HRVATSKA AKADEMSKA I ISTRAŽIVAČKA MREŽA
CROATIAN ACADEMIC AND RESEARCH NETWORK

Napadi na LDAP umetanjem znakova

CCERT-PUBDOC-2003-08-34

CARNet CERT u suradnji s LS&S

Sigurnosni problemi u računalnim programima i operativnim sustavima područje je na kojem CARNet CERT kontinuirano radi.

Rezultat toga rada ovaj je dokument koji je nastao suradnjom CARNet CERT-a i LS&S-a, a za koji se nadamo se da će Vam koristiti u poboljšanju sigurnosti Vašeg sustava.

CARNet CERT, www.cert.hr - nacionalno središte za **sigurnost računalnih mreža** i sustava.

LS&S, www.lss.hr - laboratorij za sustave i signale pri Zavodu za elektroničke sustave i obradbu informacija Fakulteta elektrotehnike i računarstva Sveučilišta u Zagrebu.

Sadržaj

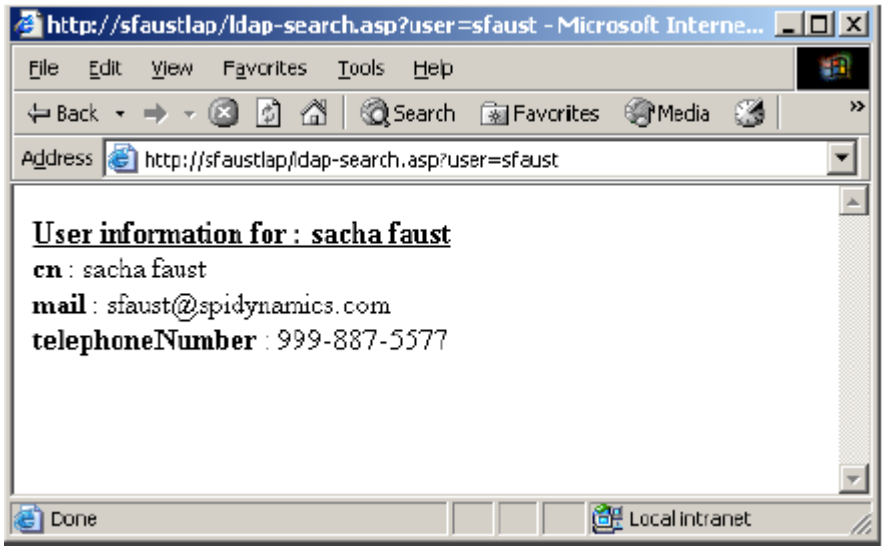
1. Uvod

Lightweight Directory Access Protocol (LDAP) je protokol koji se koristi za pristup informacijama organiziranim u imenike (engl. *directory*). Napadi na LDAP umetanjem znakova temelje se na iskorištavanju nedostataka u Web aplikacijama, koje dozvoljavaju unos klijentskih podataka u LDAP naredbe bez prethodne provjere i eliminacije potencijalno zlonamjerno ubačenih znakova iz upita. Ovaj dokument opisuje načine i mogućnosti izvođenja ovakvih napada, a isto tako opisuje i preventivne mjere koje je potrebno poduzeti da bi se vjerojatnost uspjeha takvih napada svela na minimum.

Primjeri opisani u ovom dokumentu temelje se na Microsoft ASP tehnologiji. Kao Web poslužitelj korišten je IIS, dok je kao LDAP poslužitelj korišten SunOne Directory Server.

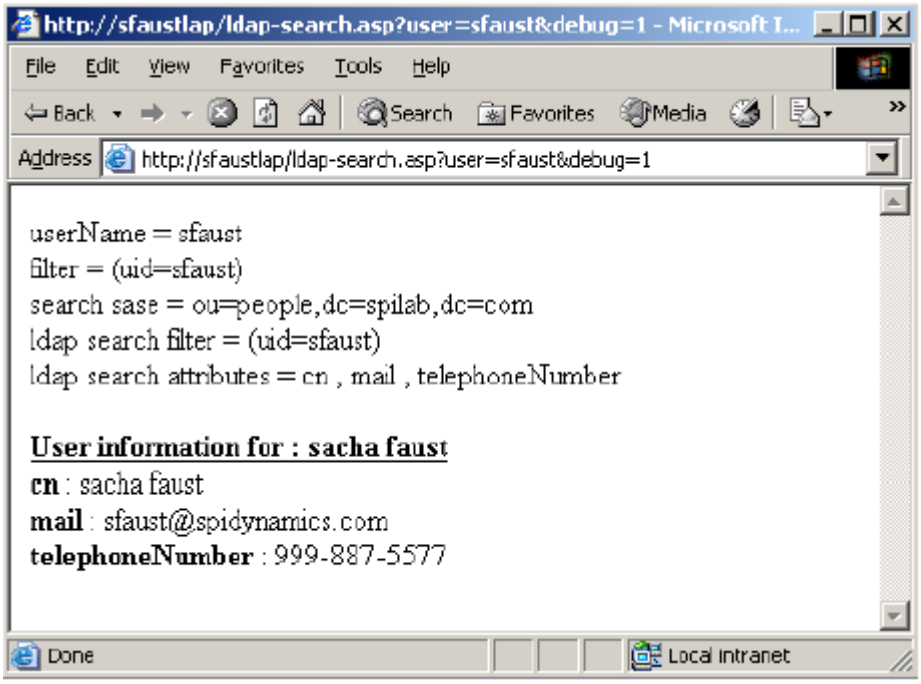
2. Oblik LDAP upita

Za razumijevanje napada na Web aplikacije koje koriste LDAP potrebno je poznavati osnove LDAP upita; kako se oni oblikuju i što vraćaju kao rezultat. Aplikacija koja se koristi kroz dokument kao primjer dana je u dodatku na kraju dokumenta (Dodatak A). U aplikaciji je implementiran upit koji uzima argument "user", te pretražuje LDAP imenik tražeći korisnički `cn` (engl. *common name*) atribut, isto kao i atribute koji definiraju korisničku e-mail adresu i telefonski broj. Kada su odgovarajući podaci pronađeni, aplikacija ih prikazuje u Web pregledniku (*Slika 1*).



Slika 1: Prikaz rada Web aplikacije

Da bi bilo jasnije kako aplikacija oblikuje upit, može se promotriti rezultat istog upita uz korištenje *debug* načina rada. *Slika 2* prikazuje oblikovanje LDAP upita temeljeno na korisničkoj informaciji.



Slika 2: Prikaz pretraživanja u debug načinu rada

Prema prikazanim informacijama, aplikacija pretražuje `ou`=people, `dc`=spilab, `dc`=com imeničko stablo tražeći `cn`, `mail` i `telephoneNumber` atribute. Dio koji vrši filtriranje je složeniji i on je cilj napada na LDAP umetanjem znakova. Filtar koji se koristi u primjeru jest `uid=user_supplied_value`, a konkretna vrijednost je `sfaust`. Na kraju dokumenta dana je osnovna sintaksa za formiranje LDAP upita.

3. Napad na LDAP upite za pretraživanje

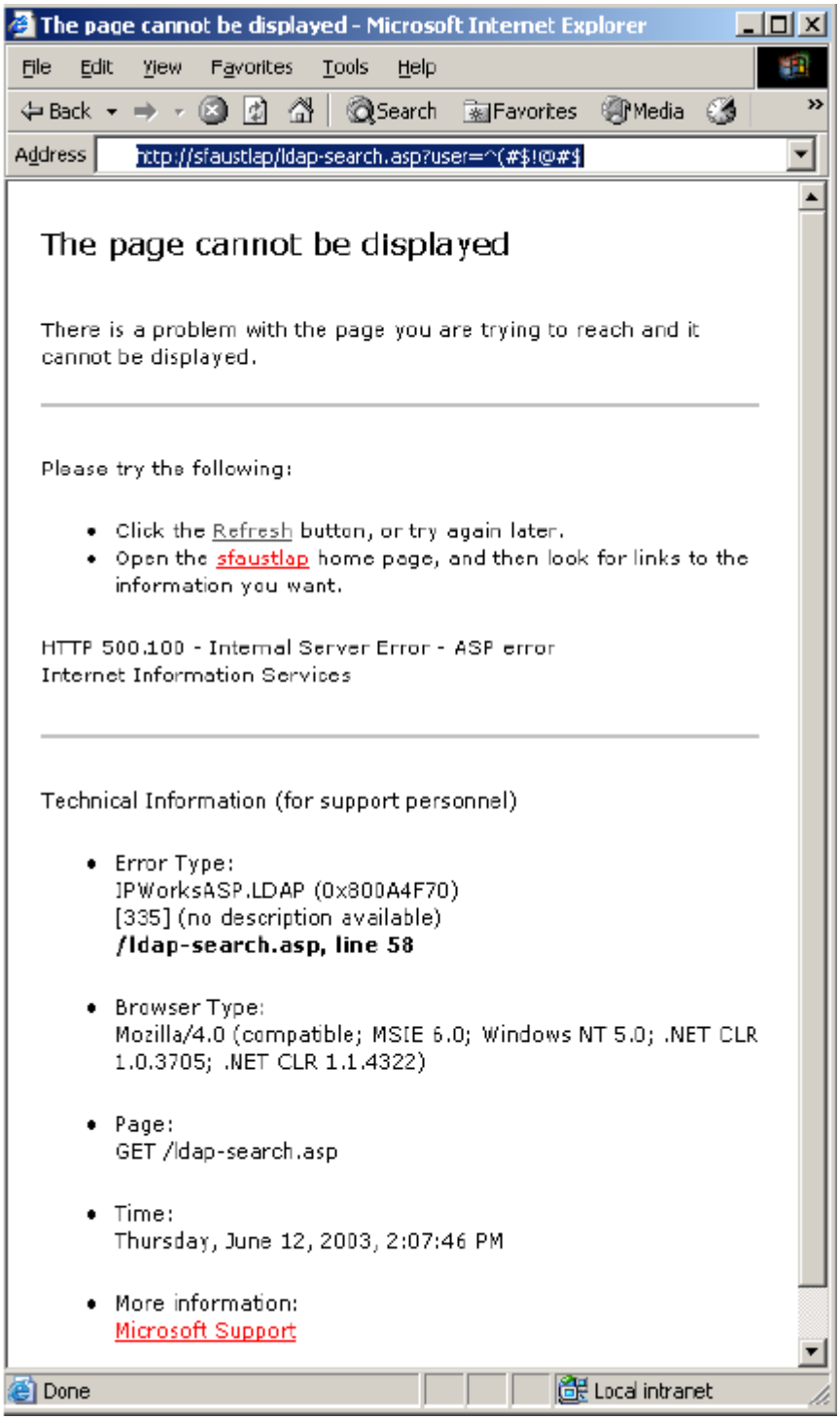
Uobičajeno korištenje LDAP-a u Web aplikacijama je omogućavanje pretraživanja specifičnih podataka na Internetu za korisnike. Na primjer, sveučilište ili fakultet može objaviti popis svih profesora ili studenata s odgovarajućim dodatnim informacijama. *Slika 1* prikazuje primjer Web aplikacije koja korištenjem LDAP-a prikazuje specifične informacije o korisniku upotrebom korisničkog imena kao parametra upita.

3.1. Oblikovanje upita

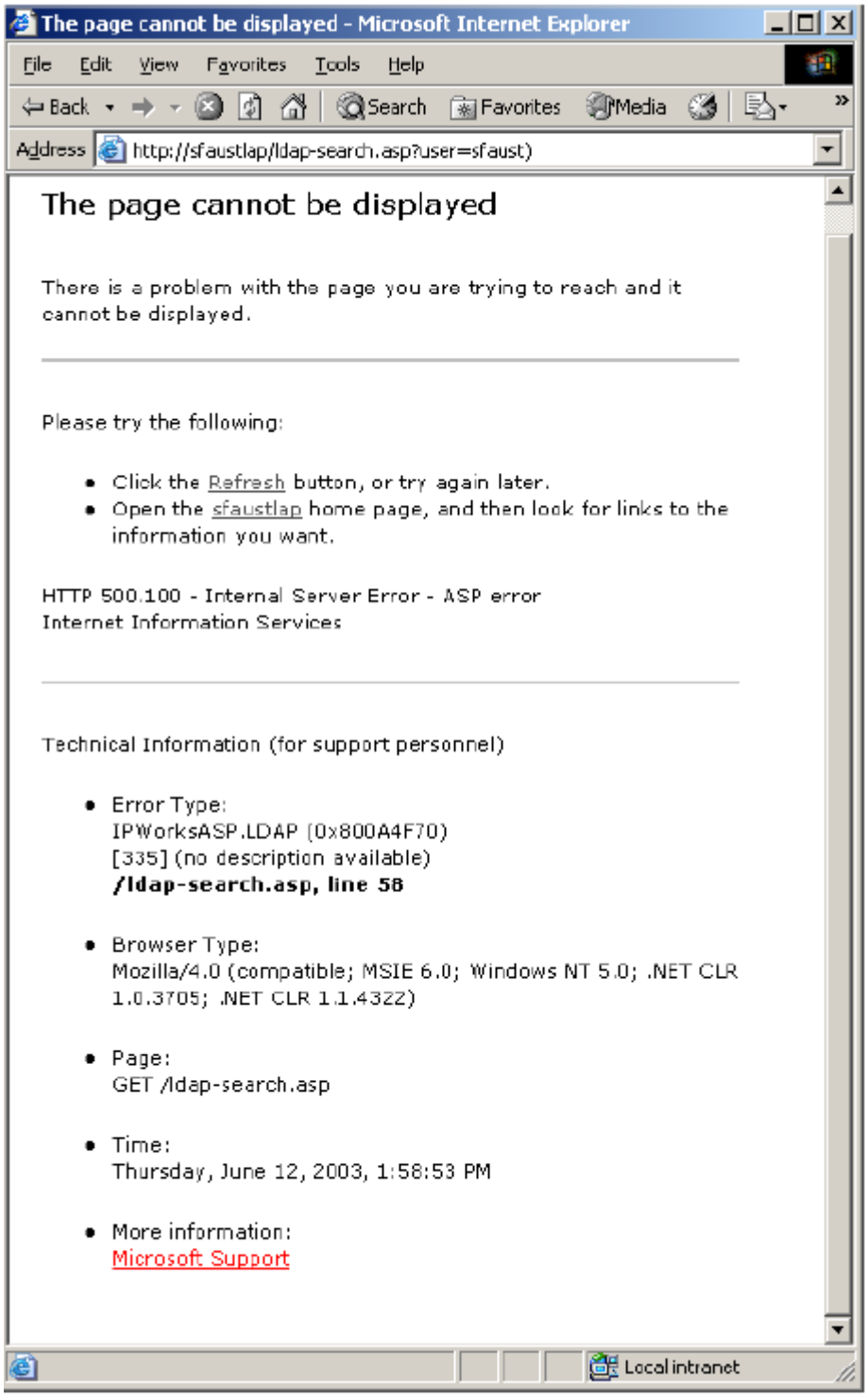
Za provođenje napada prvo je potrebno utvrditi da li aplikacija na bilo koji način provjerava podatke koje unosi korisnik. Za testiranje dovoljno je poslati nekoliko neobičnih znakova i promatrati kako aplikacija odgovara na takve upite. Slike (*Slika 3*, *Slika 4*) u nastavku prikazuju takve jednostavne upite.

U prvom primjeru šalju se podaci koje bi i najjednostavnija aplikacija trebala odbaciti. U drugom primjeru se pak šalje znak koji na prvi pogled može izgledati kako valjani dio LDAP upita. Odgovor aplikacije, kako je i vidljivo u primjerima (*Slika 3*, *Slika 4*), indicira da aplikacija ne provjerava unesene podatke nego ih direktno pohranjuje u LDAP objekt. Pošto su određeni podaci umetnuti u upit, aplikacija kao rezultat vraća pogrešku pošto je tako stvoreni LDAP upit bio neispravan.

Nakon što se utvrdi kakva provjera se provodi na strani ciljne aplikacije, napadač može pokušati predvidjeti strukturu LDAP upita ne bi li odredio kako se korisnički uneseni podaci upotrebljavaju prilikom pretraživanja. LDAP filtri za pretraživanje su uvijek odvojeni zagradama. Za lociranje podataka u znakovnom nizu za filtriranje potrebno je pokušati generirati valjani LDAP filtar dodavanjem znakova na početak ili na kraj argumenta, te promatrati odgovore na tako generirane upite.

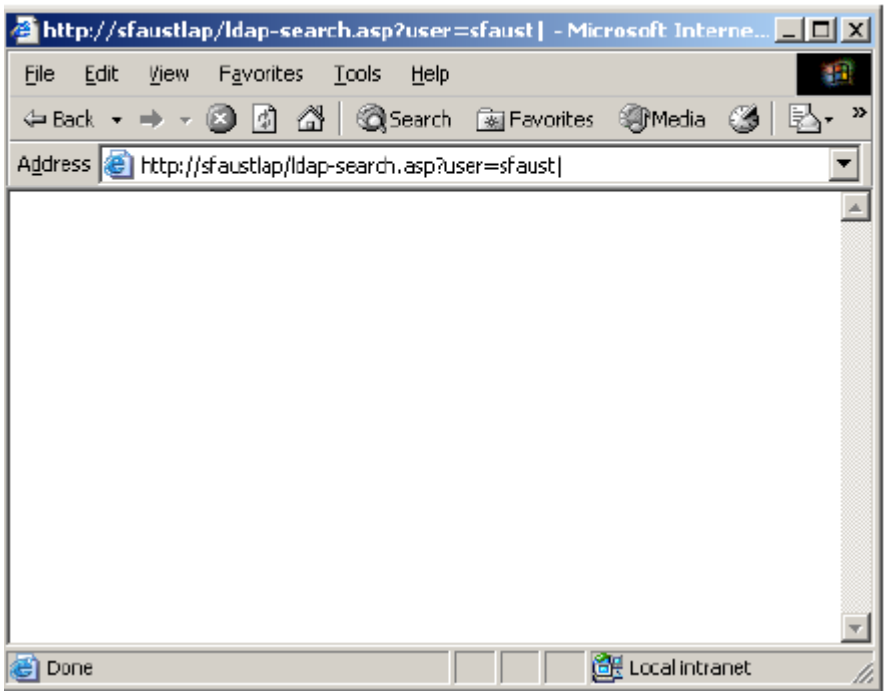


Slika 3: Prvi primjer umetanja kroz korisnički unos

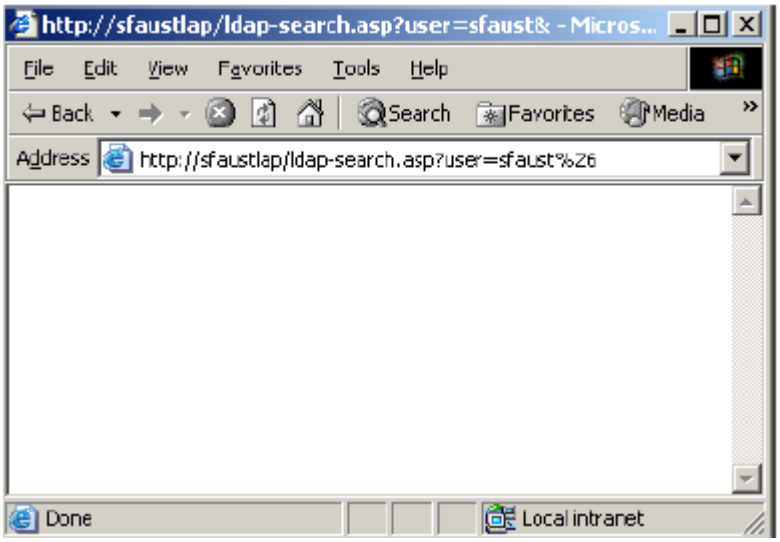


Slika 4: Drugi primjer umetanja kroz korisnički unos

Ukoliko aplikacija kao odgovor vraća pogrešku, upit očito nije ispravan. Ukoliko pak aplikacija vraća odgovor koji ne sadrži pogrešku, napadačkim unosom je stvoren valjani upit. Ovaj proces može biti vrlo dugačak i složen, ovisno o veličini upita i broju poslanih argumenata. Slike (*Slika 5*, *Slika 6*) u nastavku dokumenta prikazuju neke primjere.



Slika 5: Slanjem "|" znaka dobiva se prazan odgovor bez poruke o pogrešci



Slika 6: Slanjem "&" znaka (%26) dobiva se prazan odgovor bez poruke o pogrešci

Kako je prikazano na slikama (*Slika 5*, *Slika 6*), slanjem logičkih operatora "AND" ili "OR" moguće je vidjeti kako je upit oblikovan. Stvarni upit mijenja simbole "|" i "&" (URL-enkodiran kao %26, da bi se izbjegla kriva interpretacija) u "OR" i "AND". Pošto ciljna aplikacija ne vraća pogrešku, umetnute vrijednosti stvorile su valjani upit. Imajući to u vidu i pregledavanjem sintakse LDAP upita, napadač može zaključiti da aplikacija generira upit u sljedećem obliku:

`(neki atribut = korisnički unos).`

Umetnuta vrijednost na prvoj slici (*Slika 5*) bi generirala upit

`(neki atribut = korisnički unos|)`,

dok bi umetnuta vrijednost na drugoj slici (*Slika 6*) generirala upit

`(neki atribut = korisnički unos&)`.

Oba ova upita su valjana i kao rezultat ne vraćaju nikakvu vrijednost.

Za provjeru ovih pretpostavki može se pokušati dohvatiti `cn` vrijednost korisnika `sfaust`. To zahtijeva umetanje podataka da bi se generirao upit koji izgleda na sljedeći način:

`(neki atribut = korisnički unos)(|(cn=*))`.

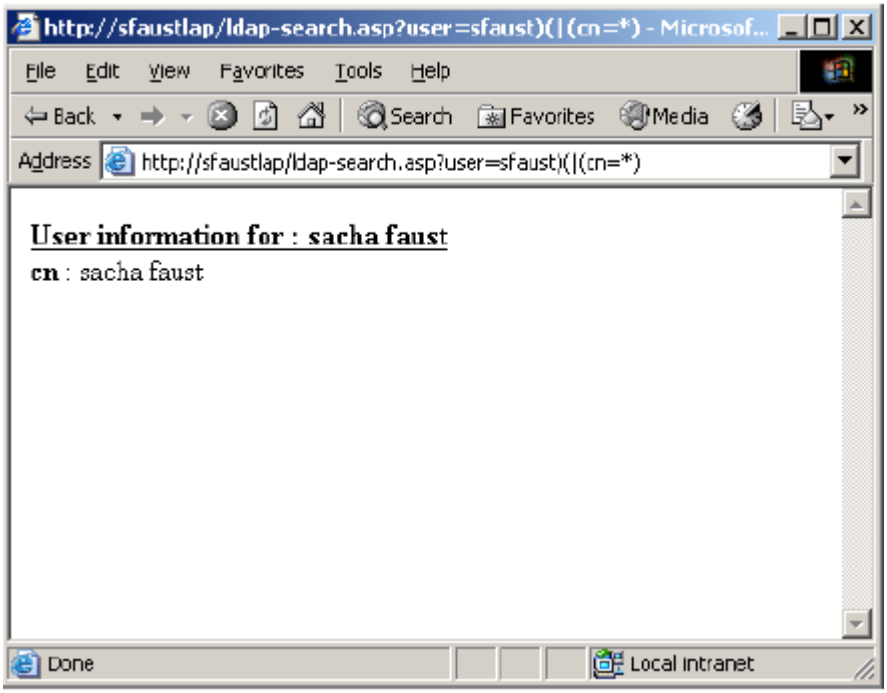
Tako oblikovani upit uzrokuje da poslužitelj vrati bilo koju cn vrijednost. Pošto je pretpostavka da upit izgleda kao

`(neki atribut = korisnički unos)`,

da bi se dobio željeni upit oblika

`(neki atribut = sfaust)(|(cn=*))`,

potrebno je umetnuti `sfaust)(|(cn=*))`. *Slika 7* prikazuje stvarni unos i rezultat upita.



Slika 7: Dohvaćanje cn vrijednosti korisnika

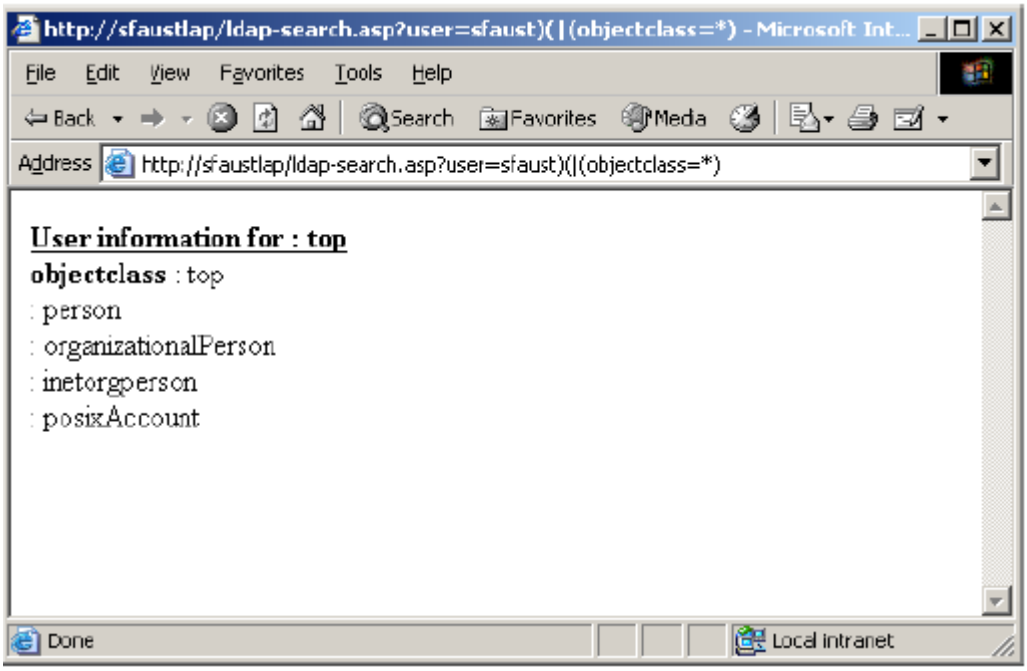
U ovom primjeru umetanje je uspješno, čime je potvrđena pretpostavka o strukturi upita.

3.2. Generiranje napada

Nakon što je određena struktura upita, napadač može generirati dodatne napade da bi pristupio drugim informacijama. Kao prvo, potrebno je otkriti koji atributi su dostupni. To se postiže generiranjem upita LDAP poslužitelju koji sadrži `objectclass` popis. Tada je dovoljno provjeriti sljedeći URL:

http://docs.sun.com/source/816-6699-10/objclass.html

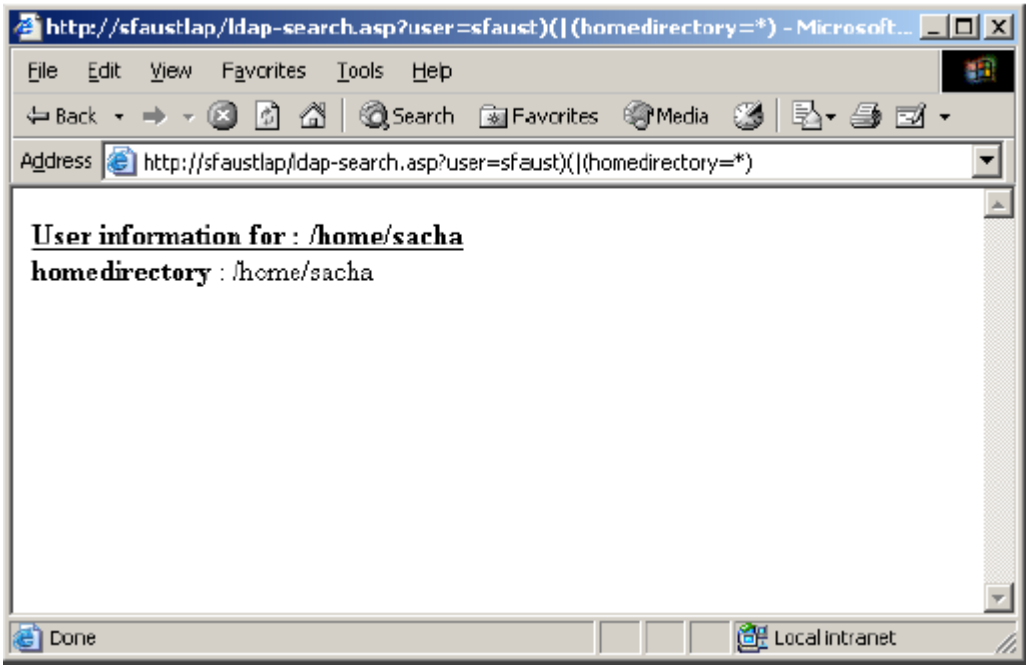
i pogledati koji atributi su uključeni u svakoj klasi. Ukoliko klasa objekta nije sadržana na gornjoj adresi, obično se može pronaći pretraživanjem Interneta. *Slika 8* prikazuje popis dostupnih klasa objekata u ovom primjeru.



Slika 8: Dohvat dostupnih klasa objekata

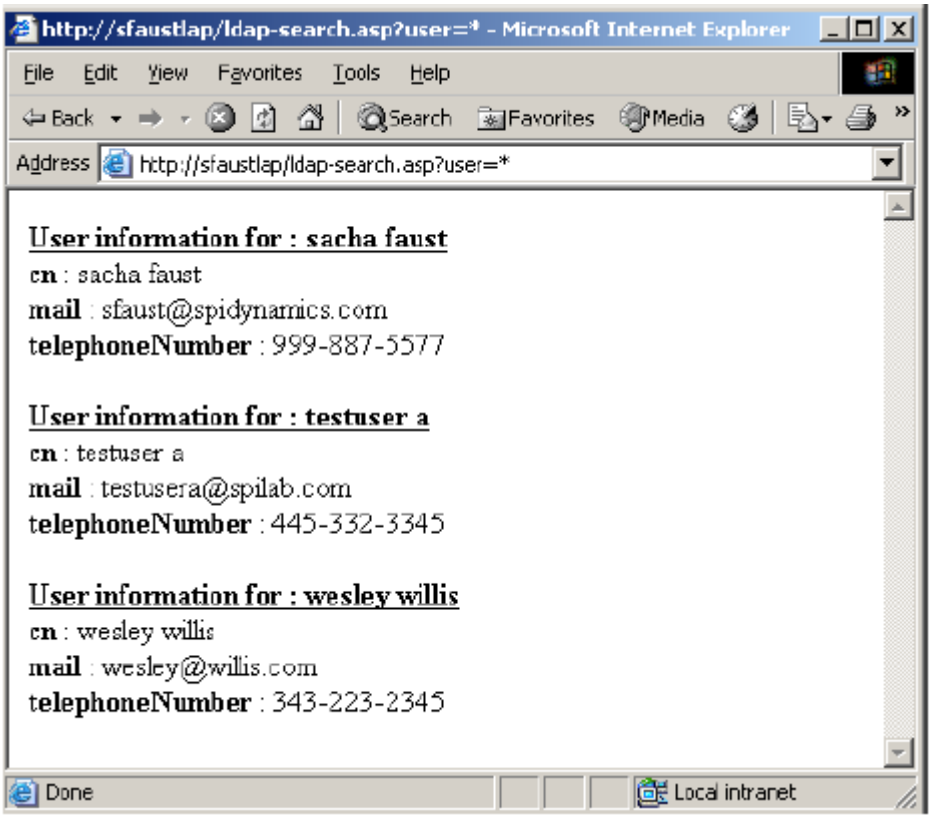
Uz popis klasa objekata, moguće je pokušati dohvatiti jedan objekt da se provjeri da li postoje dozvole za pregled podataka. U primjeru se koristi *posixAccount* klasa objekta. Pregledom definicije klase (RFC 2307) može se uočiti da su sljedeći atributi nužni: `cn`, `uid`, `uidNumber`, `gidNumber`, `homeDirectory`. Teoretski, pregled bilo kojeg od ovih atributa trebao bi biti moguć.

Slika 9 prikazuje pokušaj dohvaćanja korisničkog `home` direktorija za korisnika *sfaust*. Kako je vidljivo, pregled atributa *posixAccount* klase je moguć. Istu tehniku moguće je primijeniti na sve klase objekata te dohvaćati odgovarajuće podatke.



Slika 9: Dohvat korisničkog home direktorija

Da bi se dohvatio popis svih korisnika na sustavu (i pregledali njihovi podaci), dovoljno je koristiti zamjenski znak kao korisničko ime. *Slika 10* prikazuje takav upit i pripadajući odgovor.



Slika 10: Dohvaćanje popisa svih korisnika

4. Zaštita

Zaštita LDAP temeljenih Web aplikacija zahtijeva angažiranje programera i LDAP administratora. Ovo poglavlje opisuje postupke koji mogu poslužiti za umanjivanje rizika od napada na LDAP umetanjem znakova. No, uvijek valja imati na umu da je sigurnost Web aplikacija dinamički proces i da je nemoguće naći rješenje koje bi se moglo smatrati apsolutno sigurnim te budući da napadači mijenjaju tehnike napada, nužno je i stalno unapređivanje sigurnosti Web aplikacija.

4.1. Provjera unosa

Svaki korisnički unos mora se filtrirati, te se tako moraju filtrirati i svi znakovi ili nizovi znakova koji se mogu potencijalno zlonamjerno iskoristiti. To je potrebno provoditi u svim aplikacijama, a ne samo u onima koje služe za LDAP upite. Uklanjanje navodnika i njihovo maskiranje nije dovoljno. Najbolji način filtriranja podataka je da aplikacija predefinirano odbacuje sve znakove osim onih koji su eksplicitno dozvoljeni. Na primjer, sljedeći regularni izraz će vratiti samo brojke i slova:

```
s/[^0-9a-zA-Z]//g
```

Izrada vlastitih filtara je poželjna gdje god je to moguće. Gdje god je moguće potrebno je koristiti samo brojke. Sljedeći korak je dozvola korištenja brojki i slova. Ukoliko je zbog bilo kakvog razloga potrebno uključiti i simbole ili interpunkcijske znakove bilo koje vrste, nužno je osigurati da oni budu prevedeni u HTML zamjenske oblike (npr. `"&quote"` ili `">"`). Na primjer, ukoliko korisnik unosi vlastitu e-mail adresu elektroničke pošte, osim brojki i slova, potrebno je dodatno dozvoliti unos samo znakova `"@"`, `"_"`, `"."` i `"-"`, i to samo nakon što su ti znakovi prevedeni u HTML zamjenski oblik.

4.2. Provjera rezultata

Svi podaci koji se vraćaju kao rezultat upita trebaju se provjeriti, a količina podataka koja se vraća kao rezultat upita mora biti ograničena čime se osigurava dodatna razina sigurnosti.

4.3. LDAP konfiguracija

U LDAP imeniku je nužna implementacija stroge kontrole pristupa. To je posebno važno prilikom konfiguracije dozvola na korisničkim objektima, a još važnije ukoliko se imenički servisi koriste kao *single sign-on* rješenje. Potrebno je potpuno razumjeti kako se koristi svaka klasa objekata i odrediti da li će korisniku biti omogućeno unošenje promjena. Na primjer, dozvola da korisnici mijenjaju `uidNumber` atribut može dovesti do toga da korisnik promijeni vlastitu razinu ovlasti prilikom pristupa pojedinim sustavima. Razina ovlasti koju koristi Web aplikacija za spajanje na LDAP poslužitelj mora biti ograničena na minimalnu. Na taj način, čak i ukoliko napadač pronađe način da zaobiđe sigurnosne postavke aplikacije, nastala šteta će biti ograničena. Osim toga, LDAP poslužitelj nikad ne smije biti direktno dostupan s Interneta. Na taj način eliminira se mogućnost direktnih napada na sam poslužitelj.

Dodatak A: Izvorni kôd Web aplikacije

```
<html>
<body>
<%@ Language=VBScript %>
<%
  Dim userName
  Dim debug
  Dim filter
  Const LDAP_SERVER = "ldaptest.spilab.com" 'you need to point to your ldap
server
  debug = False
  if( Request.QueryString("debug") <> "" ) then
        debug = CBool(Request.QueryString("debug"))
  end if
  userName = Request.QueryString("user")
  if( userName = "" ) then
        Response.Write("<b>Invalid request. Please specify a valid user
name</b><br>")
        Response.End()
  end if
  if( debug ) then
        Response.Write("userName = " + userName + "<br>")
  end if
  filter = "(uid=" + CStr(userName) + ")" ' searching for the user entry
  if( debug ) then
        Response.Write("filter = " + filter + "<br>")
  end if
  Call PerformSearch(filter)
  Sub PerformSearch( filter )
        Dim ldapObj
        'Creating the LDAP object and setting the base dn
        Set ldapObj = Server.CreateObject("IPWorksASP.LDAP")
        ldapObj.ServerName = LDAP_SERVER
        ldapObj.DN = "ou=people,dc=spilab,dc=com"
        'Setting the search filter
        ldapObj.SearchFilter = filter
        'Setting the attributes we are looking for
        ldapObj.AttrCount = 3
        ldapObj.AttrType(0) = "cn"
        ldapObj.AttrType(1) = "mail"
        ldapObj.AttrType(2) = "telephoneNumber"
        if( debug ) then
              Response.Write("search sase = " & ldapObj.DN & "<br>")
              Response.Write("ldap search filter = " & ldapObj.SearchFilter
& "<br>")
              Dim searchAttrStr
              For i = 0 To ldapObj.AttrCount -1
                    if( i = 0 ) then ' for cleaner output
                          searchAttrStr = "ldap search attributes = " &
```

```
                              ldapObj.AttrType(i)
                        else
                              searchAttrStr = searchAttrStr & " , " &
ldapObj.AttrType(i)
                        end if
                Next
                if( i > 0 ) then
                        Response.Write(searchAttrStr & "<br>" )
                end if
        end if
        ldapObj.Search
        'Showing the user information
        While ldapObj.NextResult = 1
                Response.Write("<p>")
                Response.Write("<b><u>User information for : " +
ldapObj.AttrValue(0) + "</u></b><br>")
                For i = 0 To ldapObj.AttrCount -1
                        Response.Write("<b>" + ldapObj.AttrType(i) + "</b> : "
+ ldapObj.AttrValue(i) + "<br>" )
                Next
                Response.Write("</p>")
        Wend
  End Sub
%>
</body>
</html>
```

Dodatak B: Sintaksa LDAP upita

```
<filter> ::= '(' <filtercomp> ')'
<filtercomp> ::= <and> | <or> | <not> | <item>
<and> ::= '&' <filterlist>
<or> ::= '|' <filterlist>
<not> ::= '!' <filter>
<filterlist> ::= <filter> | <filter> <filterlist>
<item> ::= <simple> | <present> | <substring>
<simple> ::= <attr> <filtertype> <value>
<filtertype> ::= <equal> | <approx> | <ge> | <le>
<equal> ::= '='
<approx> ::= '~='
<ge> ::= '>='
<le> ::= '<='
<present> ::= <attr> '=*'
<substring> ::= <attr> '=' <initial> <any> <final>
<initial> ::= NULL | <value>
<any> ::= '*' <starval>
<starval> ::= NULL | <value> '*' <starval>
<final> ::= NULL | <value>
```