CWE-90 : LDAP Injection

This guide will describe LDAP Injection in more detail.

Contents:

LDAP Injection (CWE-90)

> **CWE-90** describes **LDAP Injection** as follows:
>
> "The software constructs all or part of an **LDAP query** using externally-influenced input from an upstream component, but it does not neutralize or incorrectly neutralizes special elements that could modify the intended LDAP query when it is sent to a downstream component."

LDAP directory services are software applications that store and organize sensitive information. Apps typically use LDAP services for the following purposes:

- Access control (user-password verification, etc.)
- Privilege management
- Resource management

The key to exploiting LDAP through injection consists of manipulating the filters used to search into the directory services.

> The goal of LDAP injection attacks is to inject LDAP search filters metacharacters in a query which will be executed by the application.

A successful exploitation of LDAP injection vulnerability could allow the hacker to:

- Access unauthorized content
- Overpass application restrictions
- Add or modify objects within LDAP structure

LDAP-Injection attacks are based on the same techniques and principles of SQL-Injection attacks, i.e. the attacker takes advantage of dynamically constructed queries based on non-neutralized user input data. If the app does not properly filter input data, the attacker can inject malicious code.

To guess if an application is vulnerable to LDAP-injection simply try to send some data that would produce some invalid input. If the app returns an error means, it executed the query with your input, therefore the app is vulnerable. Remember, that this "tactical" approach (trial and error) can be substituted by Kiuwan analysis: it will show to you all LDAP injection points.

As a very basic example, let's suppose you face an app that asks for *username* and *password* in a web form. If the app is vulnerable to LDAP-injection, it reads username and password from the HTPP request and (without any filtering) builds the LDAP filter:

```
String filter = "(&(USER=" + username +  ")(PASSWORD=" + password + "))";
```

resulting in a LDAP filter like:

```
 (&(USER=username)(PASSWORD=password))
```

If the user would know a valid username of another user (for example, "user2"), an entry like

```
username="user2)(&)"
password="dddddd"
```

will produce a filter like

```
(&(USER= user2)(&))(PASSWORD=password))
```

This constitutes two filters, but only the first will be executed.

As (&) is always true, just entering a valid username, without having a valid password, will grant access to any user's data.

LDAP Injection (CWE-90) coverage by Kiuwan

In Kiuwan, you can search rules covering LDAP injection (CWE-90) filtering by

- Vulnerability Type = **Injection,** and/or
- CWE tag = **CWE:90**

Kiuwan incorporates following rules for LDAP-Injection (CWE-90) for the following languages.

To obtain detailed information on functionality, coverage, parameterization, remediation, example codes, etc., follow the same steps as described in SQL Injection.

Language	Rule code
C#	**OPT.CSHARP.LdapInjection**
Java	**OPT.JAVA.SEC_JAVA.LdapInjectionRule**
Javascript	**OPT.JAVASCRIPT.LdapInjection**
PHP	**OPT.PHP.LdapInjection**