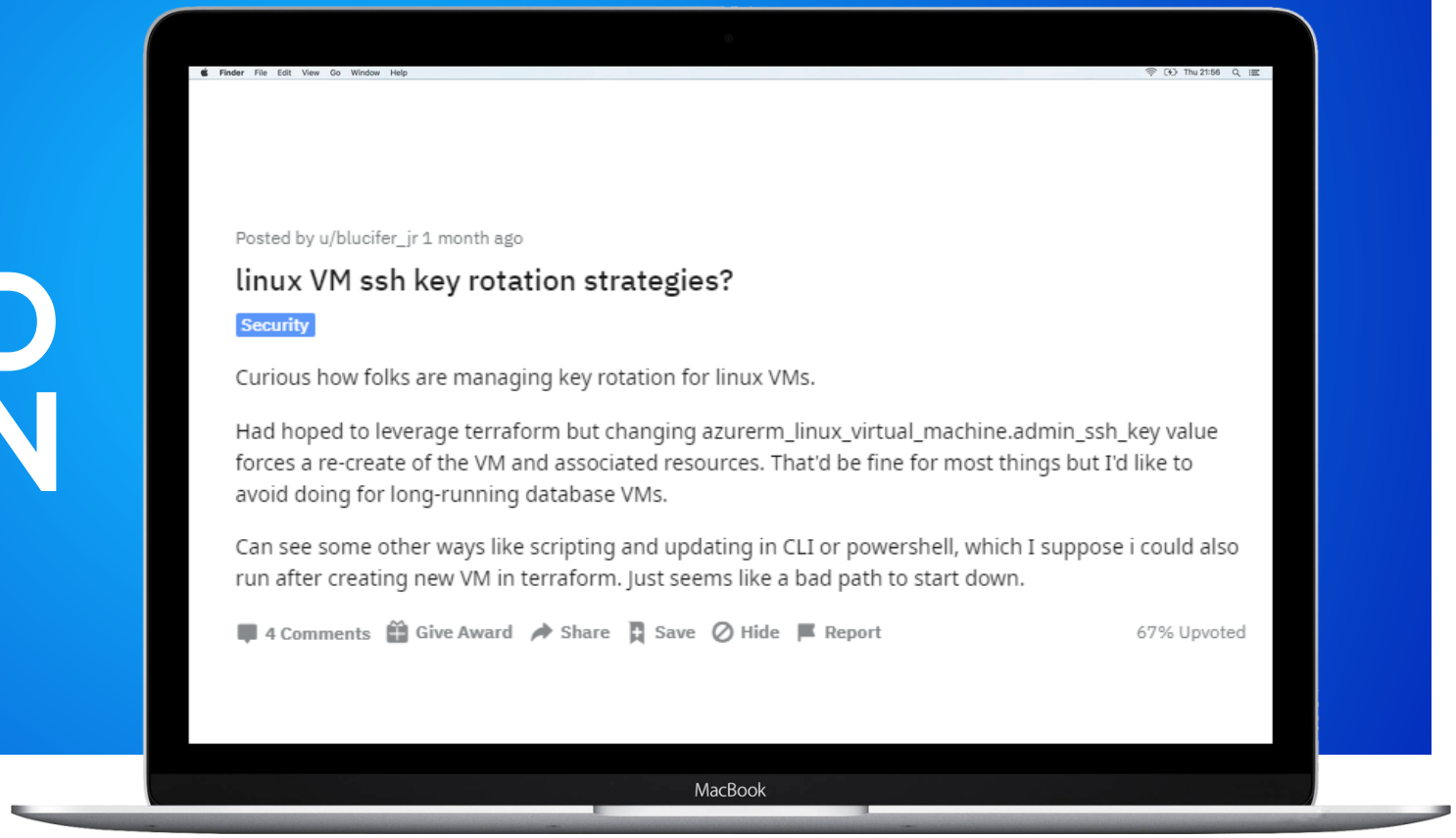
2021 EZSSH Makes SSH Certificates easy to use

KEYTOS



SSH Security
By the Numbers

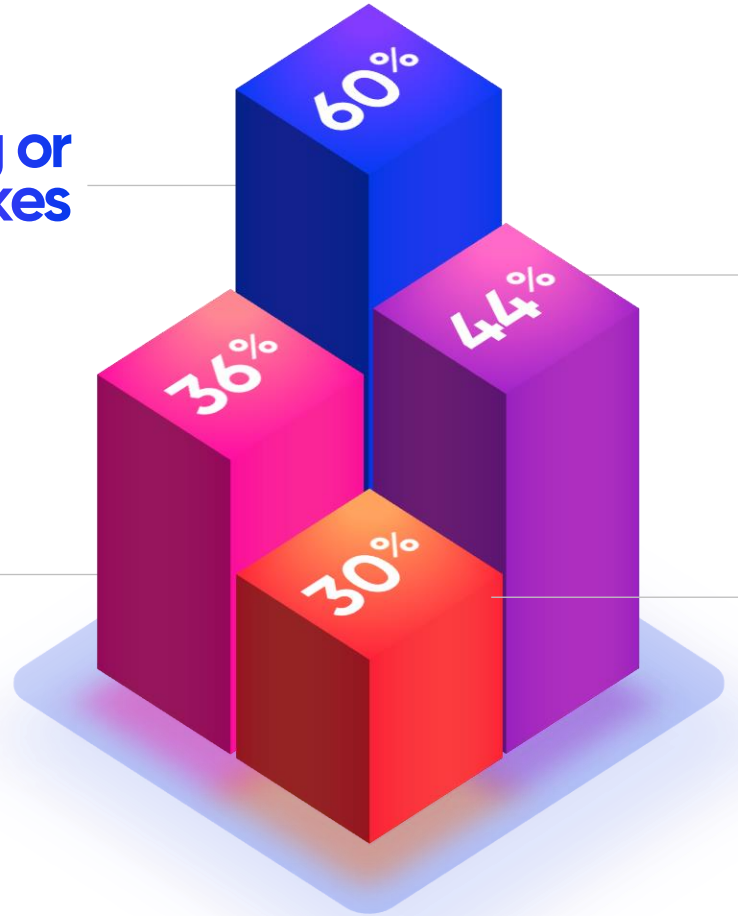
AZURE USERS NEED A SOLUTION



KEYTOS

ENGINEERS MAKE MISTAKES

The bulk of insider data breaches



Employees rushing or making mistakes

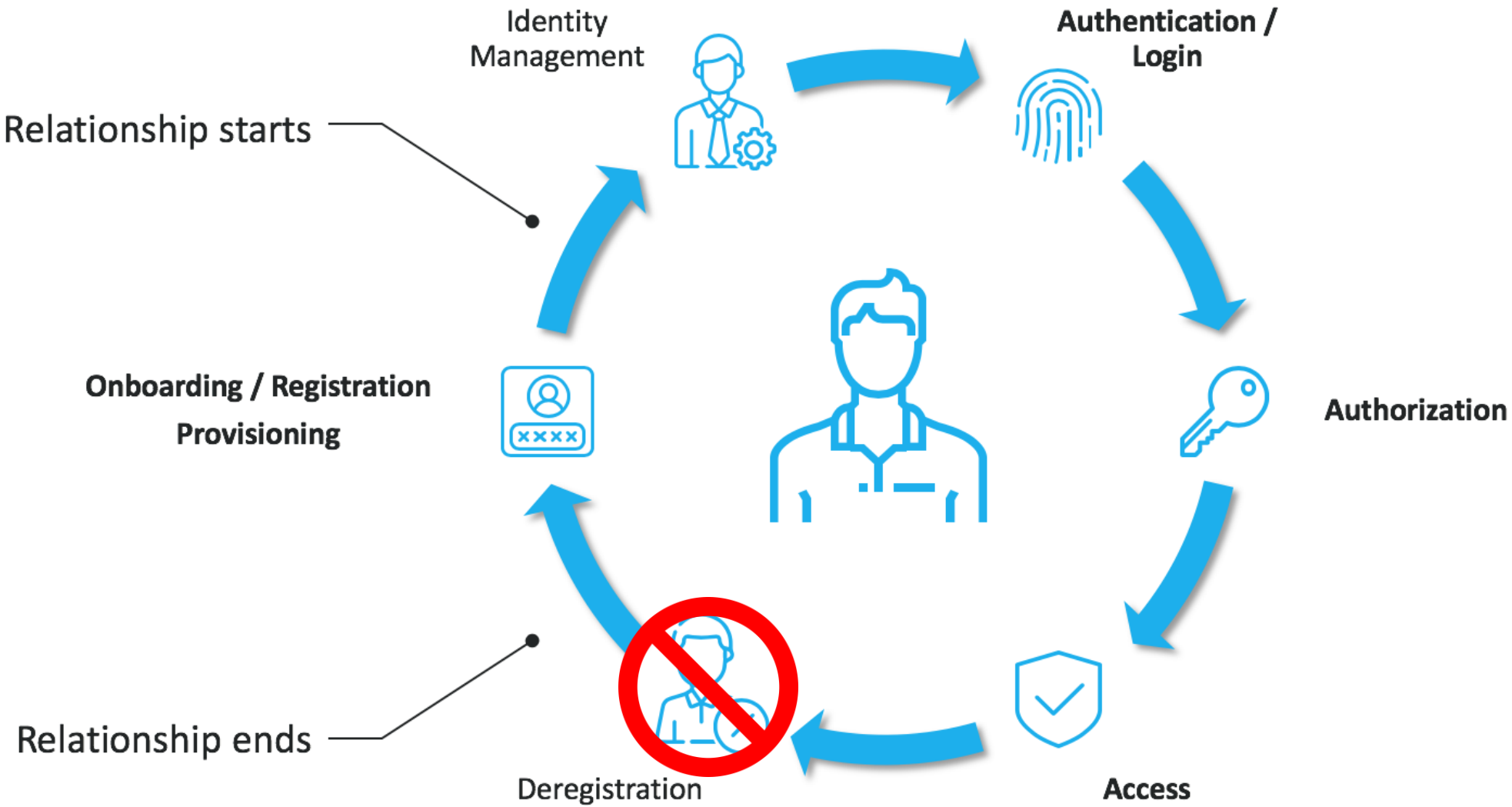
Lack of awareness

Lack of training

Intentionally leaking data

KEYTOS

IDENTITY LIFE CYCLE



KEYTOS

SSH UNDER ATTACK

- https://blog.ssh.com/ssh-key-scan-attack-honeypot
- https://www.zdnet.com/article/linux-under-attack-compromised-ssh-keys-lead-to-rootkit/
- https://securityaffairs.co/wordpress/37459/cyber-crime/compromised-ssh-keys.html
- https://www.beckershospitalreview.com/cybersecurity/github-leaks-exposed-up-to-200-000-medical-records-4-details.html
- https://thehackernews.com/2021/08/how-companies-can-protect-themselves.html
- https://www.lightreading.com/security/t-mobile-admits-breach-after-epic-hacking-claims/d/d-id/771524

2021 had the highest average cost in 17 years

Data breach costs rose from USD 3.86 million to USD 4.24 million, the highest average total cost in the 17-year history of this report.

Remote work due to COVID-19 increased cost

The average cost was USD 1.07 million higher in breaches where remote work was a factor in causing the breach, compared to those where remote work was not a factor.

Compromised credentials caused the most breaches

The most common initial attack vector, compromised credentials, was responsible for 20% of breaches at an average breach cost of USD 4.37 million.

Security AI had the biggest cost-mitigating effect

Automation and security artificial intelligence (AI), when fully deployed, provided the biggest cost mitigation, up to USD 3.81 million less than organizations without it.

A zero trust approach helped reduce cost

The average cost of a breach was USD 1.76 million less at organizations with a mature zero trust approach, compared to organizations without zero trust.

Cloud migration impacted costs and containment

Organizations further along in their cloud modernization strategy contained the breach on average 77 days faster than those in the early stage of their modernization journey.

OTHER TOOLS

Tool Name	How It Works	Key Drawbacks
Thycotic Secret Server	It is a shared password manager that allows teams to centralize their password manager.	- Requires an admin account with password to run as a high privilege user to rotate the passwords and keys.
Hashicorp Vault	Hashicorp vault is a vault service that allows you to store and create secrets for your endpoints. It also has an SSH CA feature that allows you to create SSH certificates.	- While vault offers SSH Certificates that is the same tech that we use, the process for the user is still manual (they must go to vault, create the certificate and then install it on their PCs). - Vault also lacks the advance access management that EZSSH offers.
Key Factor	Key factor allows companies to centralize their SSH key management into one portal.	- Requires admin privileges to manage SSH credentials. - While key management is centralized, input from administrators is required for lifecycle credentials.

UNIQUE EZSSH FEATURES

Designed for Zero Trust (No agent or high privilege account)

Connection with Azure security tools: Networking JIT, Azure PIM, Sentinel

Transparent security for users, with easy-to-use tools

Reduce insider threat by having Just In Time Access with appropriate approval workflows.

Reduce audit costs with easy to Audit access logs

Centralized management for hybrid and multi-cloud environments

Native Linux Authentication no custom PAM module or code runs on your servers.

Bring your own Certificate Authority support

KEYTOS