
AWS
re:Invent
NOV. 28 – DEC. 2, 2022 | LAS VEGAS, NV

SEC328

Learn to create continuous detective security controls using AWS services

Esteban Hernandez (he/him)

Sr Specialist SA Manager S&C
AWS, London

Patrick Palmer (he/him)

Sr Specialist Solutions Architect S&C
AWS, Dublin

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Agenda

- Setting the scene
- Threat modeling
- Security invariants - including examples with implementation
- Interactive breakout
- Typical challenges with invariants
- Automated reasoning
- Q&A

Note: You're encouraged to ask questions at any point during the session

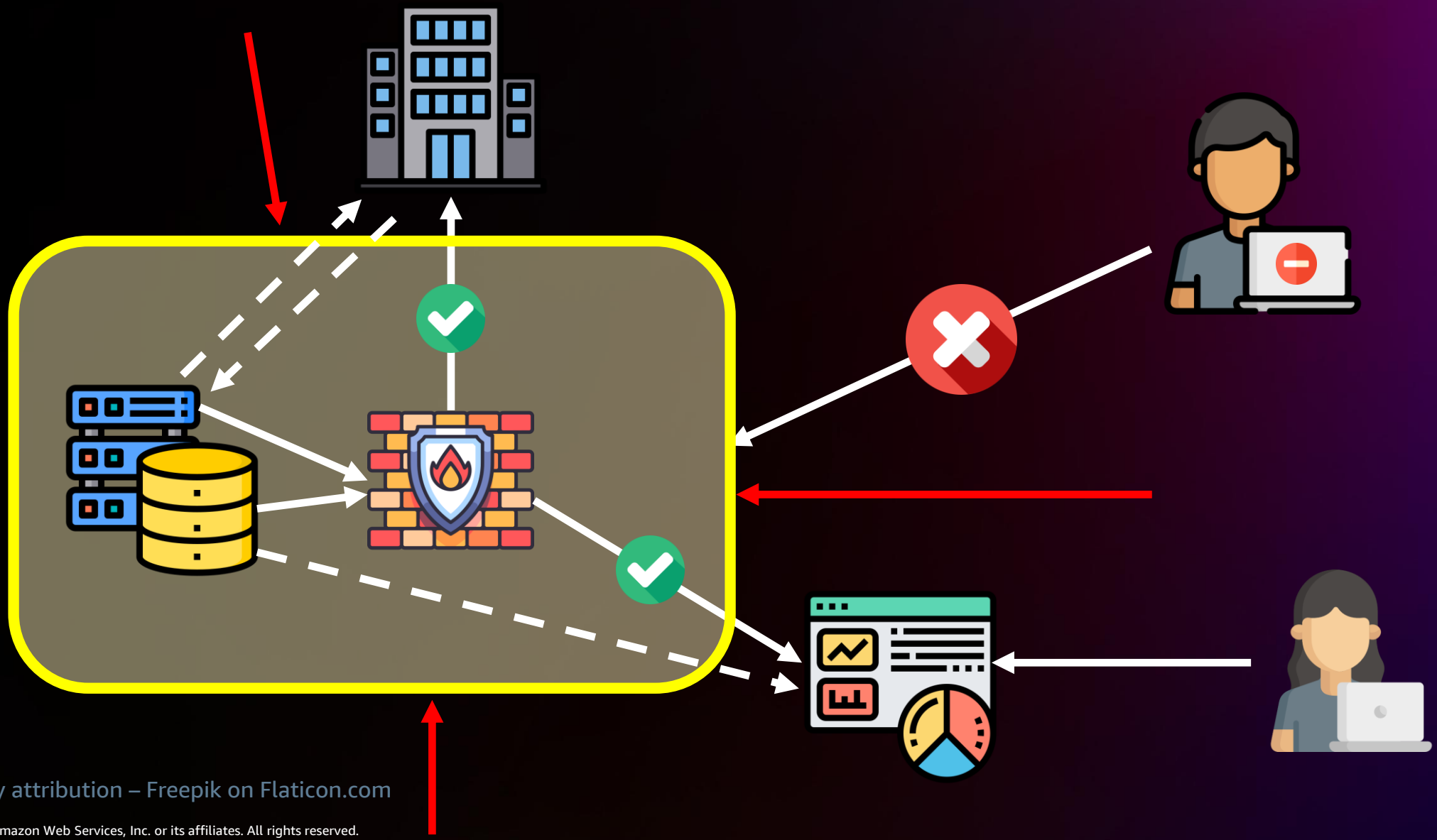
aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Security Invariants

A security invariant is a system property that relates to the system's ability to prevent security issues from happening

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Real-life scenario



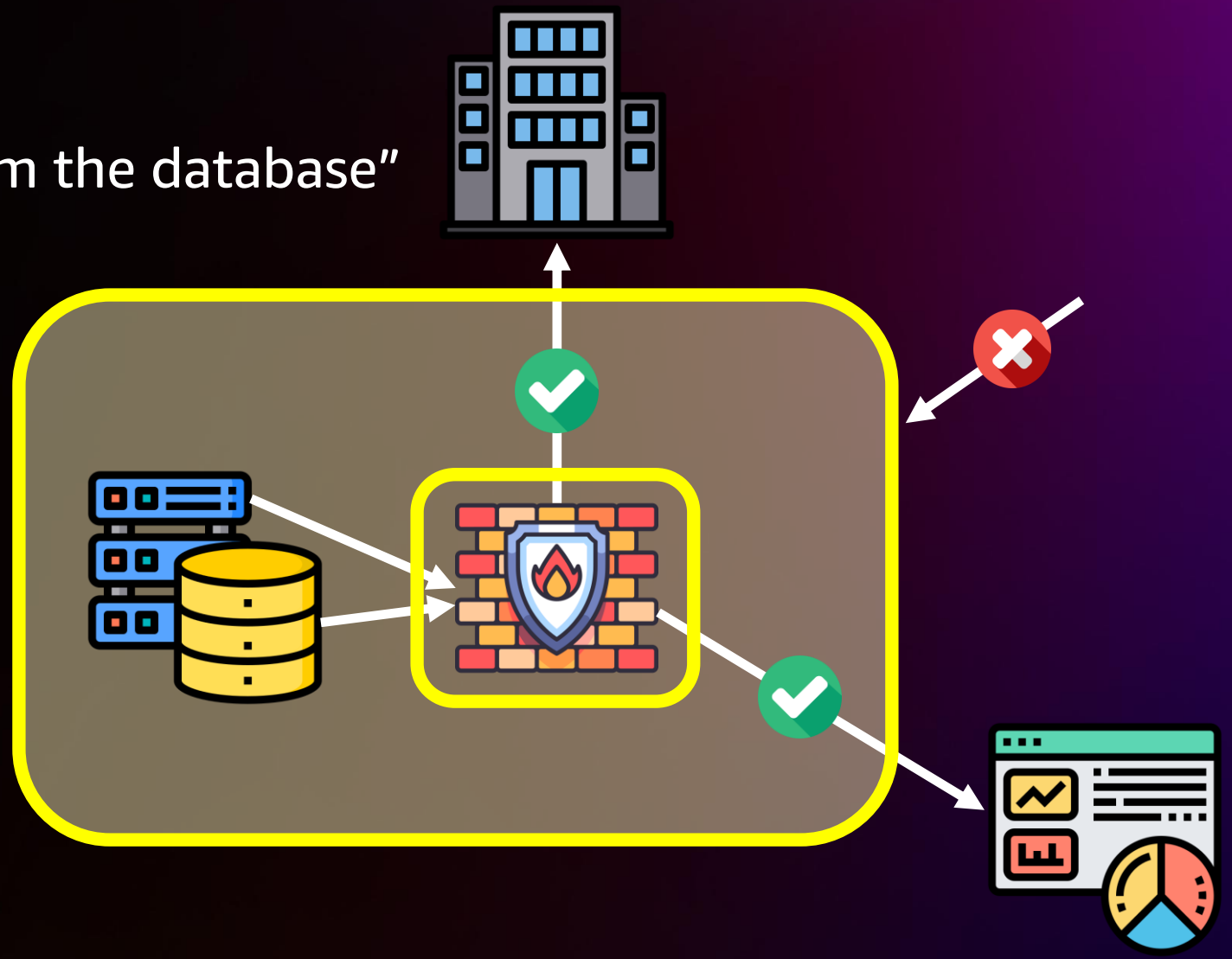
Icons: License by attribution – Freepik on Flaticon.com

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

What is the biggest risk you're trying to mitigate?

Any guesses?

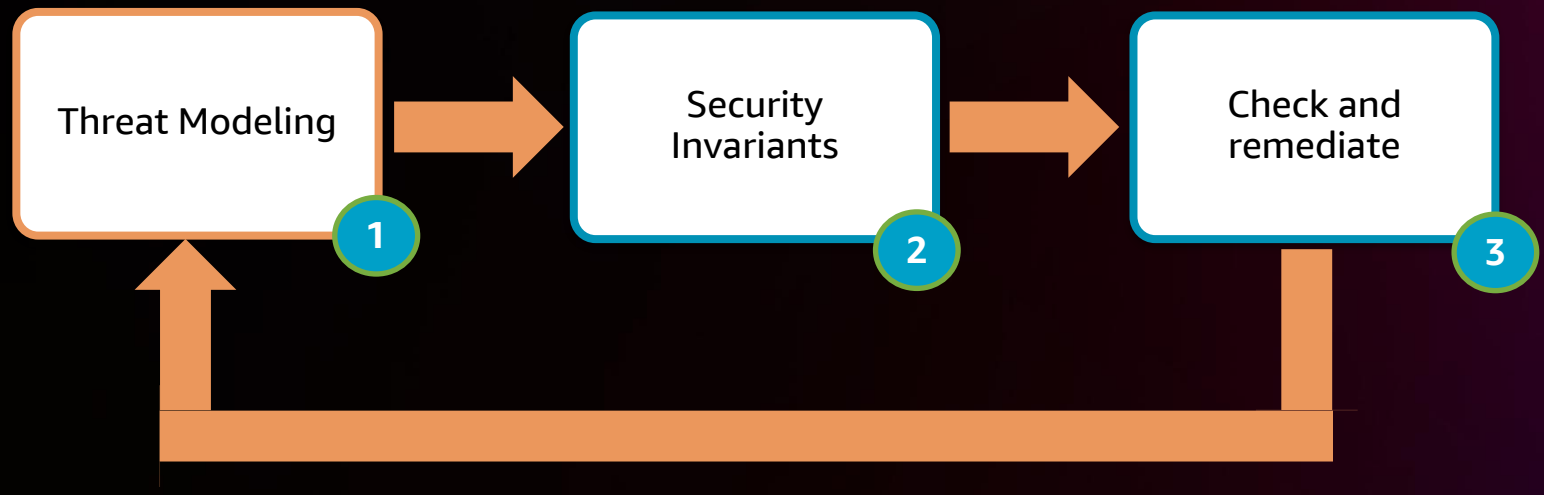
"Exfiltration of sensitive data from the database"



Icons: License by attribution – Freepik on Flaticon.com

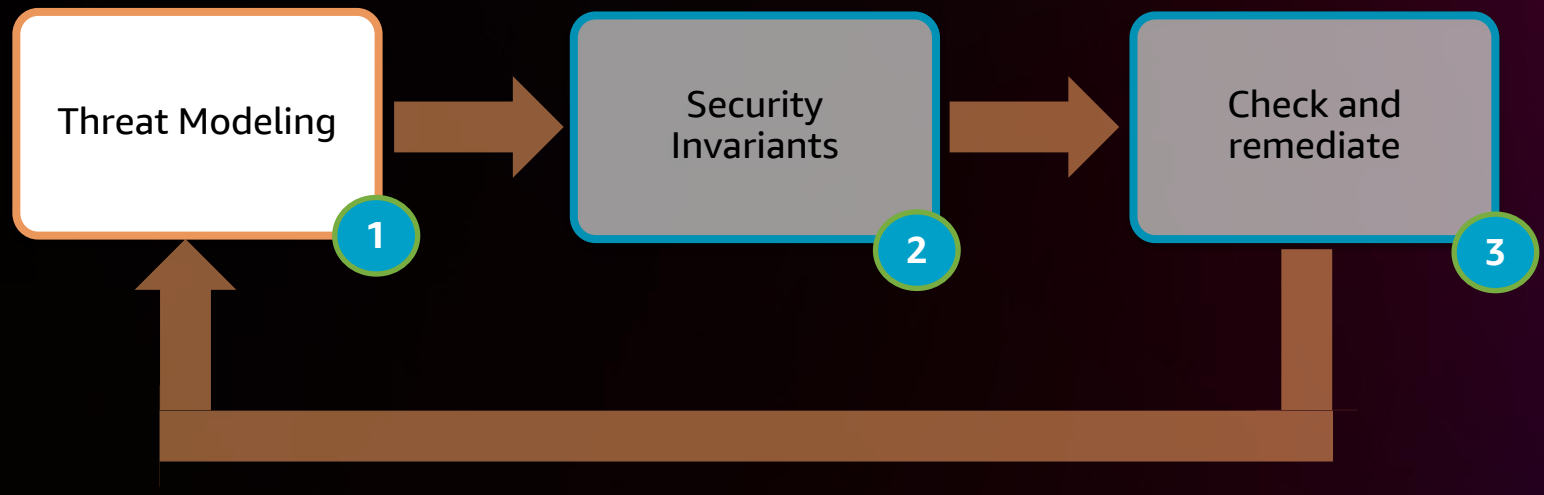
aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Security Invariant life-cycle



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Security Invariant life-cycle



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Threat modeling

- Threat modeling is a mental exercise for identifying threats for the system you are building

A threat model answers four questions*:

What are we building?	What can go wrong?	What can we do about it?	Did we do a good enough job (for now)?

* Courtesy of Adam Shostack: https://github.com/adamshostack/4QuestionFrame

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Threat modeling: real-life example

What are we building?

- A system which has sensitive third-party data stored in an enclave
- The system allowed to fetch additional data (pull)
- It generates a report with aggregated, non-sensitive data, for users to consume

What can go wrong?

- Exfiltration of sensitive data from enclave via unauthorized user

What can we do about it?

- Configure egress filtering to only allow traffic to destinations we trust

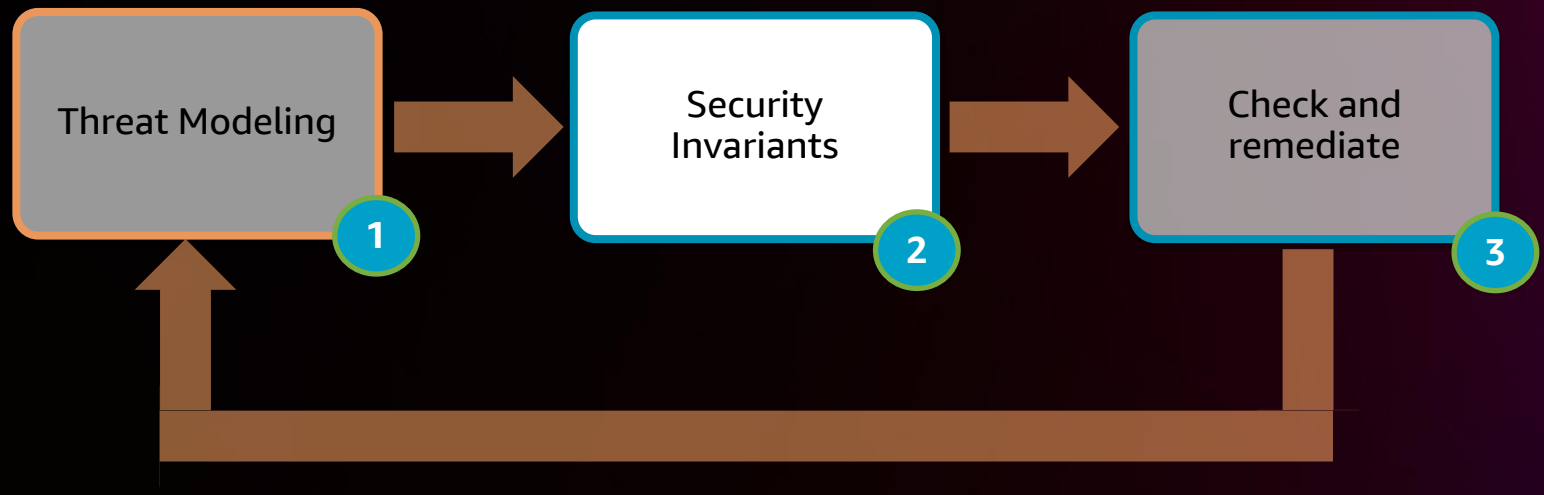
Did we do a good enough job (for now)?

- Needs to be tested and revised as changes go through

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Security Invariant life-cycle



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Security Invariants

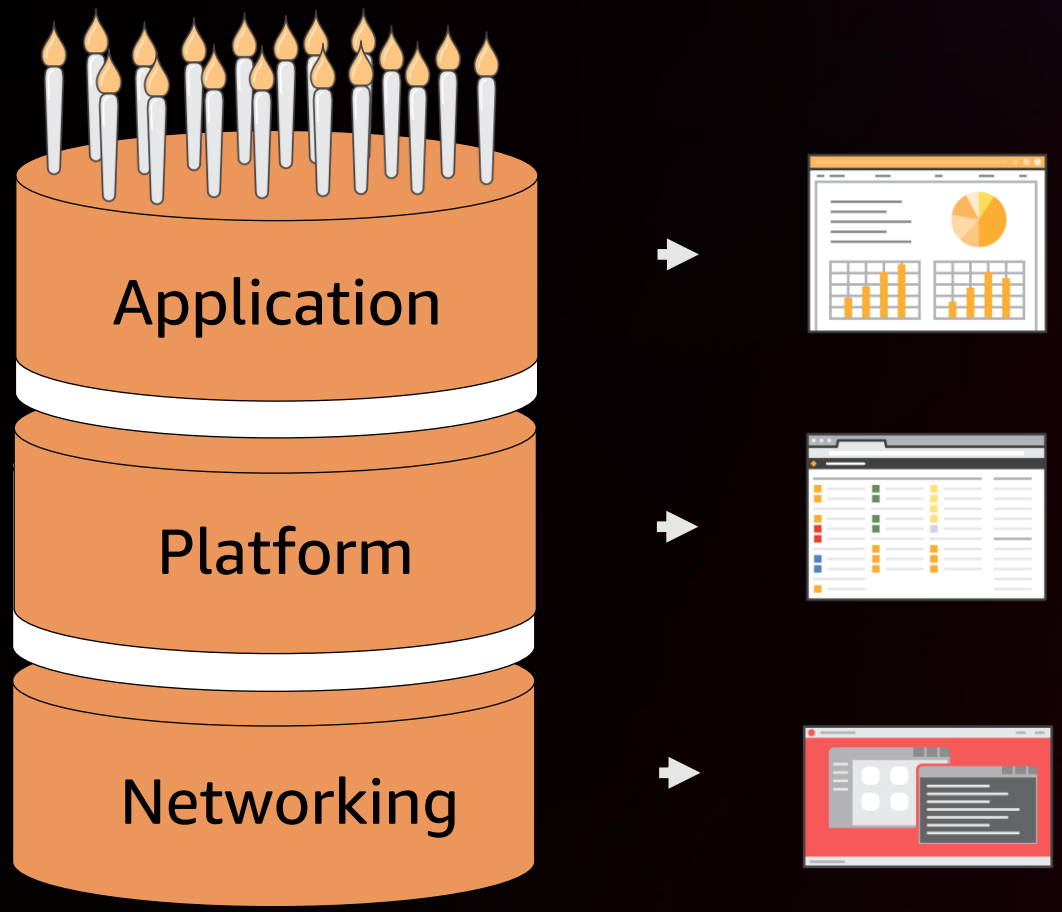
A security invariant is a system property that relates to the system's ability to prevent security issues from happening

- Example 1: Egress filtering only allows traffic to destinations we trust
- Example 2: To prevent an on-path unauthorized user from intercepting HTTP requests, our API endpoints are only available over TLS

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Security Invariants – end to end



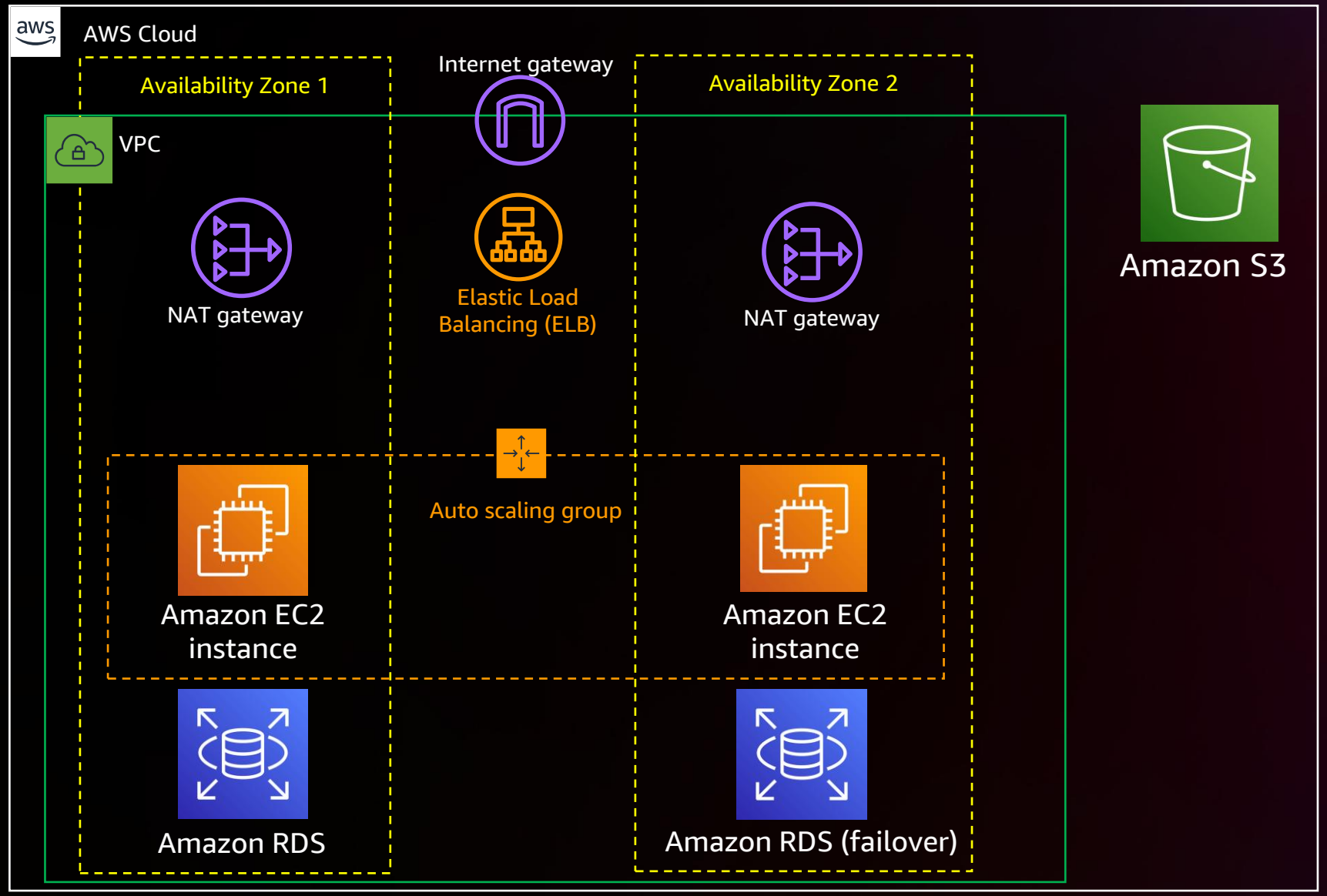
End-user applications

Infrastructure and services

Routing and networking

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Use Case – example 2



Three-tier web application

Allows users to upload and download files via their browser

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Security Invariant - Networking

Threat: If there is a path from the internet to the database, an unauthorized user could run queries to extract sensitive information.

Invariant: The database is not directly accessible through the public internet.

Check: AWS Config rules (for NACL + security groups) and AWS Firewall Manager.

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Security Invariant - Platform

Threat: The S3 bucket content becomes publicly available to unauthenticated users.

Invariant: The S3 bucket does not allow public readability.

Check and remediate: Multiple options available. Apply block public access at account level and monitor for changes to the configuration of block public access.

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Security Invariant - Application

Threat: Users can access the files of other users if requests are not authorized properly.

Invariant: Only owners and users with explicit access to a file have access to it.

Check and remediate: Run CloudWatch synthetics canaries. If a canary fails, create ticket and page on-call engineer.

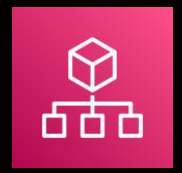
aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Typical AWS Services used to check invariants



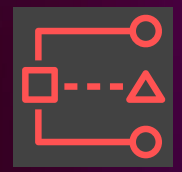
AWS Firewall Manager



Service Control Policies (SCPs)



Amazon CloudWatch Synthetics



AWS IAM Access Analyzer



AWS Network Firewall



Resolver DNS Firewall



Amazon VPC Network Access Analyzer



AWS Security Hub



AWS Identity and Access Management (IAM)



AWS Key Management Service (AWS KMS)



AWS Config



AWS Lambda

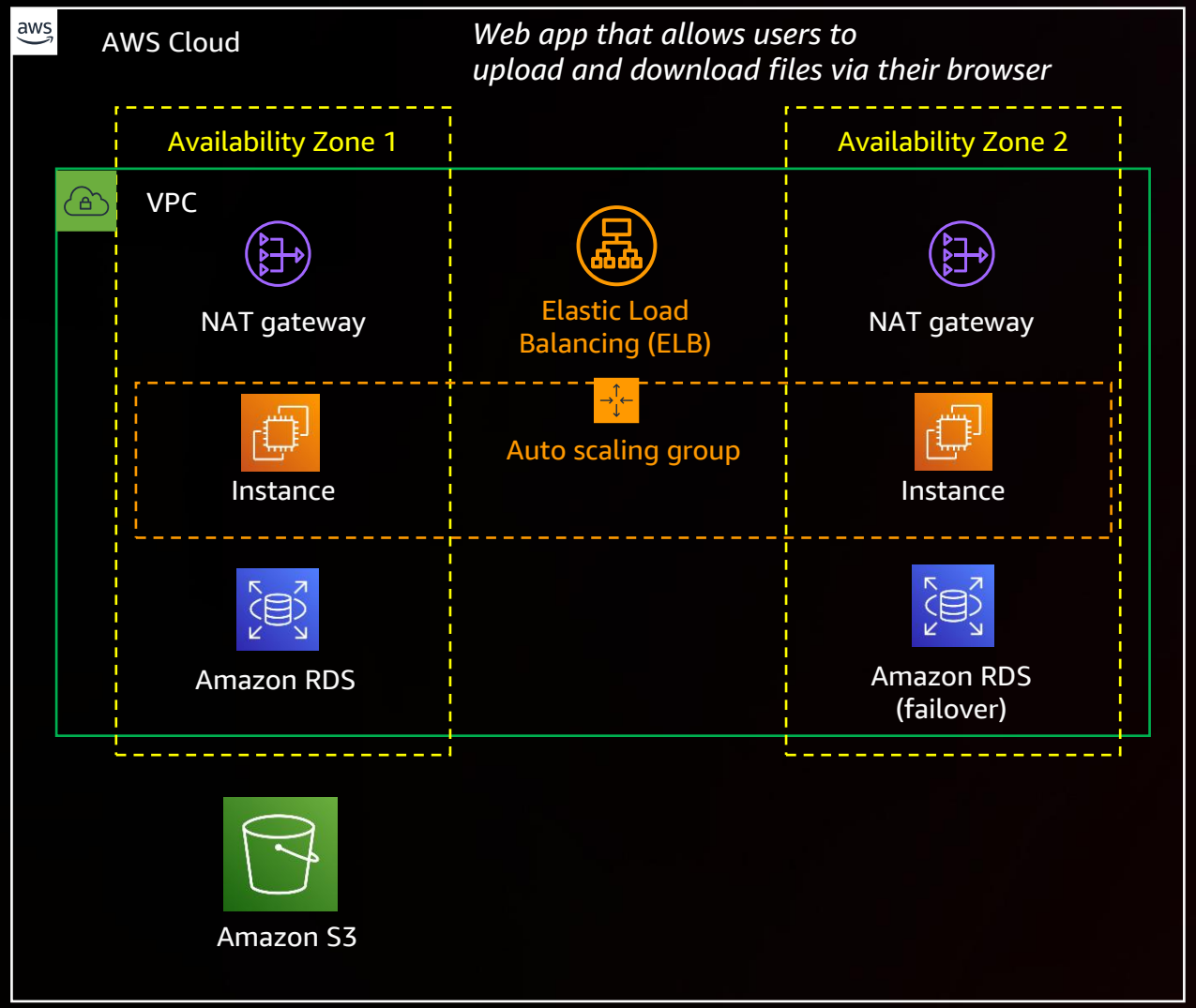
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Which invariants should I implement?

… it depends

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Breakout use case



Step 1: Pick one or two threats

Bonus points if you can come up with your own threat(s)

Step 2: For each threat, define one or two invariants

Step 3: For each invariant, try to come up with a way to check them, and remediate if they are breached

Feel free to be creative - add services/functionalities. We will discuss examples together in 15 minutes

Example list of threats:

1. Insufficient input validation could allow cross-site scripting
2. A sustained DDoS renders the application unavailable
3. Logs can be tampered with to prevent detection of unauthorized actions

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Discussion

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Typical challenges

- What do you think?
- How can you be sure you have covered all the scenarios for your invariants?
- Define where to execute your invariant checks
- There could be overlap between invariants
- Who owns the invariant?

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Provable security

- Better than exhaustive testing
- Uses advanced mathematics and logic to prove that systems are operating as intended
- Formalizes intended protocol and code requirements by translating to mathematical specifications
- Systems are tested to make sure they align to mathematical specifications
- Example: Network Access Analyzer

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Thank you!

Patrick Palmer
/patrick-palmer-aws

Esteban Hernandez
/estebanhernandez



Please complete the session survey in the **mobile app**

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.