CWE-99: Improper Control of Resource Identifiers ('Resource Injection').

7.2.5 LDAP Injection

Come riconoscerla

LDAP è una base dati che censisce in forma di directory le utenze del sistema. Se l'input dell'utente viene utilizzato, senza subire alcun controllo o filtro, per comporre una query LDAP, è facilmente intuibile come possa trasformarsi in un mezzo per sferrare un attacco di LDAP injection.
Il danno che può derivarne dipende da quanto la directory delle utenze venga inquinata.
Con un attacco di LDAP injection è possibile leggere dati riservati, come è possibile modificarli, cancellarli o inserire utenze che poi possono essere utilizzate per successivi attacchi.
Se nomeUtente è "Mario Rossi", la query restituirà i dati relativi all'utente in questione, ma se viene fornito il carattere "*", verrà restituito l'intera directory di utenze.

Come difendersi

Come in altri tipi di injection è fondamentale il controllo e l'encoding dell'input, se deve servire per costruire filtri e query verso server LDAP.
L'encoding deve filtrare i seguenti caratteri: `\ # + < > , ; " =`
Altri caratteri scpeciali sono utilizzati all'interno delle query LDAP e quindi non possono essere eliminati in automatico: * ( ) . & - _ [ ] ` ~ | @ $ % ^ ? : { } ! '
Il controllo applicativo, dipendente dal contesto, assume un'importanza fondamentale.
Anche ridurre al minimo i privilegi assegnati all'utenza con la quale il server LDAP è avviato è una misura utile a minimizzare le conseguenze di un attacco.

Esempio:
Il seguente codice riceve un userid e password in input per comporre una query LDAP.

```
private void searchRecord(String userSN, String userPassword) throws
NamingException {
  Hashtable < String, String > env = new Hashtable < String, String > ();
  env.put(Context.INITIAL_CONTEXT_FACTORY,
"com.sun.jndi.ldap.LdapCtxFactory");
  try {
    DirContext dctx = new InitialDirContext(env);
    SearchControls sc = new SearchControls();
    String[] attributeFilter = {
      "cn",
      "mail"
    };
    sc.setReturningAttributes(attributeFilter);
    sc.setSearchScope(SearchControls.SUBTREE_SCOPE);
    String base = "dc=example,dc=com";
    // The following resolves to (&(sn=S*)(userPassword=*))
    String filter = "(&(sn=" + userSN + ")(userPassword=" + userPassword +
"))";
    NamingEnumeration << ? > results = dctx.search(base, filter, sc);
    while (results.hasMore()) {
      SearchResult sr = (SearchResult) results.next();
      Attributes attrs = (Attributes) sr.getAttributes();
      Attribute attr = (Attribute) attrs.get("cn");
      System.out.println(attr);
      attr = (Attribute) attrs.get("mail");
      System.out.println(attr);
    }
    dctx.close();
  } catch (NamingException e) {
```