CRASHTEST **SECURITY**

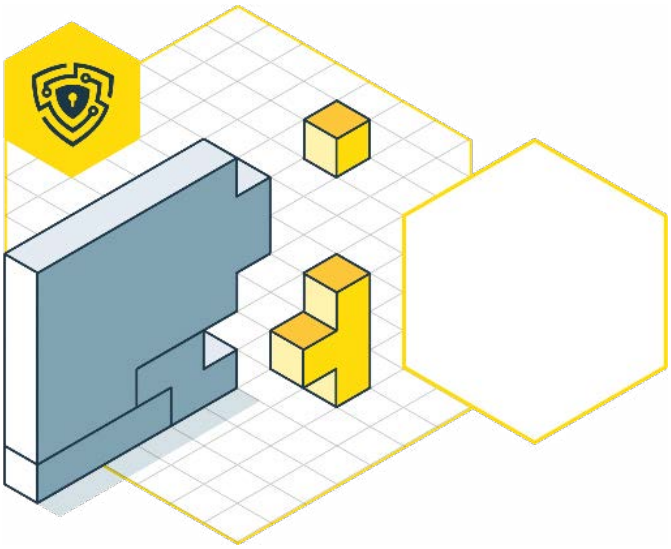
GUIDE FOR

PREVENTING CRYPTOGRAPHIC FAILURES

WHAT ARE THE STEPS TO KEEP YOUR WEB APP OR API SAFE FROM SUCH VULNERABILITY

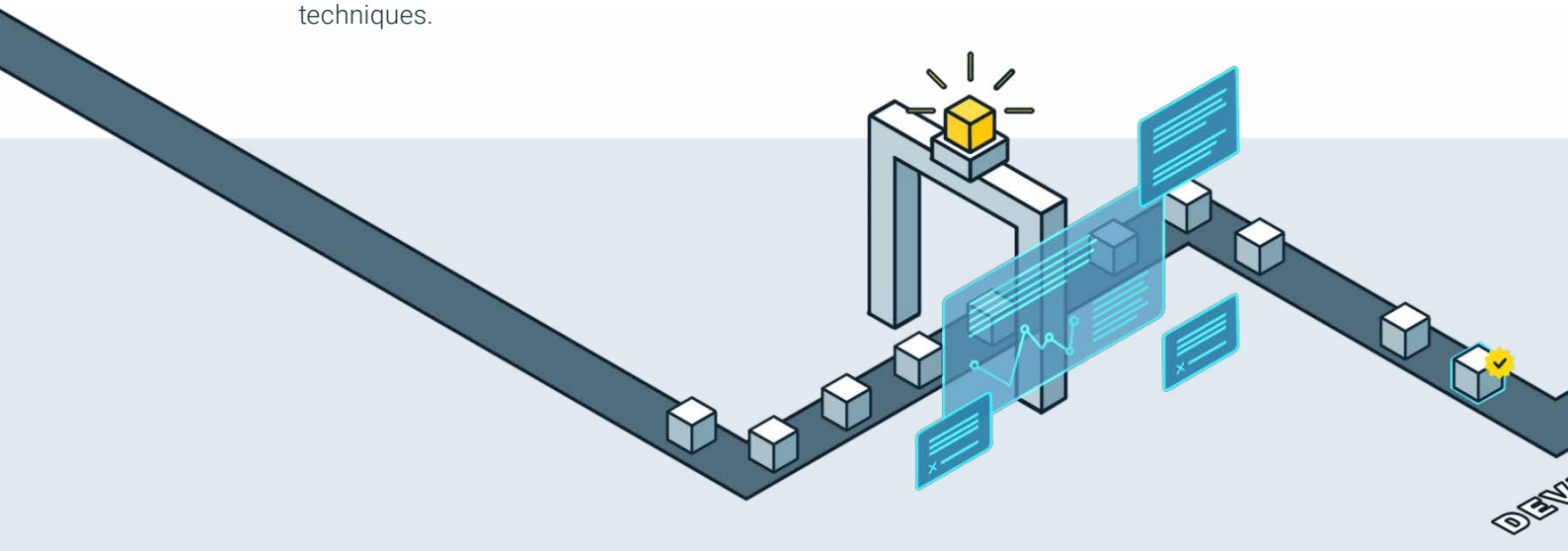
GUIDE FOR CRYPTOGRAPHIC FAILURE PREVENTION

Table of Contents



INTRODUCTION TO THIS GUIDE

Cryptography is one of the key components in cyber security that relies on codes to ensure that a message can only be read by the sender and intended recipient. The data encryption security feature guarantees confidentiality, integrity, non-repudiation, and user authentication in modern web systems. Cryptographic functions encrypt and decrypt plain-text messages to ensure secure electronic data transmission between entities, preventing a successful man-in-the-middle attack. Cryptographic failure encompasses a collection of application security risks that expose sensitive data and files through weak encryption techniques. This guide discusses the cryptographic failure vulnerability, its types, and possible prevention techniques.



WHAT IS CRYPTOGRAPHIC FAILURE?

Cryptographic failures occur when third-party entities unintentionally expose sensitive data. Insufficient cryptography may arise due to inadequate protection, security misconfiguration, or inappropriate usage of cryptographic algorithms. Also known as sensitive data exposure, a cryptographic failure breaks the cybersecurity trust chain, providing unauthorized access to restricted content.

The vulnerability encompasses a wide range of security flaws when attackers compromise the cryptographic protocol to intercept communication between a sender and the intended recipient. While it is not possible to build a completely uncrackable encryption algorithm, stronger protocols are less likely to be exploited through basic computational capabilities and knowledge. Even with a robust encryption protocol, security professionals may ignore data transmission secure design best practices, subsequently exposing sensitive information.

TYPES OF CRYPTOGRAPHIC FAILURES

Cryptographic failures are categorized according to the specific flaw that results in the exposure of sensitive data, such as user credentials and health records. Some common forms of cryptographic failures include:

USE OF HARD-CODED PASSWORD

This is a commonly found security flaw in which the software exposes plain-text passwords and other secrets within the source code. Listed as CWE-59, the flaw has two variations:

Inbound

The software contains embedded credentials within its authentication mechanism in this scenario. When an account is created, the application assigns a simple default password that is hard-coded and associated with the account. This password is the same for each product deployment that super-users cannot change without considerable modification of the application's source code repositories. If the password is ever exposed to attacker-controlled password databases, then anybody with knowledge of the client application can exploit it to gain access. Since all installations use this default password for new accounts, attackers can orchestrate deep system-level attacks such as denial of service.

Outbound

This flaw occurs when the vulnerable application contains a hard-coded password for connecting to other systems/components. The vulnerability applies to front-end services authenticating to back-end systems, typically requiring a fixed, complex password. If developers hard-code strong passwords, threat actors can employ advanced hacking techniques to obtain and use them for advanced attacks.

INSUFFICIENT ENTROPY

Cryptographic functions use a pseudo-random number generator (PRNG) to generate hashes, salts, and other random values in encryption protections. In vulnerable applications, a cryptographically weak pseudo-random number generator creates values that are easy for the attacker to guess. Insufficient randomness/entropy offers attackers access to the seed or internal state of PRNG, which generates values or patterns that are easy to predict. Since the application uses these values for authentication and authorization, attackers can gain access to sensitive, restricted information. Insufficient entropy is a base-level weakness in the Common Weakness Enumeration database as entry CWE-331.

USE OF RISKY/BROKEN CRYPTOGRAPHIC ALGORITHMS

Developers are advised to use standard algorithms such as SHA 256, SHA 384, and SHA 512 that are robust and have been developed to tackle modern threats. When the web application uses a less-secure algorithm for encryption, a determined attacker can break it to compromise the protected message contents. Applications that use outdated, weak encoding and encryption algorithms allow attackers to intercept secured communications and successfully orchestrate a man-in-the-middle attack. The flaw is listed as CWE-331 in the Common Weakness Enumeration database and is attributed to a high-severity vulnerability because of the severe consequences of a successful attack.

IMPROPER FOLLOWING OF A CERTIFICATE‘S CHAIN OF TRUST

The server certificate is a crucial security feature that helps secure communications by verifying the website‘s ownership and preventing attackers from creating a site clone. In most cases, the certificate traverses multiple intermediate entities who vouch for one another, with the user at the end of the trust chain. Improper following of the certificate‘s trust chain (CWE-296) reduces the usefulness of the certificate. In such cases, the client may receive a certificate and only check the first entity within the chain, which derives no real trust since a valid certificate should be traced back to the source.

A trust chain can be broken due to several different ways, including:

- Self-signed, non-root certificates
- Improper validation of intermediate certificates
- Lack of expected primary constraints and crucial extensions in intermediate certificates
- Compromised/broken root certificates

REVERSIBLE ONE-WAY HASH

This vulnerability occurs in websites that use hashing algorithms that can be used to generate the original input. Hackers leverage a brute-force attack to determine the actual value or find a value that can generate the same hash functions. This enables successful cryptanalysis of the hashing algorithms. Attackers also exploit the reversible one-way hash (CWE-328) flaw to bypass authentication by entering an alternate password that generates the identical pre-calculated hashes, enabling them to take over a victim's account entirely.

UNPROTECTED TRANSPORT OF CREDENTIALS

This flaw occurs when an application fails to protect the username and password while in transit from the login database to the application server. The login page fails to apply effective security measures to the credentials as they are transmitted from the client machine to the server. Websites commonly use a secure sockets layer (SSL) to ensure the integrity and confidentiality of credential data. An improper SSL implementation for the login page allows attackers to eavesdrop and alter user credentials, leading to compromised accounts.

CRYPTOGRAPHIC FAILURE - SEVERITY LEVEL

Moving up from number five on the 2017 top vulnerabilities list, the cryptographic failure vulnerability is ranked number two on the 2021 OWASP top 10 lists. As of 2021, 29 common weaknesses and enumerations were mapped to the cryptographic failure vulnerability. The flaw has an average incidence rate of 4.49%, a weighted exploit of 7.29, and an average impact score of 6.81, making it a high-priority vulnerability.

HOW TO IDENTIFY CRYPTOGRAPHIC FAILURE WITH CRASHTEST SECURITY?

The Crashtest Security Suite provides advanced ethical hacking and automated vulnerability scanning functionalities to keep websites safe from cryptographic attacks. Crashtest Security's black box penetration testing helps security professionals identify cryptographic attack vectors in web applications and APIs even with limited knowledge of client libraries, application architecture, or internal traffic procedures.

The platform employs various automated scanners for identifying potential vulnerabilities within cryptographic systems. These include:

- **Privilege escalation scanner** helps identify accounts that may have been compromised through an insufficient transport of credentials
- **HTTP header scanner** helps developers identify cryptographic attack attempts via malformed requests.
- **SSL/TLS scanner** identifies flaws in implementing HTTPS communication, which may lead to encrypted content interception.

Crashtest Security also offers comprehensive security reports that help analyze every identified vulnerability in detail and provide remediation guidance so that developers and security professionals don't waste time researching security controls.

CRYPTOGRAPHIC FAILURE PREVENTION TECHNIQUES

Encryption protections form the first line of defense against cryptographic failures. Other security measures to prevent cryptographic failures include:

HTTP STRICT TRANSPORT SECURITY (HSTS)

The HSTS policy mechanism helps prevent man-in-the-middle attacks by informing the client application that the website should only be accessed via HTTPS. Since attackers can intercept the initial HTTP connection, all incoming HTTP requests should be converted to HTTPS by default and are more effective than a server-side HTTP to HTTPS redirect.

HSTS is defined as a header field within the HTTP response named strict-transport-security. The HSTS protocol is supported by all major browsers and is effective against active or passive cryptographic attacks.

SERVER-SIDE CIPHER PRIORITIZATION

The server should be configured to evaluate the list of supported cipher suites, check which ones are compatible with the client, and use it during the handshake of the connection. The selected cipher suite is a combination of:

- The key generation and exchange algorithm
- Authentication algorithm
- Bulk encryption algorithm
- Message Authentication Code (MAC) algorithm

BEST PRACTICES IN PREVENTING CRYPTOGRAPHIC FAILURES

Some best practices to prevent cryptographic attacks include:

DISCARD UNNECESSARY DATA

To avoid exposure to sensitive data, developers should avoid storing the data unnecessarily. A recommended approach is to employ message truncation or PCI-DSS compliant tokenization to replace sensitive data with non-sensitive placeholders or remove a portion of the data altogether.

ENFORCE CRYPTOGRAPHIC RANDOMNESS USING APPROPRIATE INITIALIZATION VECTORS (IVS)

An IV should be chosen appropriately to suit the same mode of operation. For most applications, this means using a cryptographically secure pseudo-random number generator. It is also recommended that the initial vector is not used twice for proper key management.

AVOID LEGACY PROTOCOLS WHEN TRANSPORTING SENSITIVE DATA

Legacy protocols such as SMTP and FTP include insecure design flaws that hackers can crack with minimal resources and within a reasonable time frame. Modern, advanced, mainstream algorithms address these flaws to ensure data is protected at rest and in transit.

START AUTOMATED TESTING AND SCANNING TODAY

The Crashtest Security Suite implements vulnerability scanners to automate the testing of web applications, APIs, and JavaScript testing to help spot common web application security risks. The suite offers rapid security assessment that enables C-level management, security, and DevOps teams to swiftly assess and mitigate potential security exploits.

Contact us here to know more and start a free,14-day trial of the Crashtest Security platform.

Start 2-Week Trial for Free

CRASHTEST **SECURITY**

WWW.CRASHTEST-SECURITY.COM