Amazon Web Services Security

Joel Leino / Solinor Oy

SOLINOR

About me

- CTO & interim CSO at Solinor
- >8 years with PCI DSS certified payment systems – 2.5 years in AWS
- Serverless Architecture and Microservices enthusiast

My perspective to security

- ”The main question about it is not whether we are safe or not but whether it is worth it.”
  - Bruce Schneier
- Security is essentially a tradeoff
  - Security
  - Cost
  - Performance
  - Reliability
  - Maintainability
- Information, Awareness, Context

AWS Security & Compliance

- AWS is probably much more secure than your current data center / service provider
- All major security certifications
- Security economies of scale – same features for everyone
- Reduced scope of compliance
- Security tools & services

AWS Security & Compliance

Certifications / Attestations	Laws, Regulations, and Privacy	Alignments / Frameworks
DoD CSM	CS Mark [Japan]	CJIS
FedRAMP	EAR	CLIA
FIPS	EU Model Clauses	CMS EDGE
IRAP	FERPA	CMSR
ISO 9001	GLBA	CSA
ISO 27001	HIPAA	FDA
ISO 27017	HITECH	FedRAMP TIC
ISO 27018	IRS 1075	FISC
MLPS Level 3	ITAR	FISMA
MTCS	My Number Act [Japan]	G-Cloud
PCI DSS Level 1	U.K. DPA - 1988	GxP (FDA CFR 21 Part 11)
SEC Rule 17-a-4(f)	VPAT / Section 508	IT Grundschutz
SOC 1	EU Data Protection Directive	MITA 3.0
SOC 2	Privacy Act [Australia]	MPAA
SOC 3	Privacy Act [New Zealand]	NERC
	PDPA - 2010 [Malaysia]	NIST
	PDPA - 2012 [Singapore]	PHR
		UK Cyber Essentials

AWS Shared Responsibility Model

- Security of the cloud
  - AWS's responsibility
- Security in the cloud
  - Customer's responsibility
- Achieving compliance:
  - E.g. PCI DSS Compliance Package
  - Responsibility Matrix, which describes the customer and AWS shared responsibility for each of the 200+ PCI Data Security Standard controls.

Attacker motivation (few examples)

- Accidental discovery / Bots / Script kiddies
  - API / access keys in CI/CD systems, Github etc.
- Resources
  - Unlimited capacity & computing power
  - Botnets, crypto-currency mining, etc.
- Access to sensitive data
  - Identity theft, Credit Cards, Medical records
- Extortion
  - Demise of Service (through AWS root console access)

SOLINOR

AWS Attack Vectors

- AWS Root Account & Identity & Access Management (IAM) users
- API / access keys
- Managed Services
- Network
- Instances (Virtual Machines, EC2)
- Custom applications & 3rd party software

SOLINOR

AWS Attack Vectors

- Leaked credentials
- Access control misconfiguration
- Managed service misconfiguration
- Network security misconfiguration
- Instance misconfiguration
- Software security holes
- Insecure custom applications

Account security

- Root account has unlimited access to everything.

- Protect the root account
  - Create IAM user accounts for day-to-day use
  - Use strong passwords, multi-factor authentication
  - Do not use root access keys

- Create separate AWS accounts different microservices, testing, production, etc.

- Create separate AWS accounts for security critical components
  - Easy way to limit scope of a security breach
  - E.g. backups, VPNs, sensitive data, critical services

Account security

- Protect the API / access keys
    - Avoid storing to Github (oldie but a goldie)
    - Secure credentials stored to CI/CD systems
- Always follow principle of least privilege
- Force password policies for IAM users
- Use Trusted Advisor, check IAM Credential Report
- Use CloudTrail for logging & monitoring
- Monitor: (CloudWatch alarms)
    - Root logins, IAM policy changes, unauthorized API calls, CloudTrail configuration changes, authentication failures, billing alerts, etc.

Identity and Access Management

- Use groups & role based access control
  - Attach policies to groups/roles
  - For complex environments use IAM Federation (SAML, ADFS)

- Policy parameters:
  - IP address
  - Time/date
  - Service
  - Multi-factor authentication (MFA) used
  - Region
  - Etc.

- IAM policy simulator

- Keep it simple stupid

Managed services (RDS, S3, etc.)

- Use managed services whenever possible
  - No need for EC2 instances & security patching
- Principle of least privilege
- Do not make services publicly accessible if not really needed
- Use available security features
  - Encryption, key management, TLS, etc.
- Backup your business critical data
  - Use automated backups and versioning if available (RDS, S3, etc.)

Network Security

- Automatic DDoS protection
- Virtual Private Cloud (VPC)
  - Logically isolated section
- Public and private IP subnets
  - Public = traffic is routed to an Internet gateway
- Network Access Control Lists (ACLs)
  - Stateless firewall at subnet level
  - Allow all by default
- Security Groups
  - Firewall At Instance Level
  - Deny all by default
  - Also as the source or destination for a rule
- Host firewalls / HIDS

Network Security

- Avoid too complex networks
  - Simple networks, simple services, simple AWS accounts
- Protect your Elastic IPs
  - Get reverse DNS names (request form)
- Use jump/bastion hosts, NAT, VPNs
  - Managed NAT service now available
- Use CloudWatch VPC Flow Logging
  - Accepted traffic, rejected traffic
- Do network level security scans
  - AWS Vulnerability / Penetration Testing Request

Instances (EC2)

- **Automate instance deployment & configuration**
  - **Stateless servers = easier security patching**
- **Disable remote administration access (SSH)**
  - **Use centralized user management**
- **Implement monitoring, centralized logging**
  - **CloudWatch Logs, CloudWatch Dashboards**
- **Use host audit tools / security tools / firewalls**
- **Defence in depth:**
  - **CloudFront, Web Application Firewall (WAF), ELB, EC2**
- **Use AWS Lambda :)**

Custom applications

- Secure cloud infrastructure does not make your application secure
- Use Microservices & Serverless architecture
- Minimize amount of security critical components
- Follow security best practices, guidelines, design patterns (e.g. OWASP)
- Update 3rd party software & libraries regularly

Protecting your data

- Protect data at rest
  - Again... Principle of least privilege
    - AWS root account & IAM users
    - Database user accounts (RDS)
    - IAM (S3, DynamoDB)
  - Use encryption when available (S3, RDS, EBS, ...)
  - Take backups

- Encryption
  - Use AWS Key Management Service

- Backups
  - Backup to separate AWS account and/or offsite location
  - Allow write-only access to backups
  - For example, use S3 and Glacier

SOLINOR

Lessons learned

- Automate also the security settings
  - Create baseline configuration for AWS accounts
  - Follow best practices / guidelines
- Avoid too complex system design & configurations
  - If something is possible, it does not mean you should do it
- Learn the “Cloud Way” to do things
  - Do not try to use traditional physical datacenter methodologies & practices in cloud environment

SOLINOR

Guidelines & Tools

- **Guidelines:**
  - **CIS Amazon Web Services Foundations Benchmark**
  - **AWS Security Audit Guidelines**
  - **AWS Whitepapers**

Guidelines & Tools

- **Tools:**
  - Trusted Advisor
  - IAM Credential Report
  - CloudTrail & CloudWatch

  - AWS Inspector (preview, EC2 agent)
  - AWS Web Application Firewall

  - evident.io - Security and Compliance Automation
  - Chef Compliance

  - AWS Key Management Service
  - AWS Certificate Manager
  - AWS CloudHSM

  - ...

Questions?

SOLINOR