

Cyber Security Industry Standards



NIST & National Vulnerability Database (NVD)

- US Government standards based vulnerability management data
- Originally created in 1999 (called Internet Categorization of Attacks Toolkit or ICAT
- Provides common language (taxonomy) for analyzing, scoring, and classifying vulnerabilities.
 - Common Weakness Enumeration (CWE)
 - List of software and hardware weakness types that serves as a baseline for weakness identification, mitigation, and prevention efforts.
 - Common Vulnerabilities and Exposures (CVE)
 - Known vulnerability database for specific code bases, such as software applications or open source libraries
 - Common Weakness Scoring System (CWSS)
 - Provides a mechanism for prioritizing software weaknesses in a consistent, flexible, open manner.



CVE Program

- Maintained by MITRE Corporation, sponsored by Department of Homeland Security (DHS), Cybersecurity and Infrastructure Security Agency (<u>CISA</u>) Division
- CVE IDs are primarily assigned by MITRE
 - Also by authorized CVE Numbering Authorities (CNAs) such as corporations or
- Maintains a centralized, searchable database of known vulnerabilities
 - All information contained in the project is publicly available to any interested party.
- Provides a common means of discussing and researching exploits.
- CVE IDs are used by vendors and cybersecurity personnel for research and the identification of new vulnerabilities.
- The program do not assist in mitigating or patching vulnerabilities on the CVE list
- Format for CVE IDs is: CVE-[4 Digit Year]-[Sequential Identifier]



CVE-2021-33228 Log4J JNDI Vulnerability

班CVE-2021-44228 Detail

UNDERGOING REANALYSIS

This vulnerability has been modified and is currently undergoing reanalysis. Please check back soon to view the updated vulnerability summary.

Current Description

Apache Log4j2 2.0-beta9 through 2.15.0 (excluding security releases 2.12.2, 2.12.3, and 2.3.1) JNDI features used in configuration, log messages, and parameters do not protect against attacker controlled LDAP and other JNDI related endpoints. An attacker who can control log messages or log message parameters can execute arbitrary code loaded from LDAP servers when message lookup substitution is enabled. From log4j 2.15.0, this behavior has been disabled by default. From version 2.16.0 (along with 2.12.2, 2.12.3, and 2.3.1), this functionality has been completely removed. Note that this vulnerability is specific to log4j-core and does not affect log4net, log4cxx, or other Apache Logging Services projects.

- Hide Analysis Description

rces

Analysis Description

Apache Log4j2 2.0-beta9 through 2.15.0 (excluding security releases 2.12.2, 2.12.3, and 2.3.1) JNDI features used in configuration, log messages, and parameters do not protect against attacker controlled LDAP and other JNDI related endpoints. An attacker who can control log messages or log message parameters can execute arbitrary code loaded from LDAP servers when message lookup substitution is enabled. From log4j 2.15.0, this behavior has been disabled by default. From version 2.16.0 (along with 2.12.2, 2.12.3, and 2.3.1), this functionality has been completely removed. Note that this vulnerability is specific to log4j-core and does not affect log4net, log4cxx, or other Apache Logging Services projects.



QUICK INFO

CVE Dictionary Entry:

CVE-2021-44228

NVD Published Date:

12/10/2021

NVD Last Modified:

04/19/2022

Source:

Apache Software Foundation

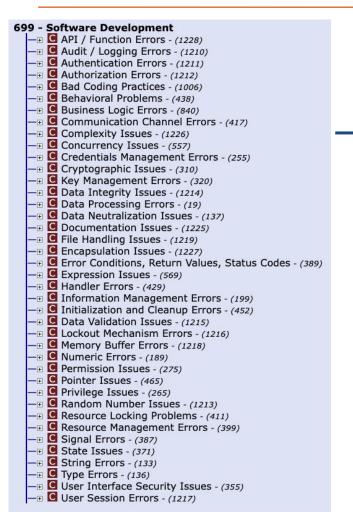
NIST CVE Database Entry

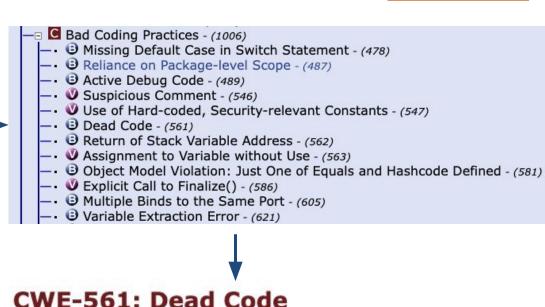
MITRE CVE Database Entry



- Common language for describing and communicating software and hardware weaknesses types
- Community driven to define concise and specific weakness types
- List is hierarchy in design for both software and hardware. Revised on an ongoing basis as threat landscape and software/hardware architectures evolve.







Weakness ID:	561			
Abstraction: Base				
Structure: Simple				
Presentation Filter:	Complete	~		

Description

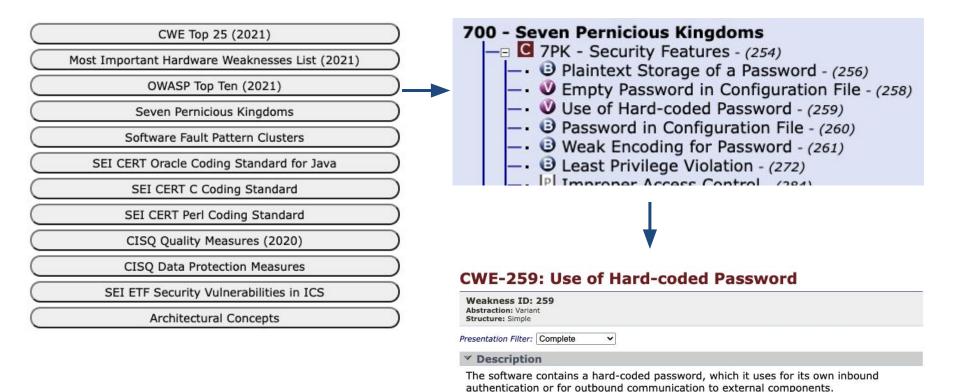
The software contains dead code, which can never be executed.

Extended Description

Dead code is source code that can never be executed in a running



External Mappings & Lists





Helpful References

Introduced During Design
Introduced During Implementation
Quality Weaknesses with Indirect Security Impacts
Software Written in C
Software Written in C++
Software Written in Java
Software Written in PHP
Weaknesses in Mobile Applications
CWE Composites
CWE Named Chains
CWE Cross-Section
CWE Simplified Mapping
CWE Entries with Maintenance Notes
CWE Deprecated Entries
CWE Comprehensive View
Weaknesses without Software Fault Patterns
Weakness Base Elements

Nature	Type	ID	Name
HasMember	V	5	J2EE Misconfiguration: Data Transmission Without Er
HasMember	V	6	J2EE Misconfiguration: Insufficient Session-ID Lengt
HasMember	V	7	J2EE Misconfiguration: Missing Custom Error Page
HasMember	V	95	Improper Neutralization of Directives in Dynamically
HasMember	V	102	Struts: Duplicate Validation Forms
HasMember	V	103	Struts: Incomplete validate() Method Definition
HasMember	V	104	Struts: Form Bean Does Not Extend Validation Class
HasMember	V	105	Struts: Form Field Without Validator
HasMember	V	106	Struts: Plug-in Framework not in Use
HasMember	V	107	Struts: Unused Validation Form
HasMember	V	108	Struts: Unvalidated Action Form
HasMember	W)	109	Struts; Validator Turned Off

CWE-5: J2EE Misconfiguration: Data Transmission Without Encryption

Weakness ID: 5 Abstraction: Variant	
Structure: Simple	
Presentation Filter: Complete	
W Becoulables	

Description

Information sent over a network can be compromised while in transit. An attacker may be able to read or modify the contents if the data are sent in plaintext or are weakly encrypted.



Common Vulnerability Scoring System (CVSS)

- Open framework for communicating the characteristics and severity of software vulnerabilities
 - Provides a numerical (0-10) representation of the severity of an information security vulnerability
 - Maintained by Forum of Incident Response and Security Teams (FIRST) comprised of 500+ member organizations.
- CVSS Measures Severity, not Risk
- A standardized scoring system provides the ability for software developers to prioritize issues so they can investigate and fix the highest risk items.