CRASHTEST **SECURITY**

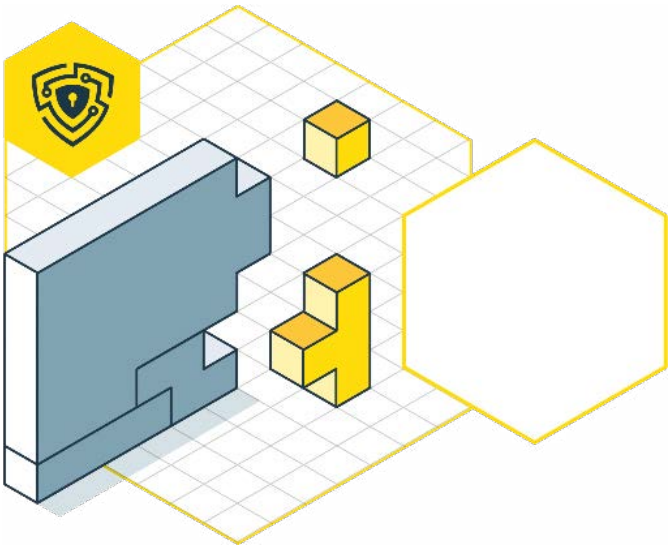
GUIDE FOR

PREVENTING SECURIY MISCON-FIGURATION

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

GUIDE FOR THE SECURITY MISCONFIGURATION PREVENTION

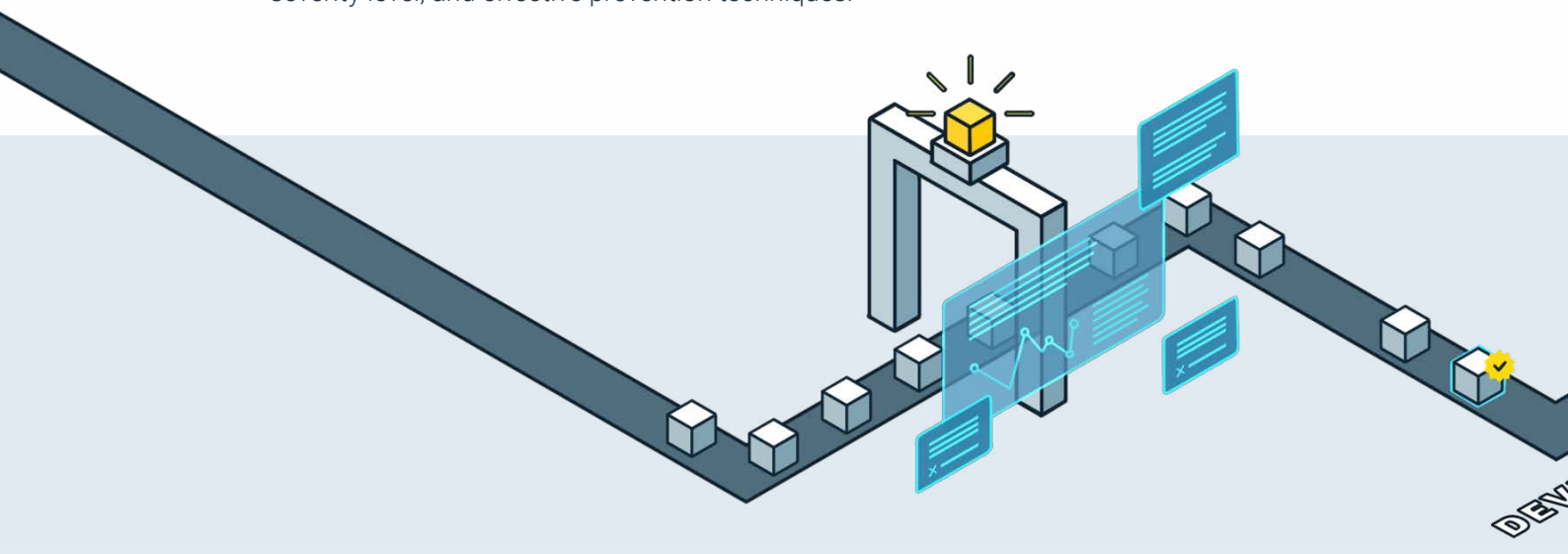
Table of Contents



INTRODUCTION TO THIS GUIDE

Modern applications include baseline security configurations before they are deployed to an operational production environment. These baseline configurations also include default security controls that define authentication mechanisms, user registration, and component update functions. Default settings in the application stack often expose security vulnerabilities since malicious actors leverage known security controls to gain access to the system. This form of exploit is known as a security misconfiguration attack and is attributed as one of the leading causes of most modern cyber attacks.

This guide explores the security misconfiguration vulnerability, common misconfigurations, severity level, and effective prevention techniques.



WHAT IS SECURITY MISCONFIGURATION?

Security misconfigurations in an application framework can occur for many reasons, including disregarding security practices, incorrect implementation, or retaining default security settings. Improper configurations may ensue across various application stack layers, such as network access controls, cloud services, production servers, container images, handheld devices, and other critical assets within the application development environment.

Insecure configurations provide threat actors with unauthorized access to the production environment, allowing them to orchestrate malicious activities that usually carry a low probability of detection. In a modern networked environment, insecure misconfigurations induce some of the most severe consequences, including costly data breaches, compromised individual devices, and cloud security failures.

TYPES OF SECURITY MISCONFIGURATION

Security misconfigurations arise due to inherent framework issues and the choice of poor practices. Some security misconfigurations may be a result of:

Default account settings

Most deployments include default credentials to access admin portals for initial security settings. Vulnerabilities arise when administrators use pre-defined default passwords and usernames that attackers can guess or obtain from dark web password databases. Since these default account settings are used as an initial access point to compromise the entire environment from the admin console, the consequences of malicious attacks orchestrated on such vulnerabilities are often far-reaching.

Error handling misconfigurations

Some applications display informative error messages that expose sensitive information. Detailed error messages can reveal stack traces, application server connections, and database dumps, which typically indicate configuration flaws or an application's response in a given scenario. Attackers frequently abuse such indications of the application's behavior for coordinating system-level attacks.

Exposed passwords in configuration files

Some deployments include plain-text passwords in unprotected files, which any malicious user with incorrect permissions can access. When applications store passwords in files accessible to unauthorized users, threat actors can access password-protected resources, perform privilege escalation attacks, lock users out of their accounts, or change device configurations to facilitate further breaches. Such vulnerabilities are prevalent when developers ignore the recommended practice of storing credentials in encrypted files and keeping them separate from configuration files.

Insufficient firewall protection

While firewalls are crucial in protecting internal assets, they may expose vulnerabilities that arise due to human errors in their configuration and implementation. A firewall cannot protect the network from security threats if it allows malicious requests to flow through it. Improper traffic scanning may also fail to distinguish ordinary users from attackers, enabling undetectable exploits. A compromised firewall can be used for various exploits, including DDoS attacks, insider attacks, and malware attacks.

Common misconfigurations in administering firewalls include:

- Retaining unnecessary features available on the firewall
- Missing or inadequate security patches
- Unplanned opening of ports
- Returning Deny instead of Drop response for blocked ports, facilitating quicker port scans
- Enabling a TCP ping of internal assets with external IP addresses
- Allowing pings of the firewall through ICMP

Exposure of sensitive information through environment variables

Developers often rely on environment variables to parse the configuration information of an application server. Some of these variables provide detailed information about the container images, file directories, and component versions used in the production environment. Since most processes reuse variables across multiple child processes, leakage of a single secret within environment variables sets the entire deployment at risk. Attackers often target improper authorization controls to interject these variables and use them to access critical resources, thereby compromising an whole environment.

Improper session handling

Developers often rely on environment variables to parse the configuration information of an application server. Some of these variables provide detailed information about the container images, file directories, and component versions used in the production environment. Since most processes reuse variables across multiple child processes, leakage of a single secret within environment variables sets the entire deployment at risk. Attackers often target improper authorization controls to interject these variables and use them to access critical resources, thereby compromising an whole environment.

A session comprises a collection of transactions and events associated with a single user within a particular time frame. Application servers rely on session IDs (URL parameters, tokens, and cookies) to eliminate the need for repeated access validation once a user is successfully authenticated. Incorrect implementation of session persistence typically leads to a leakage of the session ID, which hackers leverage to assume the permissions of a regular user. In addition, when the application fails to validate the identity of users through multi-factor authentication, the resultant broken session may lead to several security vulnerabilities.

Hackers exploit these vulnerabilities to perform a wide range of malicious activities, such as:

- Privilege escalation
- Session fixation
- Money fraud
- Denial of service

Inclusion of unnecessary features

Unused features or services expand the attack surface and are considered one of the most exploited attack vectors. As unnecessary features add to the effort overhead of security and QA teams, they are mostly ignored during security testing and software updates as those are considered to bring no value to the application stack. Some unused services commonly abused as attack vectors include unused ports, dummy accounts, pages, and privileges.

SECURITY MISCONFIGURAITON VULNEREABILITY - SEVERITY LEVEL

Having moved up one position from the 2017 list, security misconfiguration vulnerabilities are ranked number#5 on the OWASP Top 10 list of 2021 and are mapped to 20 Common Weaknesses and Enumerations (CWEs). As of 2021, 89.58% of applications tested by security researchers had some form of security misconfiguration, with an average incidence rate of 4%.

Without a repeatable hardening process, most systems are at risk of security misconfigurations, with an average weighted exploit of 8.12 (very high) and an average impact score of 6.56 (moderate).

A successful security misconfiguration attack often leads to various other forms of attacks, including:

- Man-in-the-middle attacks
- Distributed Denial-of-Service
- Privilege escalation attacks
- Account takeovers
- Ransomware attacks
- Security compliance breaches
- Money fraud

SECURITY MISCONFIGURAITON PREVENTION TECHNIQUES

Techniques to avoid security misconfigurations include:

Frequent software patches

Outdated software components often contain known vulnerabilities commonly targeted by hackers. Implementing a consistent patch management process involves updating various component versions used in the deployment to mitigate risks of vector attacks. Updating the software also ensures that enterprises take advantage of new security features for secure application development and deployment.

Automated vulnerability scanning

Continuous vulnerability scanning helps identify common vulnerabilities as soon as they arise. Before deploying applications in production environments, security teams can leverage automated vulnerability scanners to ensure application code does not retain unused features, expose session IDs, contain sensitive data in configuration files, and include compromised default credentials for robust security.

HOW TO IDENTIFY SECURITY MISCONFIGURATION WITH CRASHTEST SECURITY?

Crashtest Security Suite enables security teams to detect and identify security misconfigurations through automated vulnerability scanning and penetration testing. The platform identifies misconfiguration blind spots using a wide range of vulnerability scanners that imitate the activities of a bug bounty hunter.

The security suite comprises various vulnerability scanners to help identify security misconfigurations, including:

- API security scanner - Detects and identifies security misconfigurations in application programming interfaces
- OWASP scanner - Benchmarks all service and device configurations against the OWASP Top 10
- CSRF scanner - Helps identify session management flaws
- Privilege escalation scanner - Used to identify access control flaws
- Port scanner - Used to identify open ports as an unnecessary feature

Crashtest Security‘s automated penetration testing also helps simulate various malicious activities commonly orchestrated by hackers by exploiting security misconfigurations. By eliminating the manual processes involved in ethical hacking, Crashtest Security Suite reduces effort and budget overhead on black box penetration testing.

BEST PRACTICES IN PREVENTING SECURITY MISCONFIGURATION

Though recommended practices may differ with varying use cases, here are some commonly suggested practices for secure configuration:

ENFORCE A REPEATABLE HARDING PROCCESS

Utilize automated tools to ensure that not only configurations across all environments are synchronized, but they also use different authorization mechanisms. Along with streamlining the deployment of secure applications across different environments, the practice also reduces the chance of a full-blown attack across all environments of the application stack.

PERFORM REGULAR AUDITS

Inspect deployments frequently to identify unnecessary features, incorrectly elevated privileges, and other issues that can lead to security misconfiguration attacks. Frequent audits also eliminate the lack of performance visibility in most modern, large-scale deployment environments.

UTILIZE SEGMENTED, MICROSERVICE-BASED ARCHITECTURE

Build a robust architecture that ensures secure separation between components used in the entire network. Secure isolation reduces the blast radius of an insecure configuration attack, keeping crucial assets safe even if vulnerabilities are present within some sections of the framework.

AVOID UNNECESSARY FEATURES

Build a minimal platform that eliminates unused components, features, documentation, and applications. This approach also minimizes the application's attack surface and reduces the workload on security and testing teams to administer security controls.

DETECT AND PREVENT SECURITY MISCONFIGURATIONS WITH CRASHTEST SECURITY

Crashtest Security Suite enforces a repeatable hardening process through test automation and comprehensive vulnerability scanning. The platform integrates seamlessly with most modern development frameworks, allowing developers to build secure applications from the ground up.

To know more about how Crashtest Security can help enhance your deployment's security hygiene, try a free 14-day trial here.

Start 2-Week Trial for Free

CRASHTEST **SECURITY**

WWW.CRASHTEST-SECURITY.COM