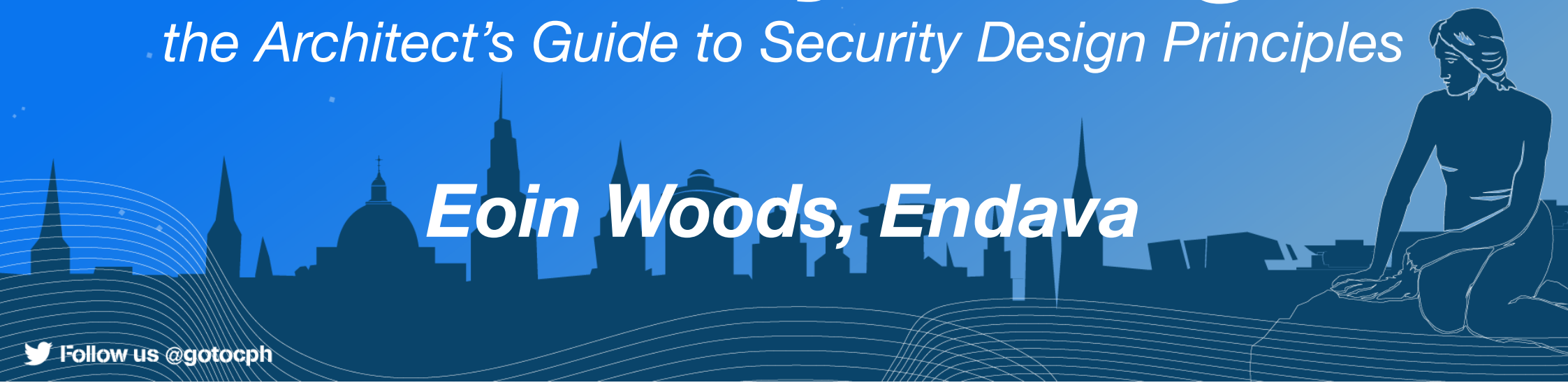
goto; copenhagen

GOTO Copenhagen 2017
Conference Oct. 1-3, 2017

Secure by Design

the Architect's Guide to Security Design Principles



Eoin Woods, Endava

Follow us @gotocph

endava

BACKGROUND

- **Eoin Woods**
  - **CTO at Endava** (technology services, ~4000 people)
  - **10 years in product development** - Bull, Sybase, InterTrust
  - **10 years in capital markets applications** - UBS and BGI
  - Software dev engineer, then architect, now CTO
- Author, editor, speaker, community guy

endava

CONTENT

- **What is security** and why do we care?
- What are **design principles**, why are they **useful**?
- **Security design principles**
  - 10 important principles useful in practice

endava

REVISITING SECURITY

- We all know security is important - but **why**?
  - protection against **malice**, **mistakes** and **mischance**
  - theft, fraud, destruction, disruption
- Security is a **risk management** business
  - **loss** of time, money, privacy, reputation, advantage
  - **insurance model** - balance costs against risk of loss

endava

ASPECTS OF SECURITY PRACTICE

Secure Application Design

Secure Application Implementation

Secure Infrastructure Design

Secure Infrastructure Deployment

Secure System Operation

endava

SECURITY DESIGN PRINCIPLES

What is a "**principle**" ?

*a fundamental **truth or proposition** serving as the foundation for **belief or action*** [OED]

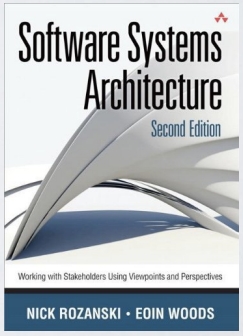
We define a **security design principle** as ….

*a declarative **statement** made with the intention of **guiding security design decisions** in order to meet the goals of a system*

endava

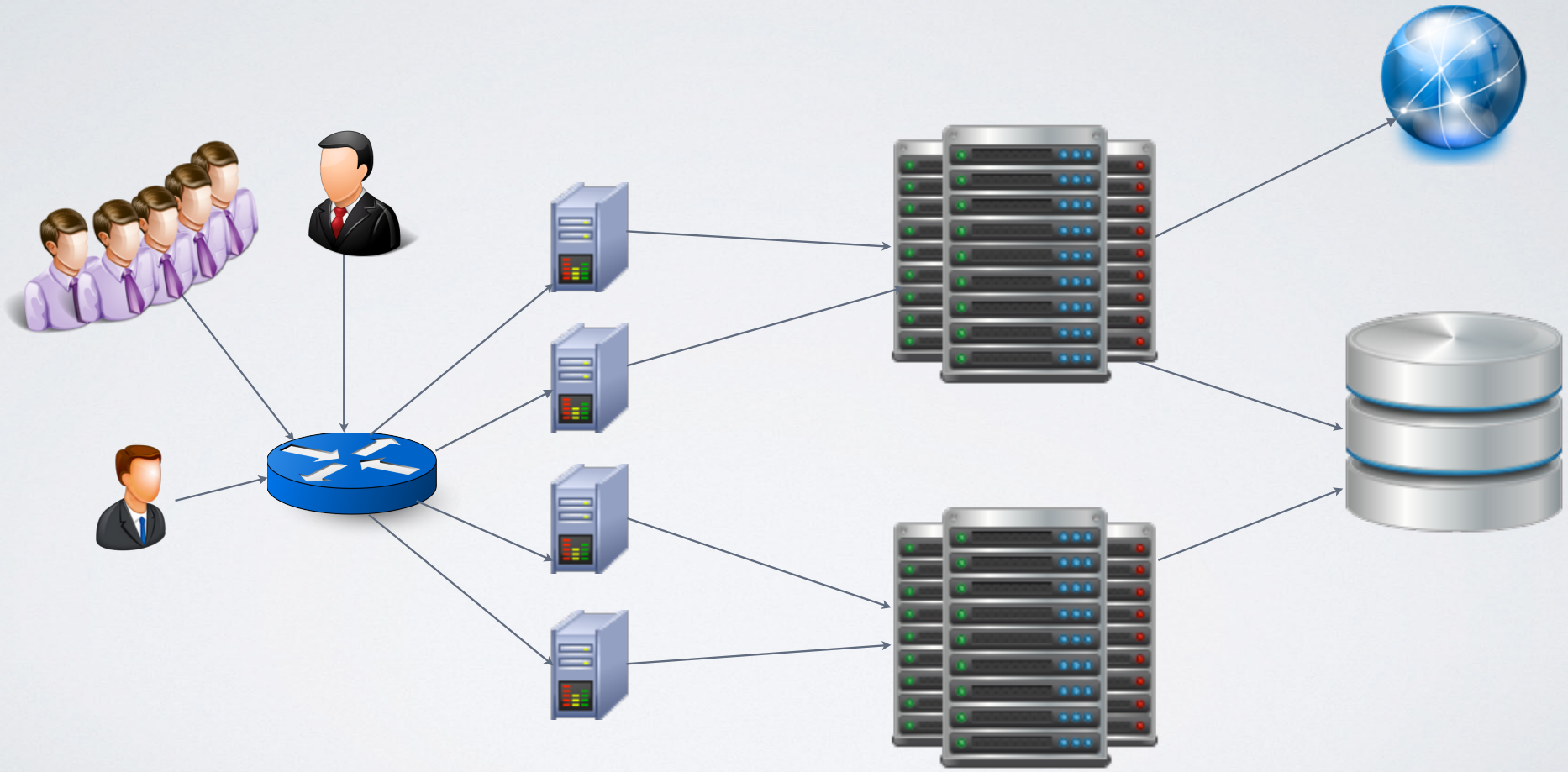
SECURITY DESIGN PRINCIPLES

- There are **many sets** of security design principles
  - Viega & McGraw (10), OWASP (10), NIST (33), NCSC (44), Cliff Berg (185) …
  - Many similarities between them at fundamental level
- I have distilled **10 key principles** as a basic set
  - these are brief summaries for slide presentation
  - www.viewpoints-and-perspectives.info



endava

A SYSTEM TO BE SECURED



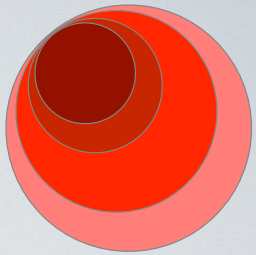
endava

TEN KEY SECURITY PRINCIPLES

- Assign the **least privilege** possible
- Separate **responsibilities**
- **Trust cautiously**
- **Simplest** solution possible
- **Audit** sensitive events
- **Fail securely** & use **secure defaults**
- Never rely upon **obscurity**
- Implement **defence in depth**
- **Never invent** security technology
- Find the **weakest link**

endava

1- LEAST PRIVILEGE



Why?	Broad privileges allow malicious or accidental access to protected resources
Principle	Limit privileges to the minimum for the context
Tradeoff	Less convenient; less efficient; more complexity
Example	Run server processes as their own users with exactly the set of privileges they require

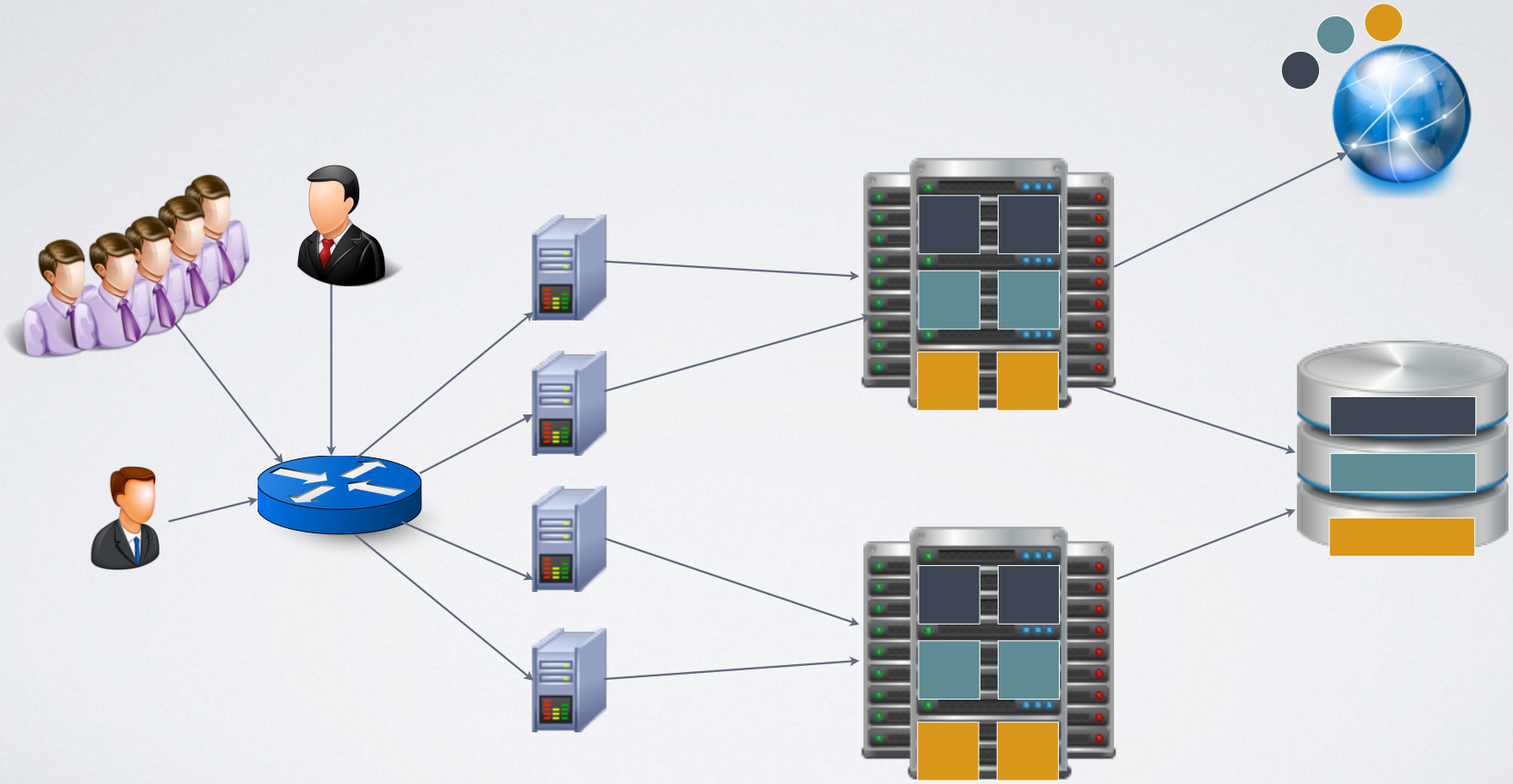
endava

2 - SEPARATE RESPONSIBILITIES

Why?	Achieve control and accountability, limit the impact of successful attacks, make attacks less attractive
Principle	Separate and compartmentalise responsibilities and privileges
Tradeoff	Development and testing costs; operational complexity: troubleshooting more difficult
Example	"Payments" module administrators have no access to or control over "Orders" module features

endava

2 - SEPARATE RESPONSIBILITIES



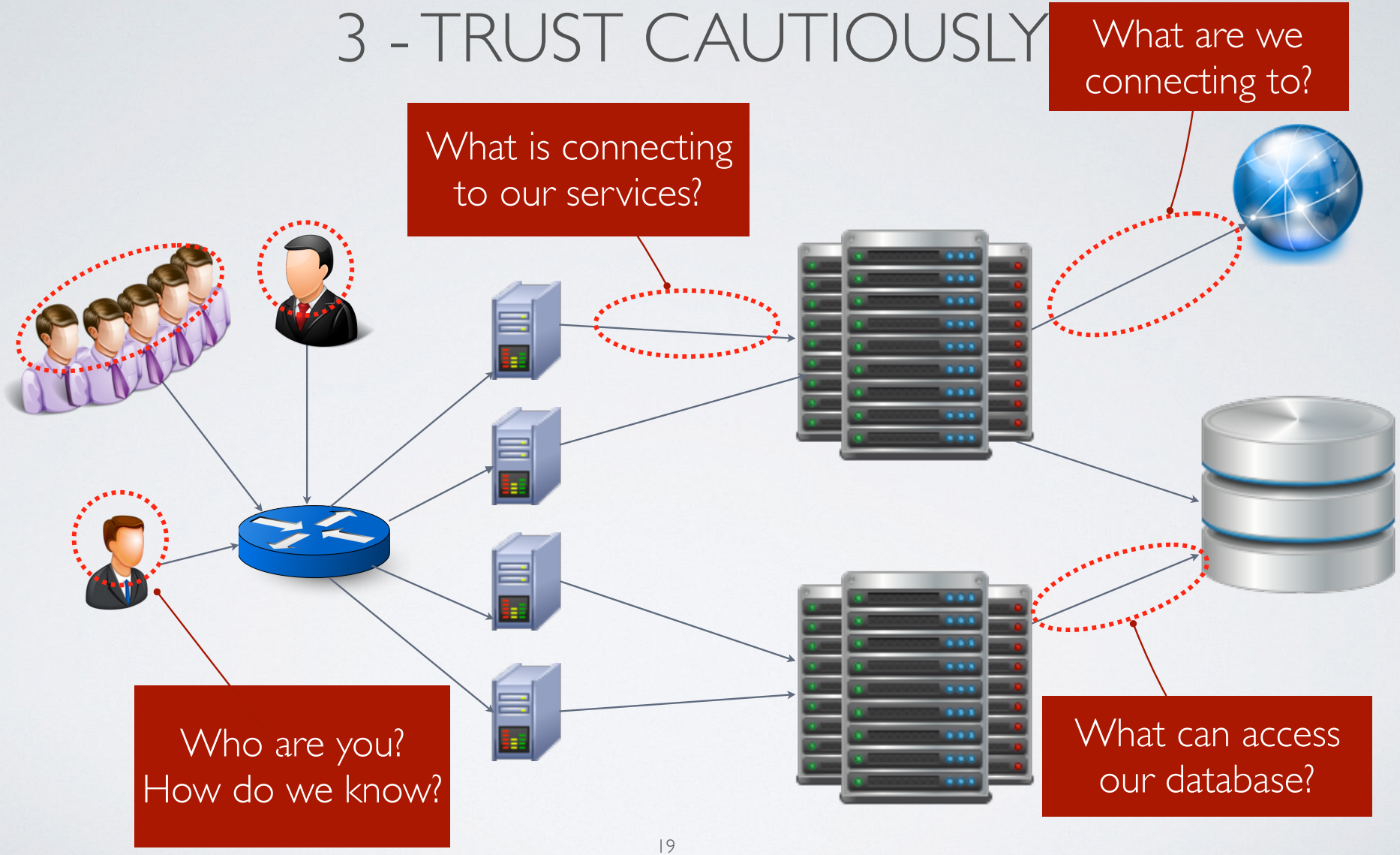
endava

3- TRUST CAUTIOUSLY

Why?	Many security problems caused by inserting malicious intermediaries in communication paths
Principle	Assume unknown entities are untrusted, have a clear process to establish trust, validate who is connecting
Tradeoff	Operational complexity (particularly failure recovery); reliability; some development overhead
Example	Don't accept untrusted RMI connections, use client certificates, credentials or network controls

endava

3 - TRUST CAUTIOUSLY



endava

4- SIMPLEST SOLUTION POSSIBLE

*The **price** of **reliability** is the pursuit of the utmost **simplicity** - **C.A.R. Hoare***

Why?	Security requires understanding of the design - complexity rarely understood - simplicity allows analysis
Principle	Actively design for simplicity - avoid complex failure modes, implicit behaviour, unnecessary features, …
Tradeoff	Hard decisions on features and sophistication; Needs serious design effort to be simple
Example	Does the system really need dynamic runtime configuration via a custom DSL?

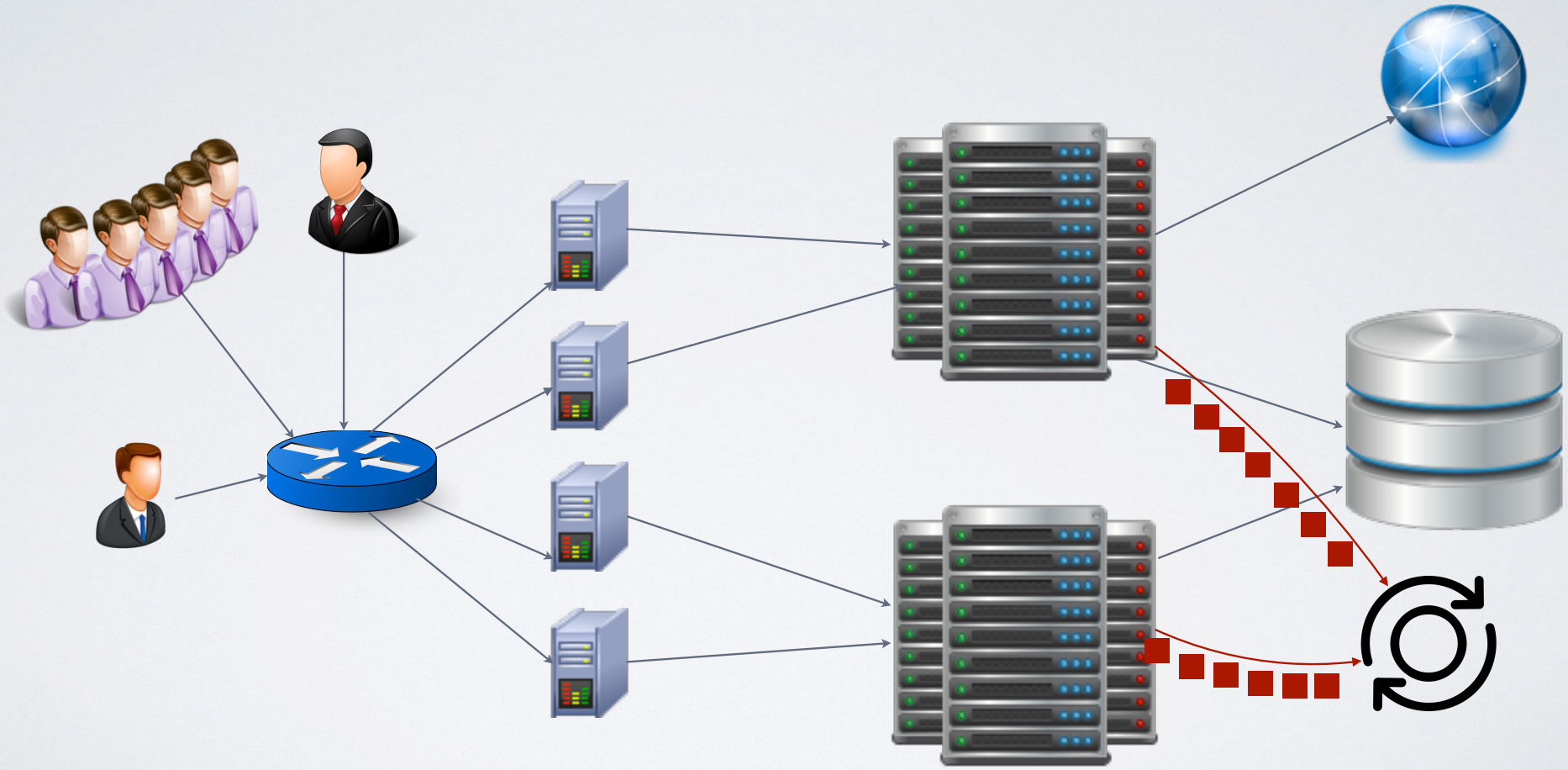
endava

5 - AUDIT SENSITIVE EVENTS

Why?	Provide record of activity, deter wrong doing, provide a log to reconstruct the past, provide a monitoring point
Principle	Record all security significant events in a tamper-resistant store
Tradeoff	Performance; operational complexity; dev cost
Example	Record changes to "core" business entities in an append-only store with (user, ip, timestamp, entity, event)

endava

5 - AUDIT SENSITIVE EVENTS



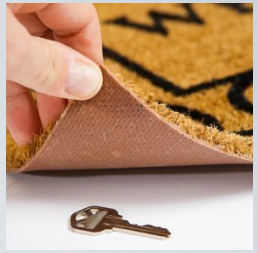
endava

6 - SECURE DEFAULTS & FAIL SECURELY

Why?	Default passwords, ports & rules are "open doors" Failure and restart states often default to "insecure"
Principle	Force changes to security sensitive parameters Think through failures - to be secure but recoverable
Tradeoff	Convenience
Example	Don't allow "`SYSTEM/MANAGER`" after installation On failure don't disable or reset security controls

endava

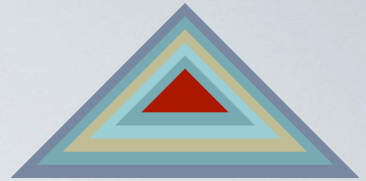
7 - NEVER RELY ON OBSCURITY



Why?	Hiding things is difficult - someone is going to find them, accidentally if not on purpose
Principle	Assume attacker with perfect knowledge, this forces secure system design
Tradeoff	Designing a truly secure system takes time and effort
Example	Assume an attacker will guess a "port knock" network request sequence or a password obfuscation technique

endava

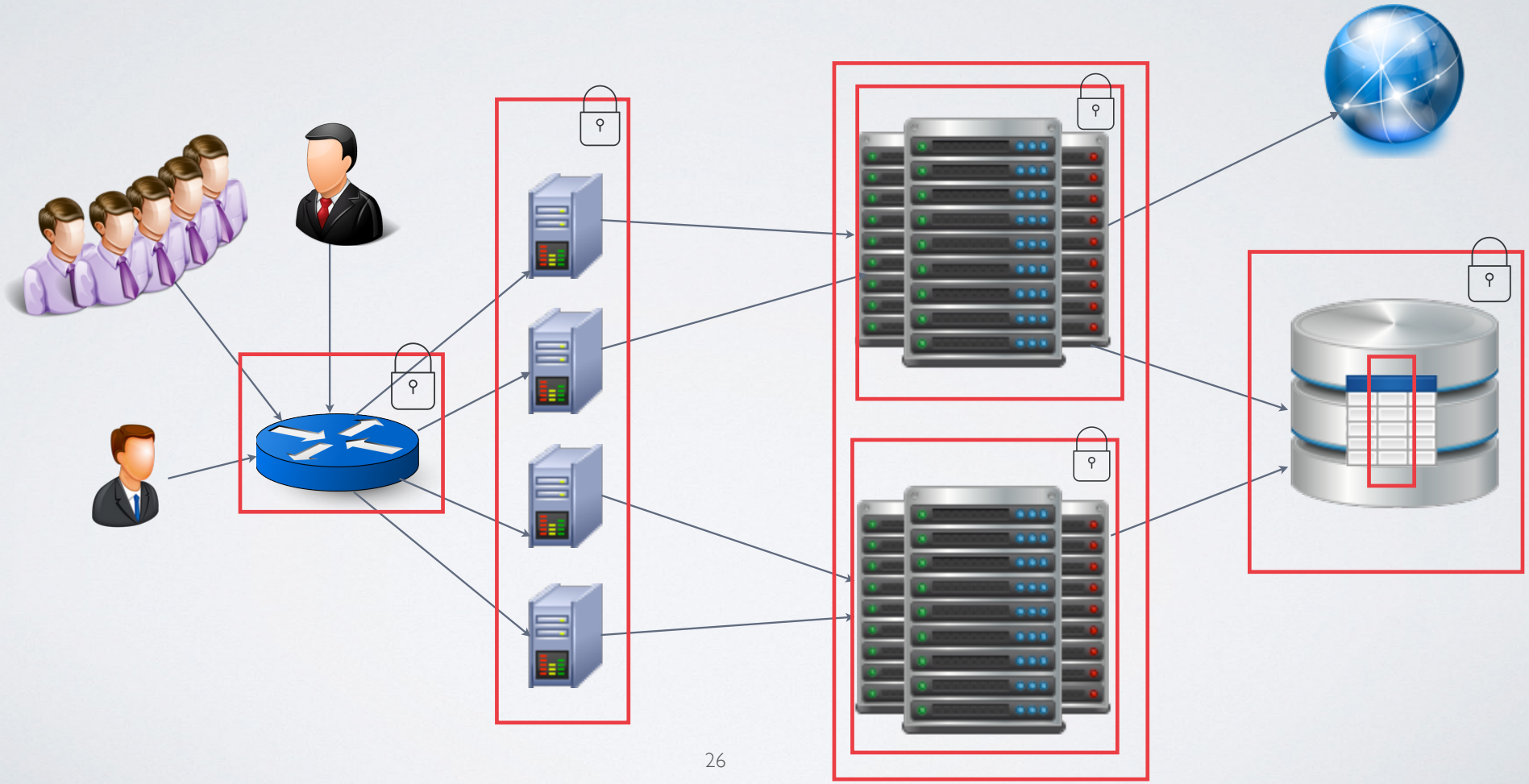
8 - DEFENCE IN DEPTH



Why?	Systems do get attacked, breaches do happen, mistakes are made - need to minimise impact
Principle	Don't rely on single point of security, secure every level, stop failures at one level propagating
Tradeoff	Redundancy of policy; complex permissioning and troubleshooting; can make recovery difficult
Example	Access control in UI, services, database, OS

endava

8 - DEFENCE IN DEPTH



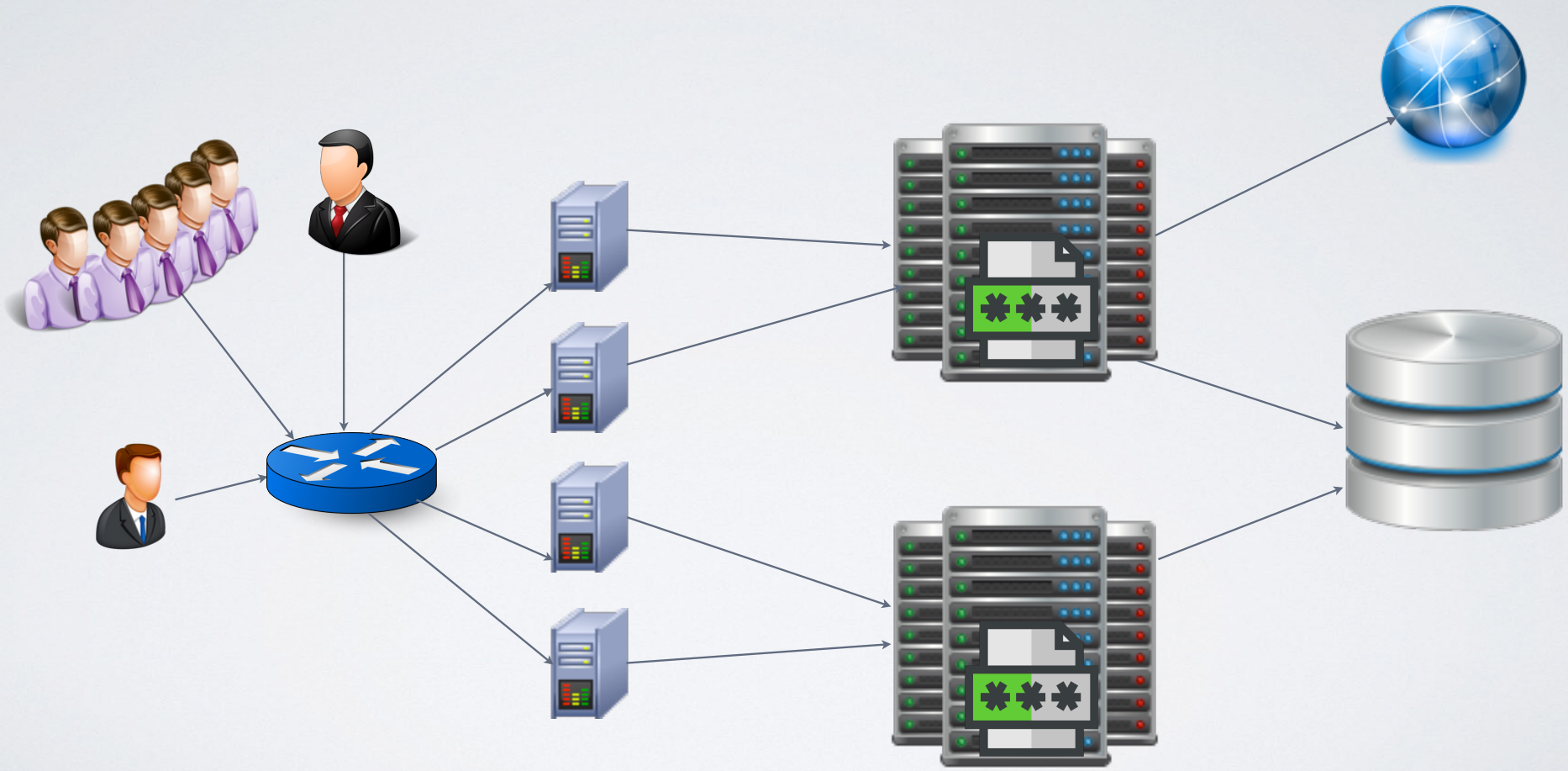
endava

9 - NEVER INVENT SECURITY TECH

Why?	Security technology is difficult to create - avoiding vulnerabilities is difficult
Principle	Don't create your own security technology - always use a proven component
Tradeoff	Time to assess security technology; effort to learn it; complexity
Example	Don't invent your own SSO mechanism, secret storage or crypto libraries … choose proven components

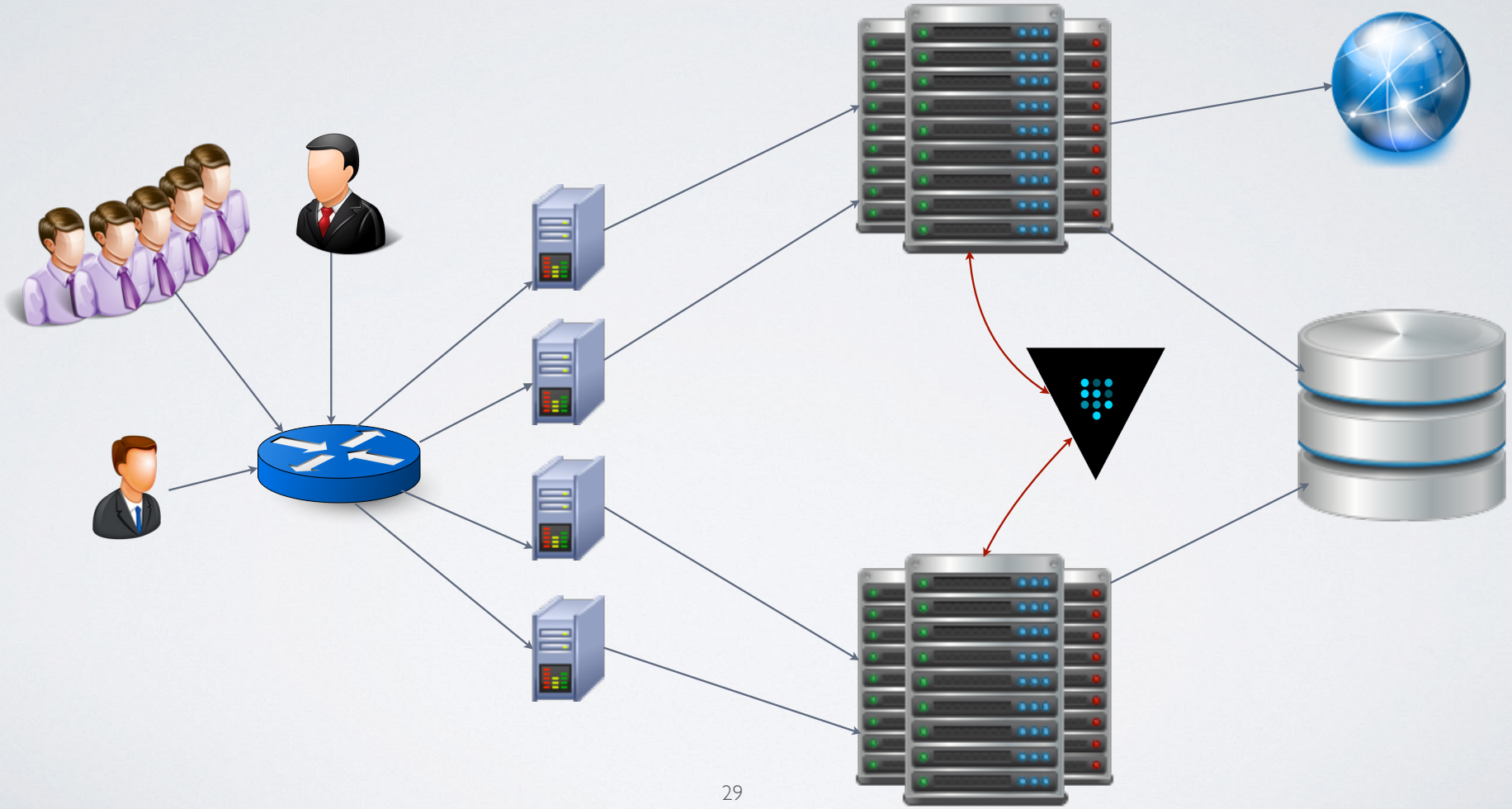
endava

9 - NEVER INVENT SECURITY TECHNOLOGY



endava

9 - NEVER INVENT SECURITY TECHNOLOGY



endava

10 - SECURE THE WEAKEST LINK

Why?	"Paper Wall" problem - common when focus is on technologies not threats
Principle	Find the weakest link in the security chain and strengthen it - repeat! (Threat modelling)
Tradeoff	Significant effort required; often reveals problems at the least convenient moment!
Example	Data privacy threat => encrypted communication but with unencrypted database storage and backups

endava

TEN KEY SECURITY PRINCIPLES

- Assign the **least privilege** possible
- Separate **responsibilities**
- **Trust cautiously**
- **Simplest** solution possible
- **Audit** sensitive events
- **Fail securely** & use secure defaults
- Never rely upon **obscurity**
- Implement **defence in depth**
- **Never invent** security technology
- Find the **weakest link**

endava

REFERENCES

- UK Government NCSC Security Principles:
  https://www.ncsc.gov.uk/guidance/security-design-principles-digital-services-main
- NIST Engineering Principles for IT Security:
  http://csrc.nist.gov/publications/nistpubs/800-27A/SP800-27-RevA.pdf
- Short intro to McGraw's set:
  http://www.zdnet.com/article/gary-mcgraw-10-steps-to-secure-software/
- OWASP Principles set:
  https://www.owasp.org/index.php/Category:Principle

endava

BOOKS

goto; copenhagen

Thank you … questions?

Eoin Woods
Endava
eoin.woods@endava.com
@eoinwoodz