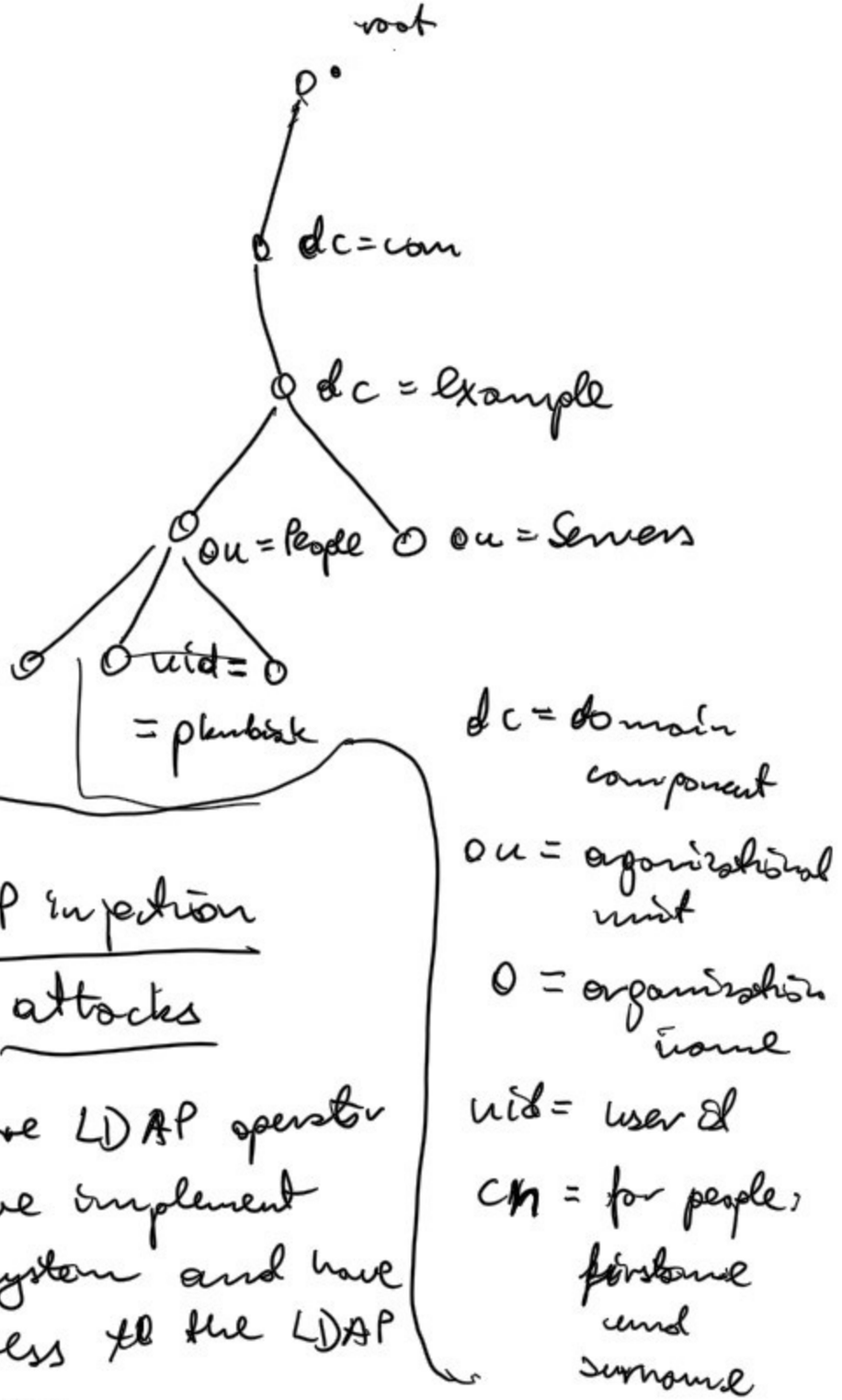
LDAP (Lightweight Directory Access Protocol)

Developed in 1993 as a lighter version of the X.500 protocols (e.g. Directory Access Protocol)

X.500 - heavy (in traffic and implementation) - so most computers (PC's) could not use it.

LDAP is much lighter, enables secure management of users in a directory:

examples:



LDAP injection attacks

We use LDAP operator or we implement a system and have access to the LDAP server

(sometimes a client must authenticate on the TLS level).

Acces controll bypass:

LDAP client constructs search filters and sends them to the server to confirm / check the existance of the pair:

(& (USER=Uname) (PASSWD= =Pwd))

If we know that a user (e.g. pkubiak) exists in the system then we can try:

(& (USER=pkubiak) (&)) (PASSWD=Pwd))

any other string

We can try more general query:

(& (USER=*) (&)) (| (cn=*) (PASSWD =hash))

Not all LDAP instances accept queries with two filters, so we can add a NULL BYTE:

(& (cn=*) (&)%00) (PASSWD= =hash))

Example 2

Web application allows to display firstname, surname for a given index number, but we know that also a phone number is stored in the LDAP.
We are going to guess it:

Legal query looks as follows:

(&(Index=[some number])(ou=
=Students))

We can check if phone numbers are stored. As an index number we enter:

84587)(telephonenumber=*

and we get the query:

(&(Index=84587)(telephonenumber=
=*)(ou=Student))

If we receive a proper answer then telephonenumber field exists.
So we can filter the phone number digit by digit:

84587)(telephonenumber=1*

↗ answer "No student with such Index"

2*
⋮
5* ← answer: OK

50*
51*

Escalation of privileges:

Assume that we are a user with low security level access rights, and in the web application form we can fill in the field:

"information" ← reports
↖ projects
etc...

So we enter:

projects)(sec_level=*))(&(directory=

=document

So as a result a filter:

(&(directory=projects)(sec_level=*))(&(directory=documents)(sec_level=low))

is sent, and the second part is ignored.

Support of DNSSEC pr X.509 slashtress:

- RFC 6698 The DNS based Authentication of Named Entries (DANE) ...

DANE allows to associate certificate with the domain - see sect. 2.1 of the RFC

So we have a kind of certificate pinning.

CAA - Certification Authority Authorization.

RFC-6844

example.com. CAA 0 issue "digicert.com"

—"— CAA 0 issue "letsencrypt.com"

—"— CAA 0 issuewild ";" ← wildcards; ; ← none is allowed

—"— CAA iodef "mailto:security@example.com"

These two RR relies on DNSSEC.

DNSSEC centralizes authentication for over 760 top-level domains and other domains under the government of each of the respective .jp, .nl, .cn, ... etc. And many of those governments do not trust each other. Even citizens of some countries do not trust their governments! Where DANE to be adopted, the mistrust would likely lead to fork/bifurcation of the Internet: US and China would run their own copies of all the name servers resolving to their own instances of sites.

So political mistrust is slowing the adoption of ~~DNSSEC~~ DANE and probably also of DNSSEC.

Content-Agnostic Detection of Phishing (October 2022)

- CT-framework
- passiveDNS
- whois data - registers maintained by the subjects responsible for assigning domains to client

In the paper a three sets of websites were selected: