Python static code analysis

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your PYTHON code

LDAP queries should not be vulnerable to injection attacks

Analyze your code

Vulnerability Blocker injection cwe owasp

User-provided data such as URL parameters should always be considered as untrusted and tainted. Constructing LDAP names or search filters directly from tainted data enables attackers to inject specially crafted values that changes the initial meaning of the name or filter itself. Successful LDAP injections attacks can read, modify or delete sensitive information from the directory service.

Within LDAP names, the special characters `' '`, `'#'`, `'"'`, `'+'`, `','`, `';'`, `'<'`, `'>'`, `'\'` and `null` must be escaped according to RFC 4514, for example by replacing them with the backslash character `'\'` followed by the two hex digits corresponding to the ASCII code of the character to be escaped. Similarly, LDAP search filters must escape a different set of special characters (including but not limited to `'*'`, `'('`, `')'`, `'\'` and `null`) according to RFC 4515.

Noncompliant Code Example

```
from flask import request
import ldap

@app.route("/user")
def user():
    dn =  request.args['dn']
    username =  request.args['username']

    search_filter = "(&(objectClass=*)(uid="+username+"))"
    ldap_connection = ldap.initialize("ldap://127.0.0.1:389"
    user = ldap_connection.search_s(dn, ldap.SCOPE_SUBTREE,
    return user[0]
```

Compliant Solution

```
from flask import request
import ldap
import ldap.filter
import ldap.dn

@app.route("/user")
def user():
    dn = "dc=%s" % ldap.dn.escape_dn_chars(request.args['dc'
    username = ldap.filter.escape_filter_chars(request.args[

    search_filter = "(&(objectClass=*)(uid="+username+"))"
    ldap_connection = ldap.initialize("ldap://127.0.0.1:389"
    user = ldap_connection.search_s(dn, ldap.SCOPE_SUBTREE,
    return user[0]
```

See

- OWASP Top 10 2021 Category A3 - Injection
- OWASP Top 10 2017 Category A1 - Injection

Bug

Silly equality checks should not be made

Bug

Granting access to S3 buckets to all or authenticated users is security-sensitive

Security Hotspot

Hard-coded credentials are security-sensitive

Security Hotspot

- RFC 4514 - LDAP: String Representation of Distinguished Names
- RFC 4515 - LDAP: String Representation of Search Filters
- MITRE, CWE-20 - Improper Input Validation
- MITRE, CWE-90 - Improper Neutralization of Special Elements used in an LDAP Query ('LDAP Injection')

Available In:

sonarcloud sonarqube Developer Edition