LONDON
INTERNATIONAL
SOFTWARE DEVELOPMENT
CONFERENCE 2015

goto; conference

Rugged Building Materials and Creating Agility with Security

David Etue (@djetue)

goto; conference

Click ‘engage’ to rate session.

Rate **12** sessions to get the supercool GOTO reward

Join the conversation #gotoldn

Rugged Building Materials

goto; conference

- SecDevOps, Rugged DevOps, DevSecOps, DevOps: Whatever you want to call it, we all need security (and compliance)
- Very little security can exist without asset, configuration and change management
- If we write good code, choose our components wisely, and manage it well, what else is left?

“Security” Holding Up DevOps Deployments

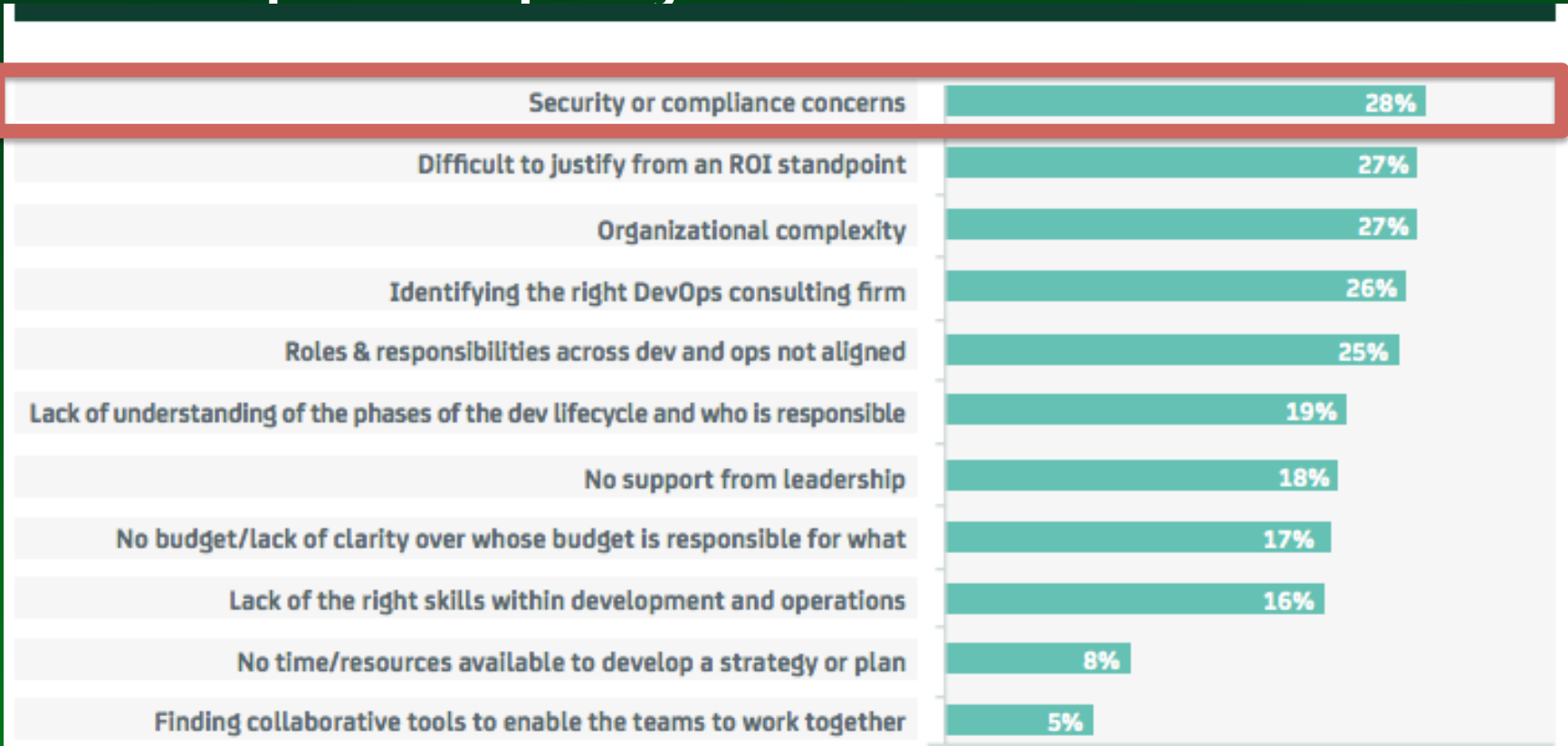
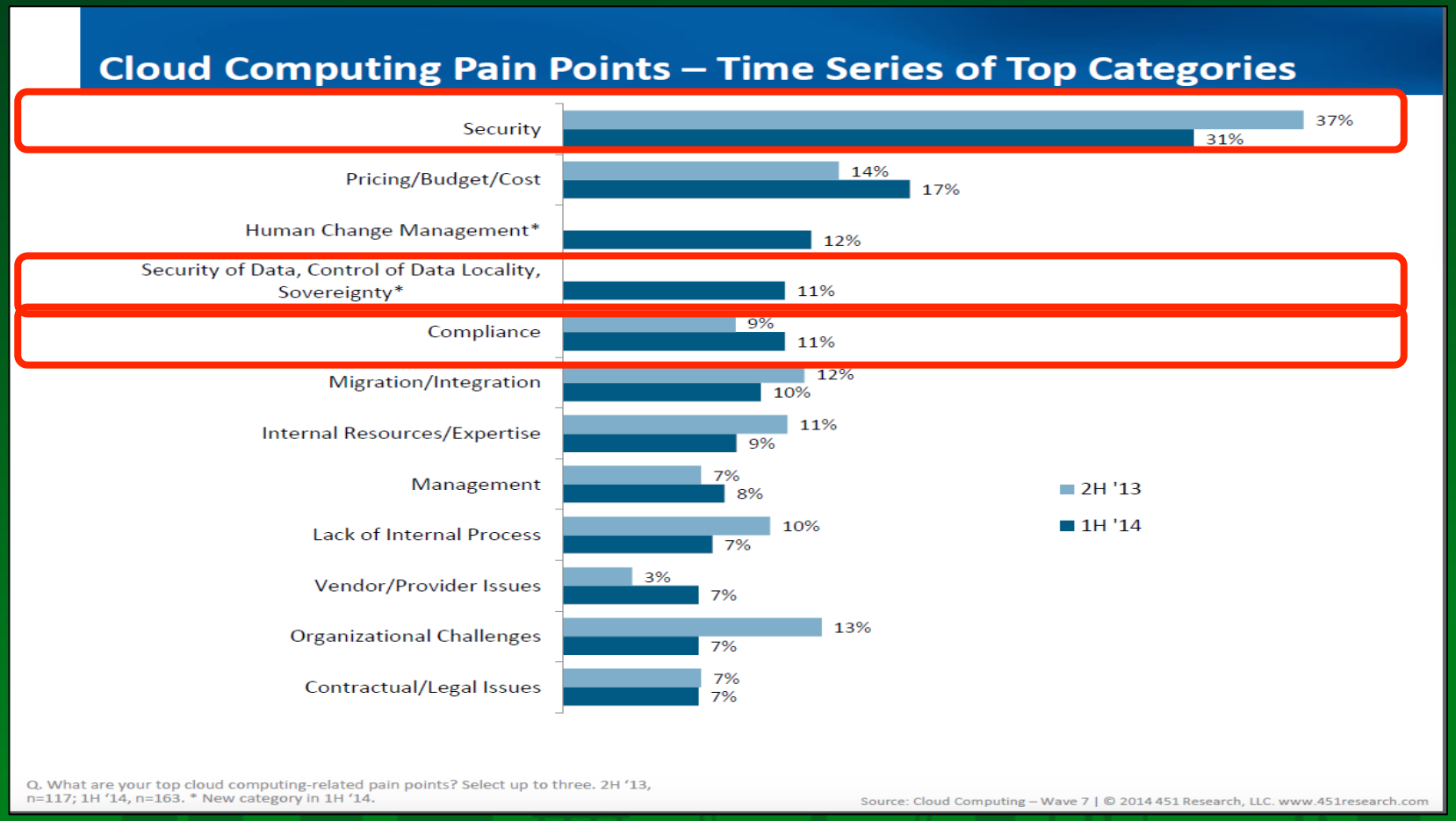


Figure 6.
What are the major obstacles to implementing a DevOps strategy in your organization? Total: 1,425

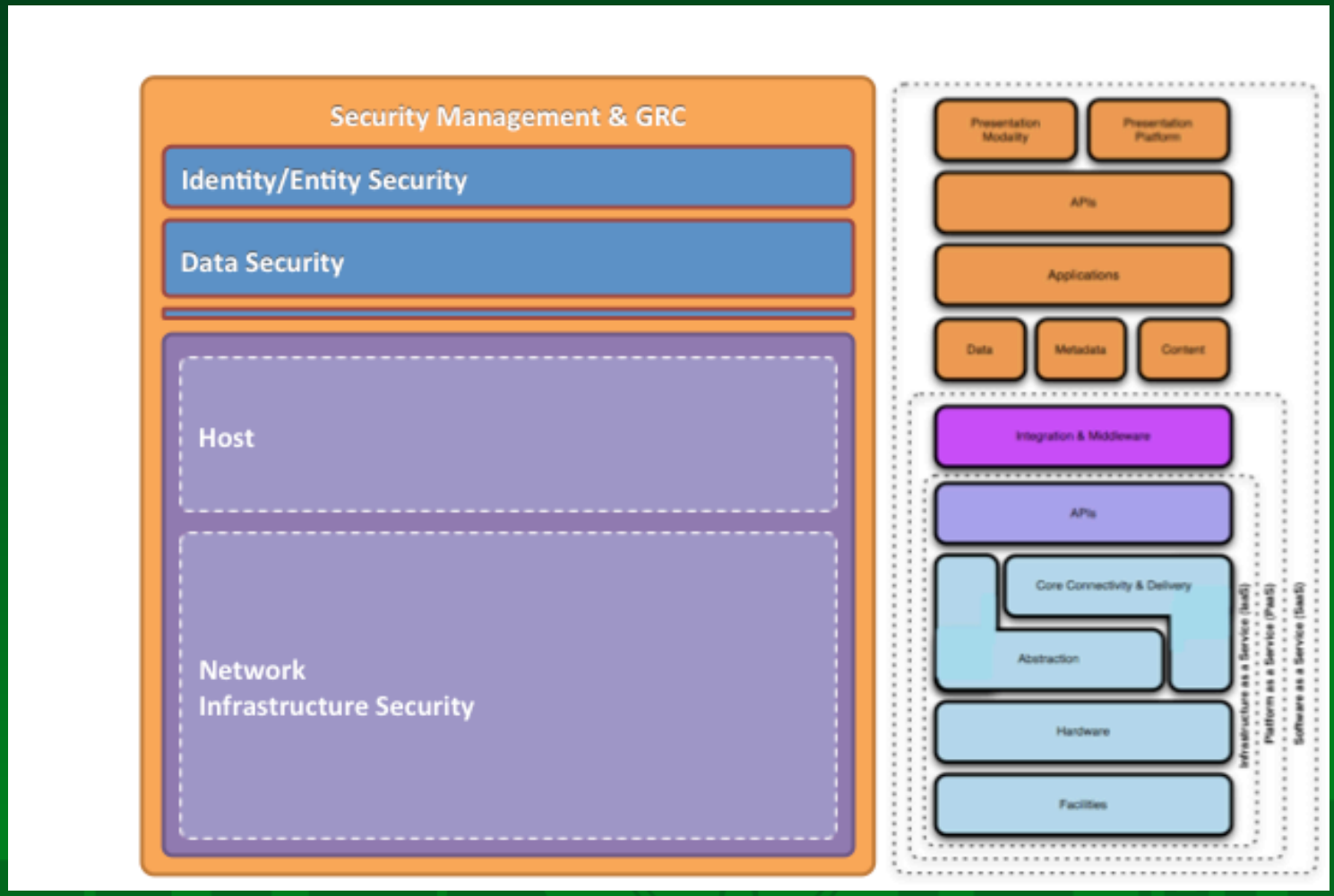
DevOps: The Worst-Kept Secret to Winning in the Application Economy by CA Technologies, October 2014

Security Struggling With Cloud Too…



451 Research - Cloud Computing Wave 7

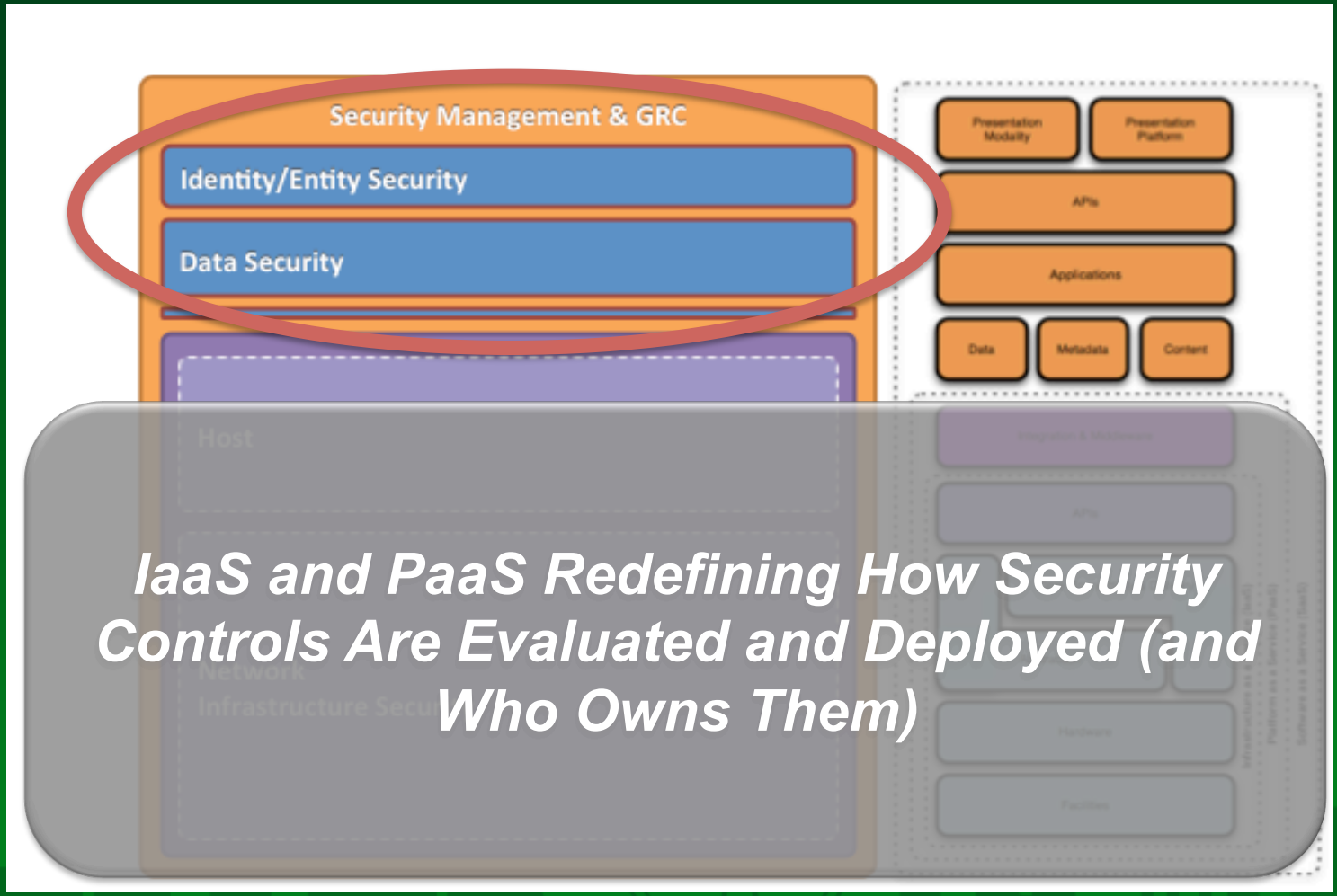
Traditional Security Controls Don't Map Well to Cloud and DevOps



Source: Control Quotient: Adaptive Strategies For Gracefully Losing Control (RSA US 2013) by Josh Corman and David Etue.

Microservices, Agility and Portability Require Focus “Up The Stack”

goto; conference



Security Wants Automation Too…

…They just might not know it yet

- DevOps wants security to be:
  - Orchestrable
  - API-driven
  - Automatically assessed
  - Portable
  - Risk-based / appropriate

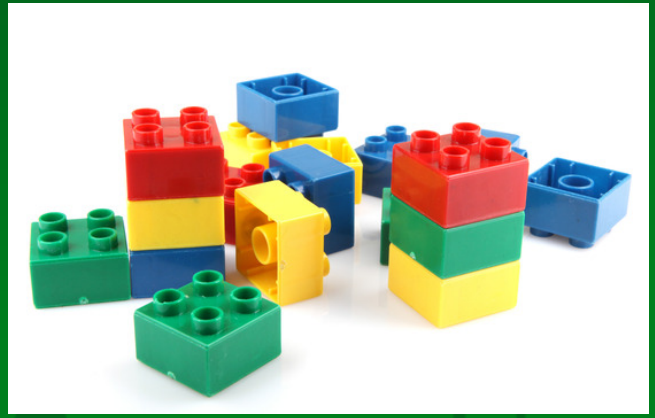
Big Gap Between Desired State and Security Solutions "As Code"

- Security wants:
  - Security closer to the data
  - Lower cost of Compliance
  - Analyst productivity
  - Better inventory / asset management
  - More uniformity
  - Faster updates (and patches)
  - Not to be "Dr No"

Core Security Building Blocks

goto; conference

- Identity to determine who (or what) did (or failed to do) something
- Controls on what privilege users and privilege infrastructure (code) can do
  - Separation of duties
  - Least privilege
- Encryption as a tool to separate data (and secrets) from inappropriate access
  - Privilege Users (internal)
  - Privilege Users (cloud / service provider)
  - Government Agencies
  - Adversaries
- Logging and Auditing to enable:
  - Granular what, where, when, and how (and sometimes why)
  - Demonstration of compliance
  - Incident response



Identity

- Lots of solutions for humans
  - IAM, PIM/PAM, Cloud IAM, etc.
  - APIs and Provisioning becoming a key platform feature
  - Key focus: He/She Who Can Deploy (or Un-deploy) is god…
- Less solutions for systems, services, processes and things, but evolving
  - UUIDs (or similar) matter
  - Automation means infrastructure and code becoming “privileged”

Credentials To The Production Stack Are Critical!

What Is A Secret

- m-w.com: *kept hidden from others : known to only a few people*
- Examples of Secrets
  - Password
  - Symmetric Encryption Key
  - Private Encryption Key
  - API Key
  - Token

Important Secret attributes:

- Where is it stored?
- Where is it used?
- Who / what is authorized to use it?
- What is it authorized to do?

How Not To Protect a Secret

goto; conference

- Embed it in source code
  - Bonus points for posting to Github once its in there…
- Put it in a configuration file or script, next to what the secret opens
- Encrypt it with a key embedded in the code (or script)

file | 31 lines (29 sloc) | 1.743 kb

```
1  -----BEGIN RSA PRIVATE KEY-----
2  Proc-Type: 4,ENCRYPTED
3  DEK-Info: DES-                0AF9C7E5FBF
4
```

Protecting A Secret

goto; conference

Attributes of Securing a Secret (from Conjur)

- *Self-Auditing*
- *Fully programmable with fine granularity*
- *Highly available across any cloud, hybrid, and global architecture*
- *The secrets should be encrypted when "at rest" in the secrets server*
- *Each secret should be encrypted with a unique key, which is itself encrypted by a master key (or set of master keys)*

- *Cryptography should be profesionally audited, and ideally open-sourced.*
- *Secrets should be encrypted in transit, using e.g. TLS*
- *SSL verification must be ON!*

My Addition: Secrets to secrets is a recursive problem…"Distributed" or "derived" secrets should be granular and less trusted.

Secret (and Crypto) Management Systems

goto; conference

- DIY (Do It Yourself)
- Traditional Crypto Key Managers
  - Definitely for “Keys”
  - But also for other objects (e.g., KMIP Blobs)
- Cloud Solutions
  - AWS CloudHSM
  - AWS KMS
  - AWS S3 (+KMS +IAM)
  - Azure Key Vaults…
- Conjur Secrets Management
- Vault from Hashicorp
- KeyWhiz (open source from Square)
- Barbican (OpenStack)
- Chef-Vault?
- And More…

Know Your Capabilities and Security Needs

Crypto

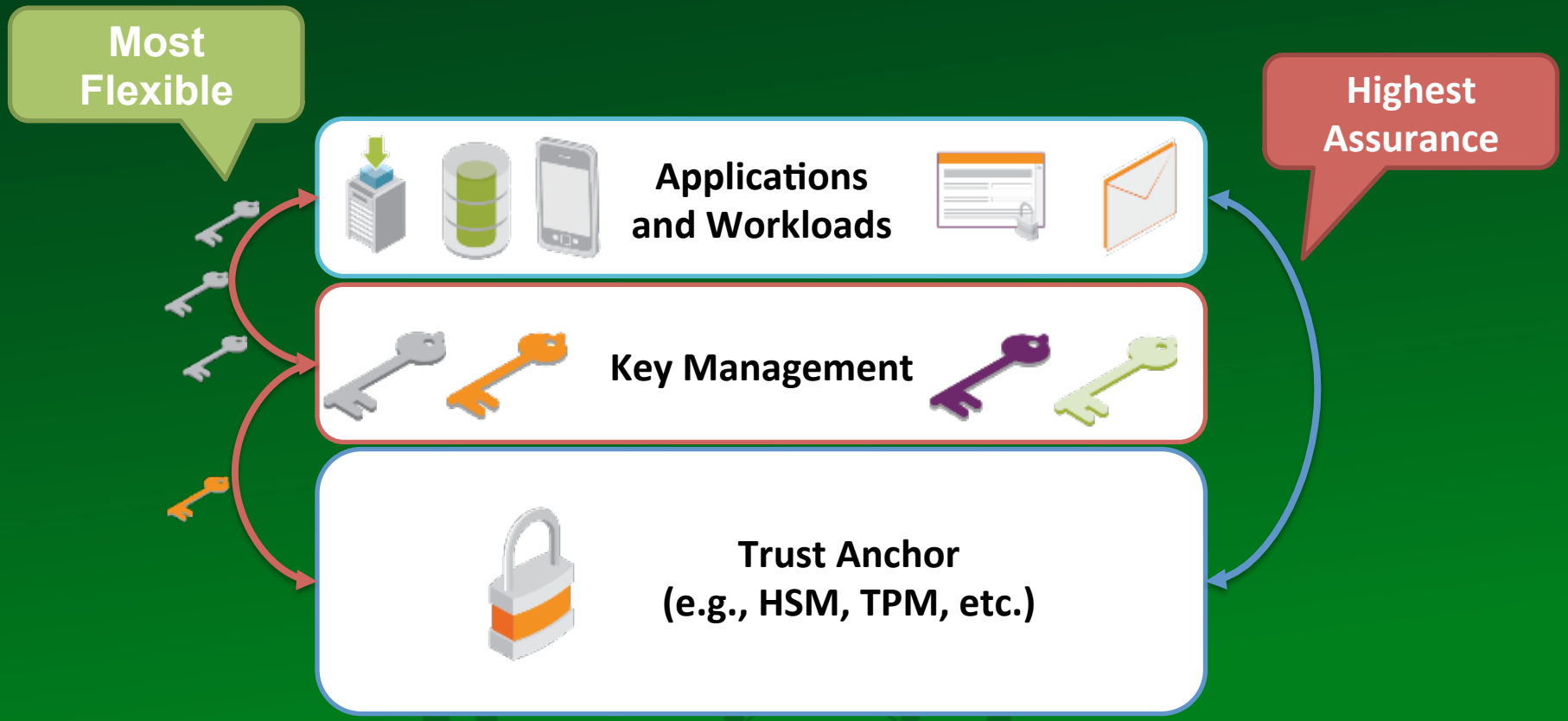
goto; conference

- Powerful tool, but crypto #fail hurts
  - Accidentally destroy a key = destroy data/value
  - Poor implementations easily breakable

Crypto Allows Your To Put Data In Hostile Environment With Near Mathematical Reliability… If Implemented Properly

Key Hierarchies and Roots of Trust

goto; conference

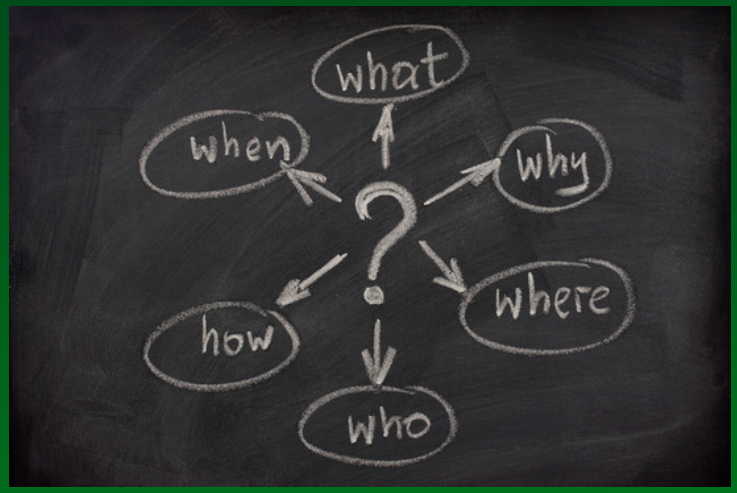


Key Management and Assurance Levels Matter…

Logging and Auditing

goto; conference

- Can be boring, but is essential
- Great starting point to automate security and compliance testing
- DevOps teams better prepared than anyone—if you can do a rollback…
- Capture and maintain key attributes (6 “W”s)
- Secure / tamper evident
- Work with compliance team to automate reports



Takeaways

goto; conference

- Find common ground with security on security and compliance automation
- Focus on privilege users and infrastructure
- If you have a secret, make it secret
  - Don’t take crypto lightly…
- Make security portable

goto; conference

Please

Remember to rate this session

Thank you!

Join the conversation #gotoldn

goto; conference

Thanks !

Join the conversation #gotoldn