# CWE Detail – CWE-295

## Description

The product does not validate, or incorrectly validates, a certificate.

## Extended Description

When a certificate is invalid or malicious, it might allow an attacker to spoof a trusted entity by interfering in the communication path between the host and client. The product might connect to a malicious host while believing it is a trusted host, or the product might be deceived into accepting spoofed data that appears to originate from a trusted host.

## Threat-Mapped Scoring

Score: 0.0

Priority: Unclassified

## Observed Examples (CVEs)

**•** CVE-2019-12496: A Go framework for robotics, drones, and IoT devices skips verification of root CA certificates by default.

**•** CVE-2014-1266: chain: incorrect "goto" in Apple SSL product bypasses certificate validation, allowing Adversary-in-the-Middle (AITM) attack (Apple "goto fail" bug). CWE-705 (Incorrect Control Flow Scoping) -> CWE-561 (Dead Code) -> CWE-295 (Improper Certificate Validation) -> CWE-393 (Return of Wrong Status Code) -> CWE-300 (Channel Accessible by Non-Endpoint).

**•** CVE-2021-22909: Chain: router's firmware update procedure uses curl with "-k" (insecure) option that disables certificate validation (CWE-295), allowing adversary-in-the-middle (AITM) compromise with a malicious firmware image (CWE-494).

**•** CVE-2008-4989: Verification function trusts certificate chains in which the last certificate is self-signed.

**•** CVE-2012-5821: Web browser uses a TLS-related function incorrectly, preventing it from verifying that a server's certificate is signed by a trusted certification authority (CA)

**•** CVE-2009-3046: Web browser does not check if any intermediate certificates are revoked.

**•** CVE-2011-0199: Operating system does not check Certificate Revocation List (CRL) in some cases, allowing spoofing using a revoked certificate.

**•** CVE-2012-5810: Mobile banking application does not verify hostname, leading to financial loss.

**•** CVE-2012-3446: Cloud-support library written in Python uses incorrect regular expression when matching hostname.

**•** CVE-2009-2408: Web browser does not correctly handle '\0' character (NUL) in Common Name, allowing spoofing of https sites.

**•** CVE-2012-2993: Smartphone device does not verify hostname, allowing spoofing of mail services.

**•** CVE-2012-5822: Application uses third-party library that does not validate hostname.

**•** CVE-2012-5819: Cloud storage management application does not validate hostname.

**•** CVE-2012-5817: Java library uses JSSE SSLSocket and SSLEngine classes, which do not verify the hostname.

**•** CVE-2010-1378: chain: incorrect calculation allows attackers to bypass certificate checks.

**•** CVE-2005-3170: LDAP client accepts certificates even if they are not from a trusted CA.

**•** CVE-2009-0265: chain: DNS server does not correctly check return value from the OpenSSL EVP\_VerifyFinal function allows bypass of validation of the certificate chain.

**•** CVE-2003-1229: chain: product checks if client is trusted when it intended to check if the server is trusted, allowing validation of signed code.

**•** CVE-2002-0862: Cryptographic API, as used in web browsers, mail clients, and other software, does not properly validate Basic Constraints.

**•** CVE-2009-1358: chain: OS package manager does not check properly check the return value, allowing bypass using a revoked certificate.

## Related Attack Patterns (CAPEC)

* CAPEC-459
* CAPEC-475

## Modes of Introduction

**•** Architecture and Design: N/A

**•** Implementation: REALIZATION: This weakness is caused during implementation of an architectural security tactic.

**•** Implementation: When the product uses certificate pinning, the developer might not properly validate all relevant components of the certificate before pinning the certificate. This can make it difficult or expensive to test after the pinning is complete.

## Common Consequences

**•** Impact: Bypass Protection Mechanism, Gain Privileges or Assume Identity — Notes:

## Potential Mitigations

**•** Architecture and Design: Certificates should be carefully managed and checked to assure that data are encrypted with the intended owner's public key. (Effectiveness: N/A)

**•** Implementation: If certificate pinning is being used, ensure that all relevant properties of the certificate are fully validated before the certificate is pinned, including the hostname. (Effectiveness: N/A)

## Applicable Platforms

**•** None (Class: Not Language-Specific, Prevalence: Undetermined)

## Demonstrative Examples

**•** In this case, because the certificate is self-signed, there was no external authority that could prove the identity of the host. The program could be communicating with a different system that is spoofing the host, e.g. by poisoning the DNS cache or using an Adversary-in-the-Middle (AITM) attack to modify the traffic from server to client.

**•** Even though the "verify" step returns X509\_V\_OK, this step does not include checking the Common Name against the name of the host. That is, there is no guarantee that the certificate is for the desired host. The SSL connection could have been established with a malicious host that provided a valid certificate.

**•** If the call to SSL\_get\_verify\_result() returns X509\_V\_ERR\_CERT\_HAS\_EXPIRED, this means that the certificate has expired. As time goes on, there is an increasing chance for attackers to compromise the certificate.

**•** Because this code does not use SSL\_get\_verify\_results() to check the certificate, it could accept certificates that have been revoked (X509\_V\_ERR\_CERT\_REVOKED). The software could be communicating with a malicious host.

**•** Note that the code does not call SSL\_get\_verify\_result(ssl), which effectively disables the validation step that checks the certificate.