SANS | CLOUD SECURITY

Beware of the Encryption Jedi Mind Trick

Kenneth G. Hartman

August 29, 2023

About Me

LUCID TRUTH TECHNOLOGIES
LucidTruthTechnologies.com

Kenneth G. Hartman

- Owner – Lucid Truth Technologies, a Digital Forensics Firm
- BS Electrical Engineering, Michigan Technological University
- MS Information Security Engineering, SANS Technology Institute
- Multiple Security Certifications: CISSP, GIAC Security Expert, etc.
- SANS Instructor – SEC488: Cloud Security Essentials
  & SEC510: Public Cloud Security: AWS, Azure, and GCP

www.kennethghartman.com
@kennethghartman

The content and opinions in this presentation are my own and do not necessarily reflect the positions, strategies, or opinions of any current client or previous employer.

SANS

CLOUD SECURITY

CURRICULUM ROADMAP

Baseline

SEC 388 — **Introduction to Cloud Computing and Security**
Ground school for cloud security

Foundational Security Techniques

SEC 488 — **Cloud Security Essentials** | GCLD
License to learn cloud security.

Security Management

MGT 520 — **Leading Cloud Security Design and Implementation**
Chart your course to cloud security.

Core

SEC 510 — **Public Cloud Security: AWS, Azure, and GCP** | GPCS
Multiple clouds require multiple solutions.

SEC 540 — **Cloud Security and DevSecOps Automation** | GCSA
The cloud moves fast. Automate to keep up.

SEC 541 — **Cloud Security Attacker Techniques, Monitoring & Threat Detection** | GCTD
Attackers can run but not hide. Our radar sees all threats.

SEC 549 — **Enterprise Cloud Security Architecture**
Design it right from the start.

Specialization

SEC 522 — **Application Security: Securing Web Apps, APIs, and Microservices** | GWEB
Not a matter of "if" but "when." Be prepared for a web attack. We'll teach you how.

SEC 588 — **Cloud Penetration Testing** | GCPN
Aim your arrows to the sky and penetrate the cloud.

FOR 509 — **Enterprise Cloud Forensics and Incident Response** | GCFR
Find the storm in the cloud.

sans.org/cloud-security @SANSCloudSec linkedin.com/showcase/sanscloudsec

Can I trust my Cloud Provider ???

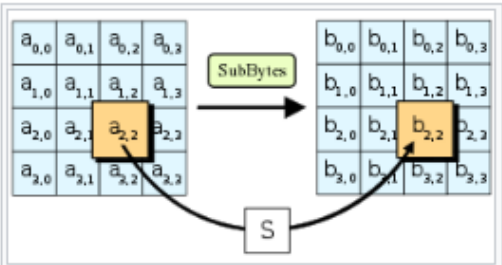

Trust us...
We Use Encryption

Encryption is Hard!

The SubBytes step [edit]

Main article: Rijndael S-box

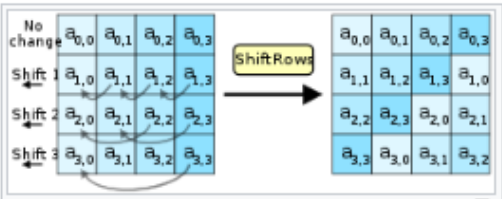
In the SubBytes step, each byte $a_{i,j}$ in the *state* array is replaced with a SubByte $S(a_{i,j})$ using an 8-bit substitution box. Note that before round 0, the *state* array is simply the plaintext/input. This operation provides the non-linearity in the cipher. The S-box used is derived from the multiplicative inverse over GF(2^8), known to have good non-linearity properties. To avoid attacks based on simple algebraic properties, the S-box is constructed by combining the inverse function with an invertible affine transformation. The S-box is also chosen to avoid any fixed points (and so is a derangement), i.e., $S(a_{i,j}) \neq a_{i,j}$, and also any opposite fixed points, i.e., $S(a_{i,j}) \oplus a_{i,j} \neq \mathrm{FF}_{16}$. While performing the decryption, the InvSubBytes step (the inverse of SubBytes) is used, which requires first taking the inverse of the affine transformation and then finding the multiplicative inverse.



In the SubBytes step, each byte in the state is replaced with its entry in a fixed 8-bit lookup table, S; $b_{ij} = S(a_{ij})$.

The ShiftRows step [edit]

The ShiftRows step operates on the rows of the state; it cyclically shifts the bytes in each row by a certain offset. For AES, the first row is left unchanged. Each byte of the second row is shifted one to the left. Similarly, the third and fourth rows are shifted by offsets of two and three respectively.[note 6] In this way, each column of the output state of the ShiftRows step is composed of bytes from each column of the input state. The importance of this step is to avoid the columns being encrypted independently, in which case AES would degenerate into four independent block ciphers.



In the ShiftRows step, bytes in each row of the state are shifted cyclically to the left. The number of places each byte is shifted differs incrementally for each row.

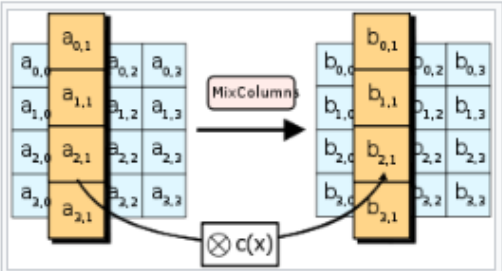
The MixColumns step [edit]

Main article: Rijndael MixColumns

In the MixColumns step, the four bytes of each column of the state are combined using an invertible linear transformation. The MixColumns function takes four bytes as input and outputs four bytes, where each input byte affects all four output bytes. Together with ShiftRows, MixColumns provides diffusion in the cipher.

During this operation, each column is transformed using a fixed matrix (matrix left-multiplied by column gives new value of column in the state):

$$\begin{bmatrix} b_{0,j} \\ b_{1,j} \\ b_{2,j} \\ b_{3,j} \end{bmatrix} = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} a_{0,j} \\ a_{1,j} \\ a_{2,j} \\ a_{3,j} \end{bmatrix} \qquad 0 \leq j \leq 3$$



In the MixColumns step, each column of the state is multiplied with a fixed polynomial $c(x)$.

Matrix multiplication is composed of multiplication and addition of the entries. Entries are bytes treated as coefficients of polynomial of order x^7. Addition is simply XOR. Multiplication is modulo irreducible polynomial $x^8 + x^4 + x^3 + x + 1$. If processed bit by bit, then, after shifting, a conditional XOR with $1B_{16}$ should be performed if the shifted value is larger than FF_{16} (overflow must be corrected by subtraction of generating polynomial). These are special cases of the usual multiplication in $\mathrm{GF}(2^8)$.

In more general sense, each column is treated as a polynomial over $\mathrm{GF}(2^8)$ and is then multiplied modulo $01_{16} \cdot z^4 + 01_{16}$ with a fixed polynomial

Why Use Encryption?

➔ To Control Access

What is a Security Control?

➔ It’s a Counter-Measure

AXIOM:
The entity that holds the key controls the access.


The Danger of
Misplaced Trust

Threat Mitigation by Type of Encryption

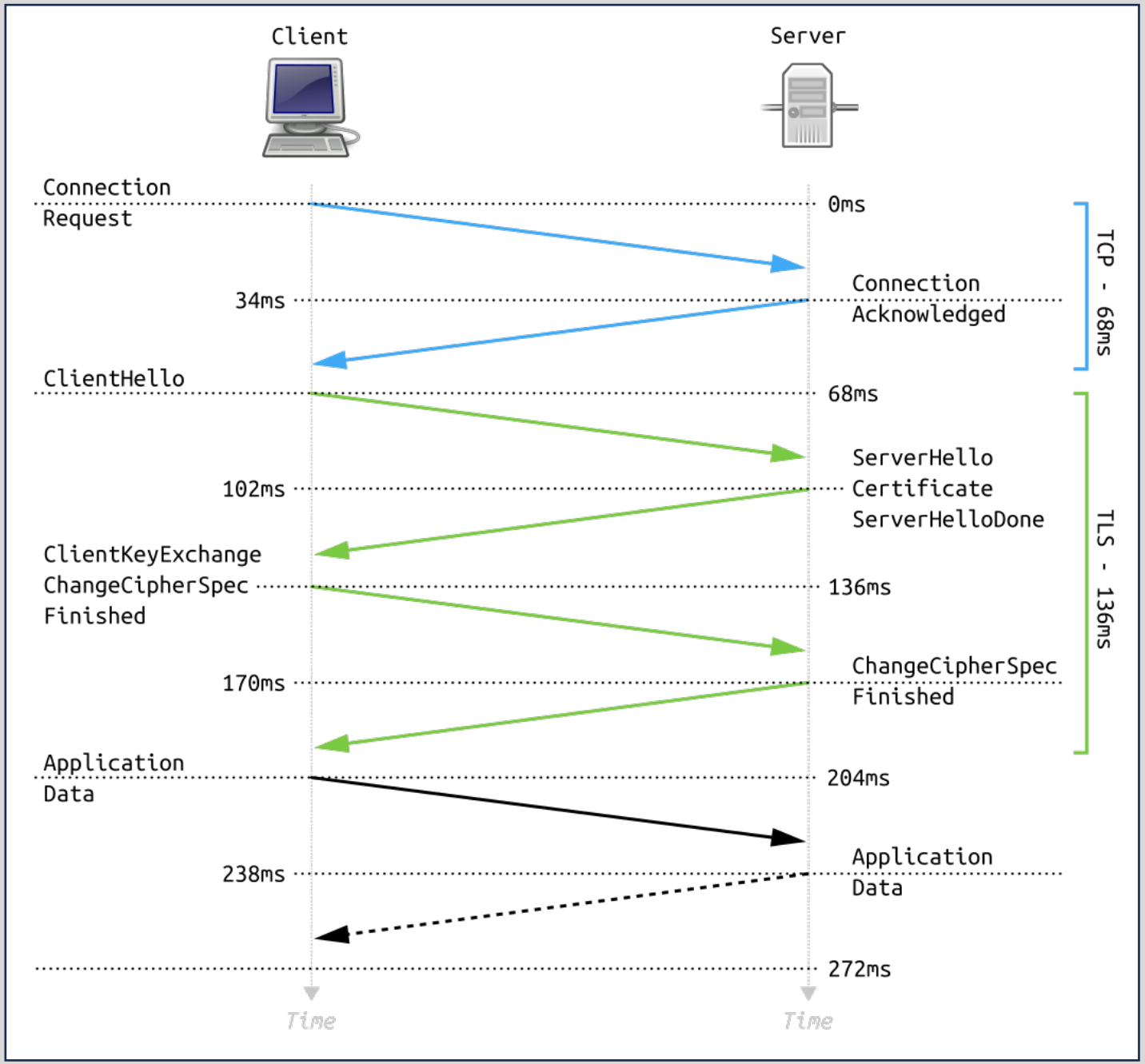
Threat	Application Encryption	Database Encryption	Encrypted Files & Folders	Full Disk / Full Volume Encryption	Self Encrypting Drives
Compromised Service Provider	NO	NO	NO	NO	NO
Compromised App User Credentials	NO	NO	NO	NO	NO
Compromise of Application Server	NO	NO	NO	NO	NO
Rogue Application Administrator	NO	NO	NO	NO	NO
Web Application Flaws	DEPENDS	NO	NO	NO	NO
Compromise of DB Credentials	YES	NO	NO	NO	NO
Compromise of Database Server	YES	NO	NO	NO	NO
Rogue Database Administrator	YES	NO	NO	NO	NO
Compromise of Storage Server Creds	YES	YES	NO	NO	NO
Compromise of Storage Server	YES	YES	NO	NO	NO
Rogue Storage Administrator	YES	YES	NO	NO	NO
Unauthorized File System Access	YES	YES	YES	NO	NO
Improper Disposal of Hard Drive	YES	YES	YES	YES	YES
Hard Drive Theft	YES	YES	YES	YES	YES

What risks are mitigated by this security control?

What Risks are NOT mitigated?

“We Use End-to-End Encryption”

Transport Layer Security



https://en.wikipedia.org/wiki/Transport_Layer_Security#TLS_handshake

Encryption-in-transit

Machine-in-the-Middle (MITM) Attack Scenario:

The Attacker impersonates the Client to the Server and the Server to the Client.

Qualys. SSL Labs

Home Projects Qualys Free Trial Contact

You are here: Home > Projects > SSL Server Test > sans.org > 45.60.103.34

SSL Report: sans.org (45.60.103.34)

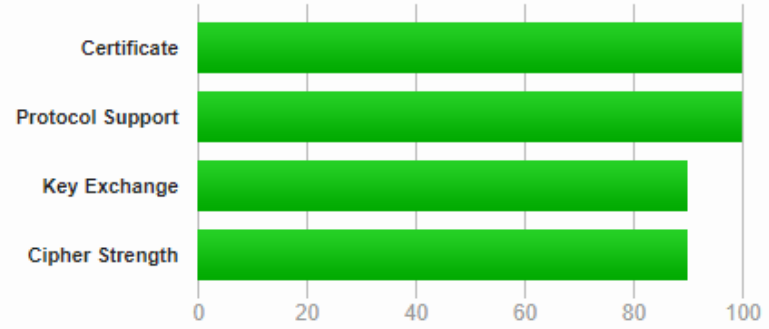
Assessed on: Tue, 29 Aug 2023 08:32:05 UTC | HIDDEN | Clear cache

Scan Another »

Summary

Overall Rating

A+



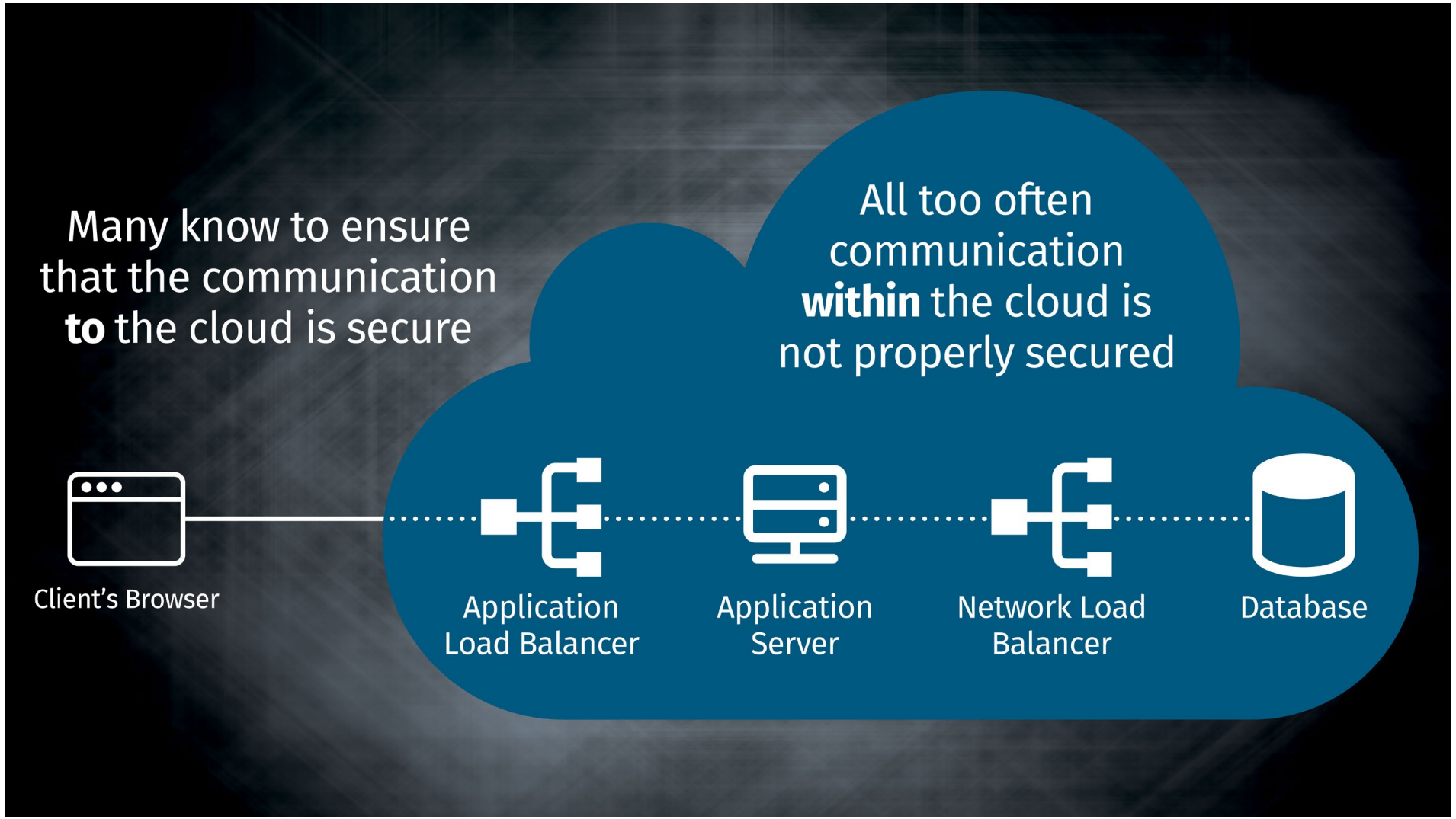
Visit our documentation page for more information, configuration guides, and books. Known issues are documented here.

This server supports TLS 1.3.

HTTP Strict Transport Security (HSTS) with long duration deployed on this server. MORE INFO »

DNS Certification Authority Authorization (CAA) Policy found for this domain. MORE INFO »

Encryption In-Transit throughout the Cloud



Which is better?

Client-Side Encryption

- Encryption and decryption of data is performed by the client, or user's device.
- The encryption keys are never stored or accessible by the server.

Server-Side Encryption

- Encryption and decryption of data is performed by the server on which the data is stored
- The encryption keys are never stored or accessible by the user

The entity that holds the key controls the access.

Hardware Security Modules

Perform a limited number of cryptographic functions:

- Key generation, hashing, symmetric and asymmetric encryption

Provides for the protection of high value keys (“root of trust”)

Designed to protect against unauthorized access, modification, or disclosure of sensitive information, including by tampering and environmental threats

Need to be clustered to protect against attacks on availability

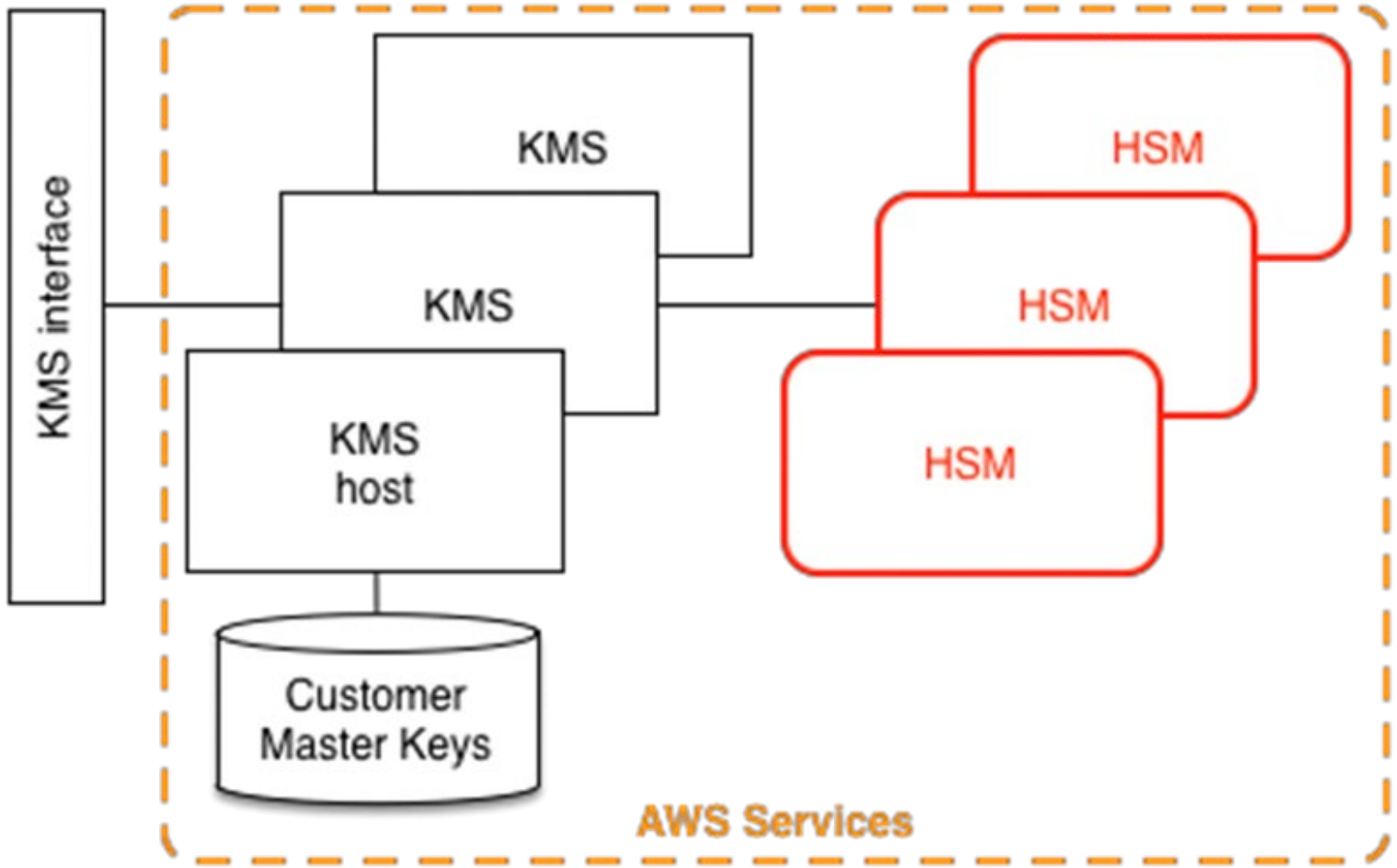
The Federal Information Processing Standard (FIPS) 140-2

- **Level 1:** This is the lowest level of security. It provides **basic protection** against unauthorized access, modification, or disclosure of sensitive information.
- **Level 2:** This level provides more security than Level 1. **It includes additional measures to protect against unauthorized access, modification, or disclosure of sensitive information.**
- **Level 3:** This level provides the highest level of security. It includes all of the measures in Level 2, plus additional measures to **protect against tampering and environmental threats.**
- **Level 4:** This level is reserved for cryptographic modules that are used to protect **national security information**. It includes all of the measures in Level 3, plus additional measures to protect against physical and environmental threats.

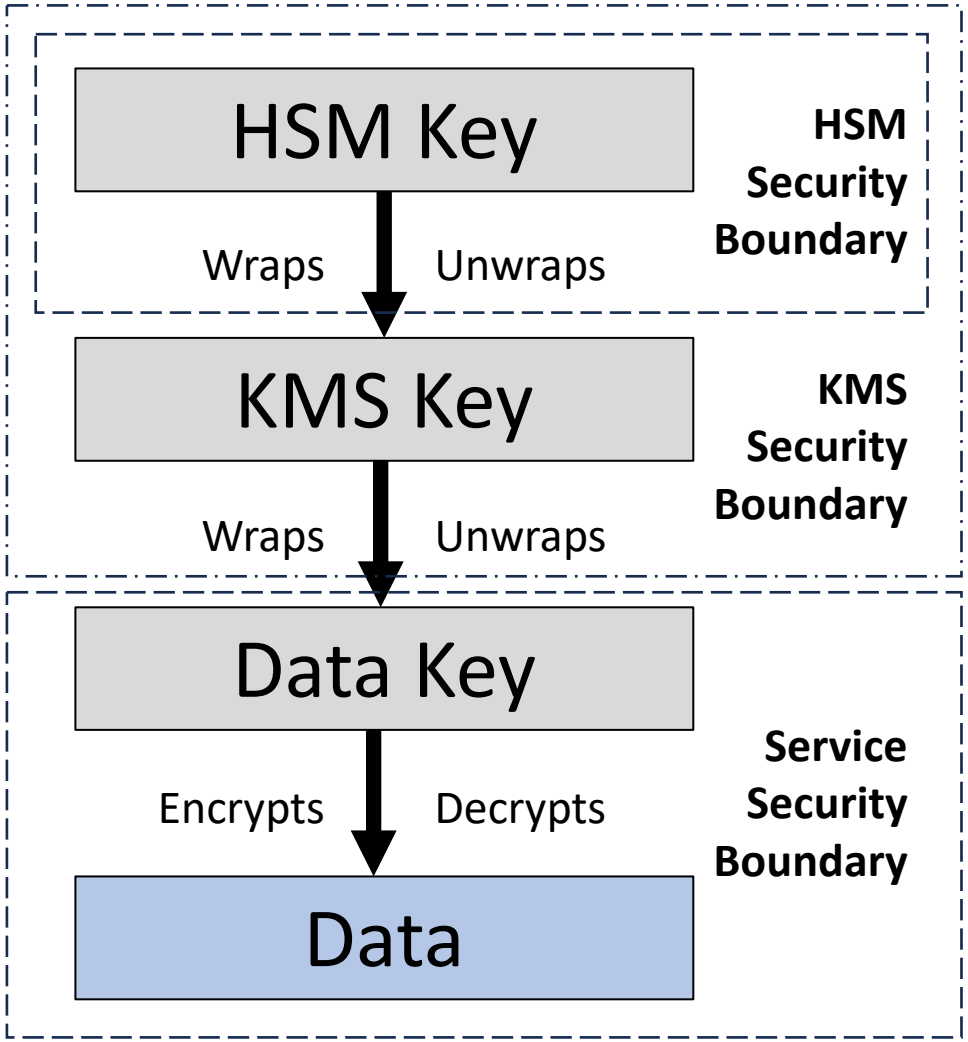
FIPS 140-2 Security Requirements Coverage

- **Physical security:** The cryptographic module must be protected from physical tampering and environmental threats.
- **Design:** The cryptographic module must be designed to be secure.
- **Implementation:** The cryptographic module must be implemented correctly.
- **Testing:** The cryptographic module must be tested to ensure that it meets the security requirements.
- **Documentation:** The cryptographic module must be documented so that its security can be properly evaluated.

Key Management Systems (KMS)

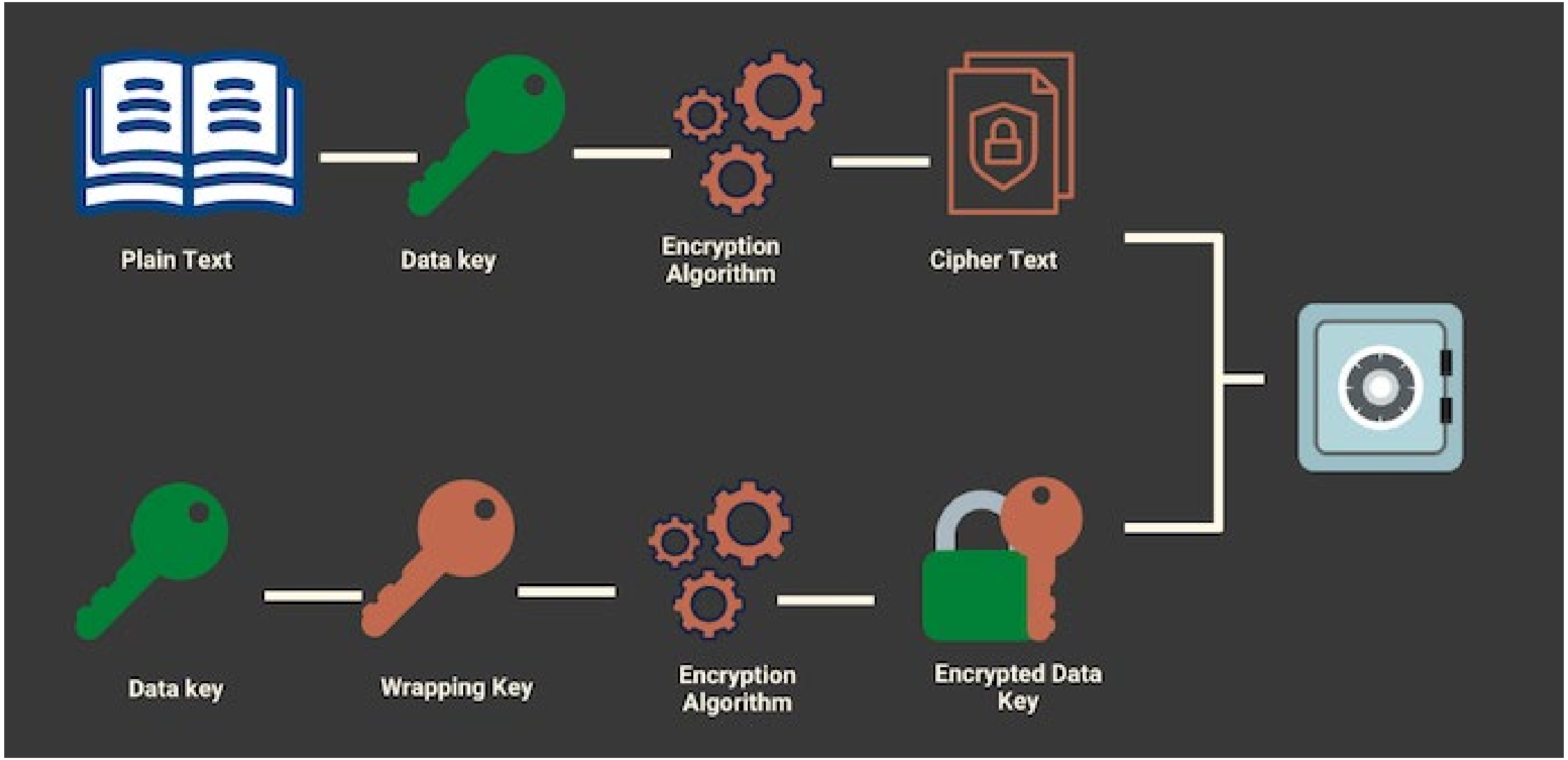


Key Hierarchy



- The HSM key never leaves the HSM Security Boundary
- The KMS Key is wrapped/unwrapped by the HSM Key in the HSM
- The KMS Key never leaves the KMS Security Boundary
- The Data Key is wrapped/unwrapped by the KMS Key in the KMS
- Unwrapped Keys are never persisted to storage
- Keys are wiped from memory after use

Envelope Encryption



https://www.archerimagine.com/articles/aws/aws-kms.html

Customer Master Keys

Designation	Source of Key Material	Manages Life Cycle
Customer Provided Keys	Customer	Customer
Customer Managed Keys	Cloud Provider	Customer
Provider Managed Keys	Cloud Provider	Cloud Provider

Cryptographic Erase

Cryptographic erasure is a method of sanitizing data by sanitizing the media encryption key (MEK). This makes it impossible to recover the decrypted data.

If strong cryptography is used, sanitization of the target data is reduced to sanitization of the encryption key(s) used to encrypt the target data.

Thus, with CE, sanitization may be performed with high assurance much faster than with other sanitization techniques.

https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-88r1.pdf

Bring Your Own Key?

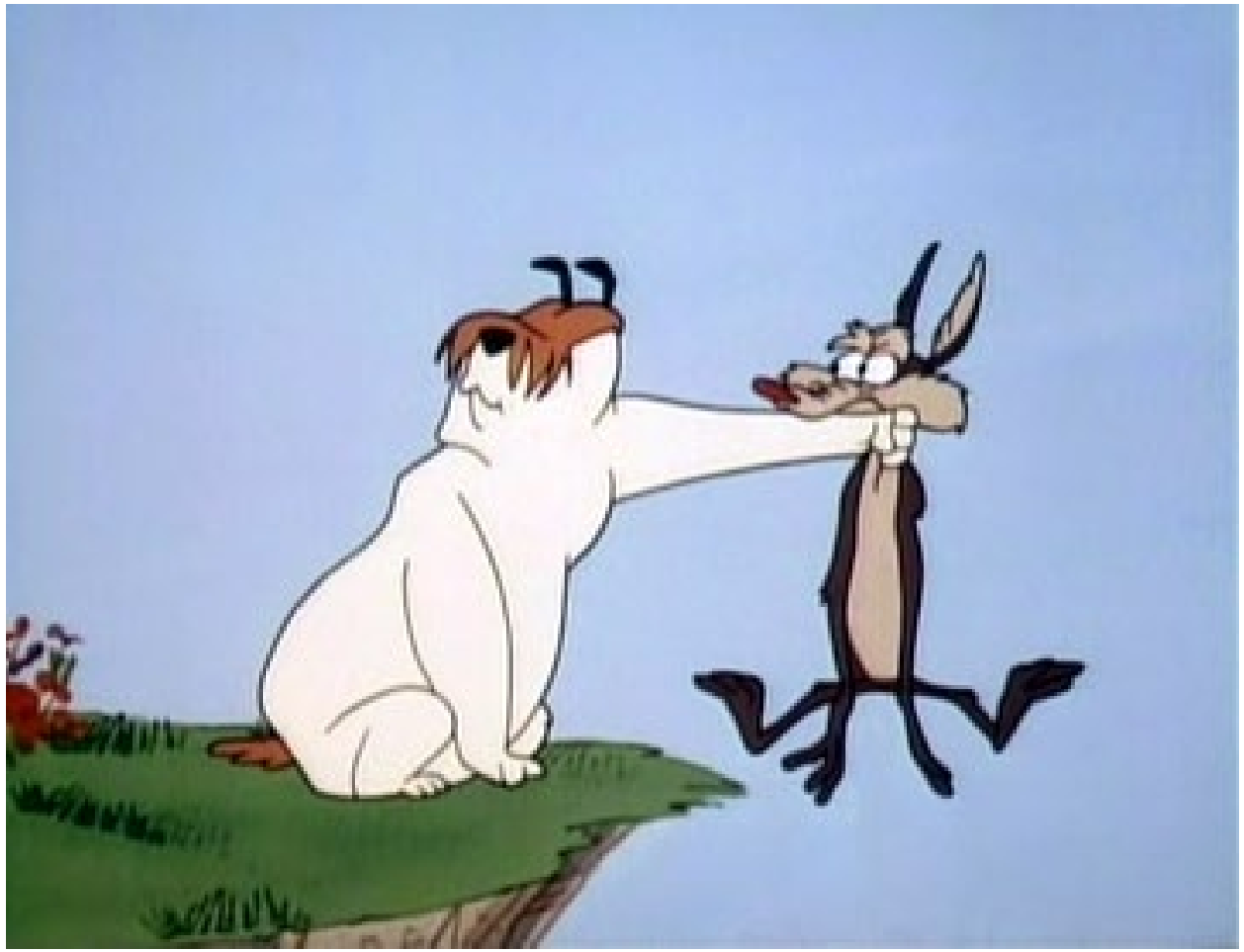
SANS CLOUD SECURITY

SEC510
Public Cloud Security: AWS, Azure, and GCP

Live Online	*In Person & Live Online*	*In Person & Live Online*
Sept 25-29	**Oct 18-22**	**Nov 6-10**
SANS Managing Risk	**CloudSecNext | Dallas**	**San Diego Fall**
Brandon Evans	**Kenneth G. Hartman**	**Brandon Evans**

GIAC PUBLIC CLOUD SECURITY GPCS

GIAC CERTIFICATIONS



Questions & Discussion