Usługi sieciowe w systemach Unix/Linux

Witold Paluszyński
Katedra Cybernetyki i Robotyki
Politechnika Wrocławska
http://www.kcir.pwr.edu.pl/~witold/

Konfiguracja usług sieciowych — superserwer `inetd`

```
# Syntax for socket-based Internet services:
#  <service_name> <socket_type> <proto> <flags> <user> <server_pathname> <args>
#
ftp     stream  tcp     nowait  root    /usr/sbin/in.ftpd       in.ftpd
telnet  stream  tcp     nowait  root    /usr/sbin/in.telnetd    in.telnetd
name    dgram   udp     wait    root    /usr/sbin/in.tnamed     in.tnamed
shell   stream  tcp     nowait  root    /usr/sbin/in.rshd       in.rshd
login   stream  tcp     nowait  root    /usr/sbin/in.rlogind    in.rlogind
exec    stream  tcp     nowait  root    /usr/sbin/in.rexecd     in.rexecd
comsat  dgram   udp     wait    root    /usr/sbin/in.comsat     in.comsat
talk    dgram   udp     wait    root    /usr/sbin/in.talkd      in.talkd
uucp    stream  tcp     nowait  root    /usr/sbin/in.uucpd      in.uucpd
#
# Tftp service is provided primarily for booting.  Most sites run this
# only on machines acting as "boot servers."
#
tftp    dgram   udp     wait    root    /usr/sbin/in.tftpd in.tftpd -s /tftpboot
#
# Finger, systat and netstat give out user information which may be
# valuable to potential "system crackers."  Many sites choose to disable
# some or all of these services to improve security.
#
finger  stream  tcp     nowait  nobody  /usr/sbin/in.fingerd    in.fingerd
systat  stream  tcp     nowait  root    /usr/bin/ps             ps -ef
netstat stream  tcp     nowait  root    /usr/bin/netstat        netstat -f inet
```



```
#
# These services implemented internally by inetd
#
time    stream  tcp     nowait  root    internal
time    dgram   udp     wait    root    internal
echo    stream  tcp     nowait  root    internal
echo    dgram   udp     wait    root    internal
discard stream  tcp     nowait  root    internal
discard dgram   udp     wait    root    internal
daytime stream  tcp     nowait  root    internal
daytime dgram   udp     wait    root    internal
chargen stream  tcp     nowait  root    internal
chargen dgram   udp     wait    root    internal
#
# RPC services syntax:
# <rpc_prog>/<vers> <endpoint-type> rpc/<proto> <flags> <user> <pathname> <args>
rquotad/1  tli rpc/datagram_v wait root /usr/lib/nfs/rquotad  rquotad
walld/1    tli rpc/datagram_v wait root /usr/lib/netsvc/rwall/rpc.rwalld rpc.rwalld
rstatd/2-4 tli rpc/datagram_v wait root /usr/lib/netsvc/rstat/rpc.rstatd rpc.rstatd
```


Kwestie bezpieczeństwa związane z `inetd`

Zagadnienia bezpieczeństwa pojawiają się w wielu elementach konfiguracji sieciowej, jednak kilka kwestii jest bardzo silnie związanych z serwerem usług sieciowych `inetd`:

- kontrola dostępu sieciowego za pomocą systemu zwanego *TCP wrapper*
- programy r∗

TCP wrapper

Konfiguracja serwera `inetd` zabezpieczonego przez program `tcpd` (*TCP wrapper*):

```
diablo-240> grep tcpd /etc/inetd.conf
#ftp    stream  tcp6    nowait  root    /usr/local/sbin/tcpd    in.ftpd
#telnet stream  tcp6    nowait  root    /usr/local/sbin/tcpd    telnetd
#shell  stream  tcp6    nowait  root    /usr/local/sbin/tcpd    in.rshd
tftp    dgram   udp6    wait    root    /usr/local/sbin/tcpd    in.tftpd -s /tftpboot
finger  stream  tcp6    nowait  nobody  /usr/local/sbin/tcpd    in.fingerd
```



```
diablo-237> cat /etc/hosts.allow
in.tftpd : mono :allow
ALL : katmai.magma-net.pl : twist=(/usr/bin/echo 'Indagacja udana')
ALL : meta.members.com.pl : twist=(/usr/bin/echo 'Poskanujcie sie sami')
telnetd : a06.ie.pwr.wroc.pl : allow
proftpd : 156.17.9.130 :allow
proftpd : 156.17.208.138: allow
#ALL : smietanka.t16.ds.pwr.wroc.pl : deny
# Dla dyplomanta
sshd : 195.94.196.142 : allow
sshd : .astercity.net : allow
#a tu koniec
ALL : pr168.wroclaw.sdi.tpnet.pl : twist=(/usr/bin/echo 'Zapraszam do 07 na rozmowe')
ALL : 156.17.9.0/255.255.255.128 rcf931 : allow
ALL :  UNKNOWN : twist=(/usr/bin/echo 'You must have proper DNS entry')
rpcbind : 156.17.9.0/255.255.255.128 : rfc931 : allow
#NA kompilator sun1000
rpcbind : 156.17.1.47 : rfc931: allow
rpcbind : 156.17.5.93 : rfc931: allow
rpcbind : ALL : rfc931 : deny
proftpd : 213.25.229.96 :allow
proftpd : 213.25.228.64 :allow
proftpd : 217.96.155.150 :allow
sshd : ALL : rfc931 :allow
in.rshd : a06.ie.pwr.wroc.pl :allow
ALL : ALL : rfc931 :deny
```

Usługi `r*`

Następujące polecenia pozwalają użytkownikowi posiadającemu konto na innej maszynie uniksowej (lub Unikso-podobnej) na wykonywanie operacji bez konieczności logowania się, o ile zdalna maszyna realizuje odpowiednie usługi, i uważa maszynę lokalną za **równoważną** sobie:

rcp — kopiowanie plików

rlogin — włączanie się na zdalną maszynę

rsh — wykonywanie poleceń na zdalnej maszynie przez interpreter poleceń

Usługi te wykorzystują koncepcję równoważności maszyn (ang. *host equivalence*) zakładającą, że użytkownik prawidłowo wlogowany na jednej z równoważnych maszyn, może być wpuszczony na drugą bez autoryzacji.

Równoważność maszyn może zdefiniować administrator w pliku `/etc/hosts.equiv`. Może ją również zdefiniować użytkownik dla swojego konta za pomocą pliku `~/.rhosts`.

```
$HOME/.rhosts                /etc/hosts.equiv
hostname    username         hostname
                             +           username
                             -hostname
```


Usługi `r*` — zagrożenia

Równoważność maszyn jest uznawana za niebezpieczną i niepożądaną własność, a ze względu na brak kontroli nad jej stosowaniem przez użytkowników, normalną praktyką jest wyłączanie usług `r*`, zwykle obsługiwanych przez `inetd`.

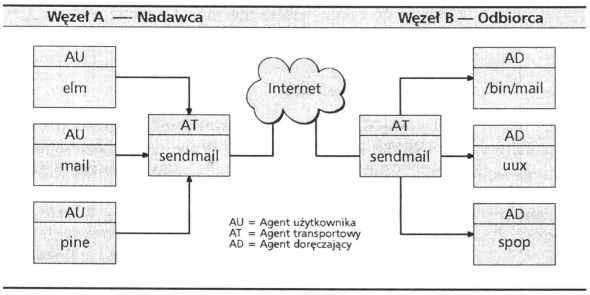
Jedną z przyczyn, dla których równoważność maszyn jest zagrożeniem, z którego administrator może nie zdawać sobie sprawy, jest niezrozumienie zasad interpretowania plików `.rhosts` i `/etc/hosts.equiv`:

- jeśli administrator wyłączy jakiś zdalny komputer lub użytkownika w pliku `/etc/hosts.equiv` to indywidualni użytkownicy i tak mogą udzielić im zezwoleń w swoich plikach `.rhosts`
- pozycja `hostname username` wpisana w pliku `/etc/hosts.equiv` jest interpretowana w ten sposób, że pozwala zdalnemu użytkownikowi `username` włączać się na konto **dowolnego** użytkownika lokalnego

Dobrym rozwiązaniem jest również stosowanie programów `scp` i `ssh` zamiast `rcp` i `rsh`, i można wręcz podłożyć programy `s*` w miejsce programów `r*` w systemie. W ten sposób efektywnie eliminujemy programy `r*` i ich odpowiadające usługi sieciowe, ale jednocześnie jakby „zachęcamy" użytkowników do stosowania bardziej bezpiecznych programów.

Przepływ poczty elektronicznej przez sieć

- UA - agent użytkownika, np. mailx, pine, mutt, outlook
- MTA - agent transferowy, np. sendmail, qmail, postfix
- DA - agent doręczający, np. lmail, procmail



System poczty elektronicznej — zagadnienia

- Protokół wymiany poczty SMTP: w założeniu dowolny komputer może przesłać pocztę do odbiorcy o dowolnym adresie, korzystając z dowolnego innego komputera jako przekaźnika.

  W ten sposób pełnosprawny serwer pocztowy jest tzw. otwartym przekaźnikiem (ang. *open relay*) poczty. Ta cecha powoduje, że, utrudnione, lub wręcz niemożliwe jest zwalczanie *spam*-u, czyli masowo rozsyłanych przesyłek reklamowych.

  Rozwiązania: możliwe, ale niezgodne z dotychczasowymi standardami.
- Inną cechą protokołu SMTP jest autentykacja klienta, a raczej jej brak. Dowolny komputer może wysłać pocztę w czyimkolwiek imieniu, przedstawiając ją jakby pochodziła od kogoś innego.

  Ponownie, rozwiązanie tego problemu w ramach istniejącego systemu wymiany poczty elektronicznej nie jest możliwe.
- Brak szyfrowania jest innym mankamentem poczty elektronicznej — listy są wymieniane w sieci otwartym tekstem. Użytkownicy mogą szyfrować treści swoich przesyłek, ale jest to uciążliwe, i przerzuca na nich problem, który jest niedostatkiem systemu poczty elektronicznej.

  Szyfrowanie pozwoliłoby rozwiązać wiele problemów poczty e-mail, przez wprowadzenie odpowiednich nowych standardów i wymagań.

Sendmail — historyczny MTA

- sendmail jest jednym z najstarszych MTA Internetu, i najstarszym aktywnie używanym
- jego głównym oryginalnym przeznaczeniem było pogodzenie różnych stosowanych konwencji i standardów z epoki, kiedy Internet funkcjonował kiepsko i był słabo ustandaryzowany
- co więcej, sendmail był napisany w realiach, kiedy Internet był względnie bezpieczny, i wysoki poziom bezpieczeństwa nie był jego celem ani cnotą
- dziś Internet działa w zupełnie innych realiach połączeniowości, niezawodności, i prędkości transmisji, i większość historycznych funkcji sendmaila nie jest potrzebna
- natomiast poziom zagrożeń sieciowych powoduje, że zagadnienia bezpieczeństwa stały się pierwszoplanowe
- dodatkowo pojawiło się zapotrzebowanie na nowe funkcjonalności, do których sendmail nie był napisany ani przystosowany, jak ochrona przed spamem
- jednak sendmail posiada na tyle ogólną i elastyczną architekturę, że został dostosowany do nowych wymagań
- jego główną wadą jest nadmierna złożoność

Sendmail — zasada działania

Historycznie sendmail słynął z dziurawych zabezpieczeń, a raczej ich braku. Jednak jest najczęściej stosowanym w systemach Unix programem MTA. Jest skomplikowanym systemem, o dużych możliwościach konfiguracji.

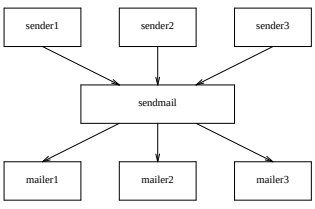


Figure 1 — Sendmail System Structure.

Dla sendmaila obojętne jest skąd pochodzi przesyłka, którą dostał. Jego zadaniem jest rozpoznać jej przeznaczenie, i przekazać ją jednemu z **mailerów**. Mailerami są, na przykład, lokalny program doręczania przesyłek do skrzynek pocztowych użytkowników (mailer local), oraz mechanizm wysyłania przesyłek do innego komputera przez sieć (mailer smtp).

Sendmail — konfiguracja (1)

Sendmail jest systemem regułowym, którego działanie sterowane jest zestawem reguł przepisujących nagłówki i/lub treść listu, jak również rozpoznających adres docelowy i wybierających odpowiedni mailer.

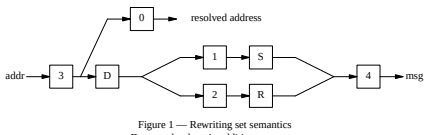


Figure 1 — Rewriting set semantics
D — sender domain addition
S — mailer-specific sender rewriting
R — mailer-specific recipient rewriting

Sendmail posiada pewne wydzielone zbiory reguł, a lokalna konfiguracja może zdefiniować dodatkowe, do zadań specjalnych. Napisanie pełnego zestawu reguł sendmaila od podstaw jest trudne, i rzadko dziś stosowane. Zamiast tego stosuje się pomocnicze systemy automatycznej generacji zestawu reguł sendmaila.

Wysyłanie poczty — sesja sendmaila

```
sendmail -bt -v -t < /tmp/testmail
```


Sendmail — operacje administracyjne

kolejka poczty

aliasy

forwarding

Sendmail — konfiguracja (2)

- „normalne" drogi przepływu poczty
- specjalne drogi przepływu poczty
- specjalne zakazy i zezwolenia
- dodatkowe konwersje adresu, np. ukrywanie części adresu
- aliasy

```
OSTYPE(hpux10)dnl
FEATURE(masquerade_envelope)dnl
MASQUERADE_AS(stud.ict.pwr.wroc.pl)dnl
MASQUERADE_DOMAIN(inyo.ict.pwr.wroc.pl)dnl
MASQUERADE_DOMAIN(diablo.ict.pwr.wroc.pl)dnl
MASQUERADE_DOMAIN(panamint.ict.pwr.wroc.pl)dnl
define(‘SMART_HOST’, smtp:diablo.ict.pwr.wroc.pl)dnl
MAILER(local)dnl
MAILER(smtp)dnl
LOCAL_NET_CONFIG
Cw stud.ict.pwr.wroc.pl
Cw inyo.ict.pwr.wroc.pl
Cw diablo.ict.pwr.wroc.pl
Cw panamint.ict.pwr.wroc.pl
```



```
OSTYPE(solaris2.ml)dnl
MASQUERADE_AS(‘ict.pwr.wroc.pl’)dnl
MASQUERADE_DOMAIN(‘palnet’)dnl
GENERICS_DOMAIN(‘ict.pwr.wroc.pl’)dnl
GENERICS_DOMAIN(‘palnet’)dnl
# te wpisy musza istniec oddzielnie chyba ze wzgledu na nocanonify
GENERICS_DOMAIN(‘sierra’)dnl
GENERICS_DOMAIN(‘shasta’)dnl
GENERICS_DOMAIN(‘shuksan’)dnl
FEATURE(masquerade_envelope)dnl
FEATURE(masquerade_entire_domain)dnl
FEATURE(genericstable)dnl
FEATURE(generics_entire_domain)dnl
FEATURE(nocanonify)dnl
FEATURE(accept_unqualified_senders)dnl
FEATURE(accept_unresolvable_domains)dnl
FEATURE(‘use_cw_file’)dnl
define(‘confDELIVERY_MODE’, ‘interactive’)dnl
define(‘confMAX_MESSAGE_SIZE’,75497472)dnl
define(‘confFORWARD_PATH’, ‘’)dnl
MAILER(smtp)dnl
MAILER(local)dnl
LOCAL_NET_CONFIG
Ct nuucp witold
R $* < @ $=w >  $#local $: $1
```

Utrzymywanie i synchronizacja czasu

Ważną kwestią we współczesnych systemach komputerowych jest kwestia utrzymywania poprawnego, i dość dokładnego czasu. Pierwszym problemem jest w ogóle konfiguracja zegara systemowego, strefy czasowej, obsługa czasu letniego, oraz synchronizacja zegara sprzętowego z zegarem systemowym. Te procedury różnią się w zależności od wersji Uniksa lub Linuksa i sprzętu, na którym pracuje system, i nie będziemy się w tym kursie w nie wgłębiać.

Drugim problemem, zwłaszcza dla systemów pracujących bardzo długo, np. wiele miesięcy, jest problem niedokładności zegara sprzętowego i wprowadzania korekt. Te niedokładności mogą wynosić wiele sekund i powodować takie problemy jak:

- trudności (lub niemożność) ustalenia dokładnej sekwencji zdarzeń analizowanych w logach systemowych (w ciągu kilku sekund mogą tam być zarejestrowane tysiące zdarzeń),
- niepoprawna praca sieciowych systemów plików (NFS), sieciowych macierzy dyskowych (SAN), itp.

W celu rozwiązania tych problemów konieczne są korekty czasu zegarowego.

Proste mechanizmy korekty czasu zegarowego

Prostym i skutecznym mechanizmem korekty czasu zegara sprzętowego jest program `hwclock` w połączeniu z plikiem `/etc/adjtime`. Pozwala on obliczyć błąd systematyczny zegara sprzętowego komputera i poprawkę dla tego błędu, a następnie okresowo korygować ustawienie zegara sprzętowego. Ponieważ błąd systematyczny zegara sprzętowego jest zwykle bardzo stabilny, ta metoda pozwala osiągnąć dużą efektywną dokładność czasu przy niedokładnym zegarze komputera, jednak pod warunkiem poprawnego i systematycznego wywoływania programu `hwclock`.

```
[root@amargosa /etc]# hwclock --adjust --debug
hwclock from util-linux-2.11y
Using /dev/rtc interface to clock.
Last drift adjustment done at 1151261947 seconds after 1969
Last calibration done at 1151261947 seconds after 1969
Hardware clock is on UTC time
Assuming hardware clock is kept in UTC time.
Waiting for clock tick...
...got clock tick
Time read from Hardware Clock: 2007/01/09 14:38:50
Hw clock time : 2007/01/09 14:38:50 = 1168349930 seconds since 1969
Time since last adjustment is 17087983 seconds
Need to insert -1791 seconds and refer time back 0.146551 seconds ago
Time elapsed since reference time has been 0.150769 seconds.
Delaying further to reach the next full second.
```



```
Setting Hardware Clock to 14:09:00 = 1168348140 seconds since 1969
ioctl(RTC_SET_TIME) was successful.
```

Inny mechanizm korekty czasu oferuje program `rdate`, który wykonuje zapytanie o czas do podanego w wywołaniu zdalnego serwera, i ustawia zgodnie z nim czas komputera lokalnego. Jest to wygodne rozwiązanie jeśli mamy w „pobliżu" inny komputer z dobrym mechanizmem synchronizacji czasu i obsługujący protokół sieciowy `time`.

Problem korekty czasu zegarowego można rozwiązać znacznie skuteczniej korzystając z serwerów czasu protokołu NTP.


```
# ntpdate -d sierra.palnet
15 May 09:11:37 ntpdate[5805]: ntpdate 3-5.93e+sun 03/06/05 23:16:45 (1.4)
transmit(172.16.0.1)
receive(172.16.0.1)
transmit(172.16.0.1)
receive(172.16.0.1)
transmit(172.16.0.1)
receive(172.16.0.1)
transmit(172.16.0.1)
receive(172.16.0.1)
transmit(172.16.0.1)
server 172.16.0.1, port 123
stratum 2, precision -15, leap 00, trust 000
refid [150.254.183.15], delay 0.02664, dispersion 0.00002
transmitted 4, in filter 4
reference time:    c812a805.12d5b000  Mon, May 15 2006  9:11:01.073
originate timestamp: c812a829.de0c0000  Mon, May 15 2006  9:11:37.867
transmit timestamp:  c812a829.c7f3b000  Mon, May 15 2006  9:11:37.781
filter delay:  0.02702  0.02679  0.02670  0.02664
                   0.00000  0.00000  0.00000  0.00000
filter offset: 0.084593 0.084692 0.084707 0.084664
                   0.000000 0.000000 0.000000 0.000000
delay 0.02664, dispersion 0.00002
offset 0.084664

15 May 09:11:37 ntpdate[5805]: adjust time server 172.16.0.1 offset 0.084664 sec

ntpdate -t 4 -v swisstime.ethz.ch sierra.palnet cyber.ict.pwr.wroc.pl
15 May 09:48:01 ntpdate[6125]: ntpdate 3-5.93e+sun 03/06/05 23:16:45 (1.4)
15 May 09:48:02 ntpdate[6125]: adjust time server 172.16.0.1 offset -0.000881 sec
```


Serwer xntpd

Program `ntpdate` pozwala korygować zegar komputera i może być wywoływany z crona. Jednak znacznie pewniejsze i dokładniejsze ustawienie czasu pozwala osiągnąć program `xntpd`, który wykorzystując szereg zaawansowanych algorytmów analizy danych może korygować czas na podstawie źródeł sprzętowych, serwerów internetowych, a także sam może być serwerem protokołu NTP.

Konfiguracja serwera `xntpd`:

```
server cyber.ict.pwr.wroc.pl
server wask.wask.wroc.pl
server vega.cbk.poznan.pl
server swisstime.ethz.ch
broadcast 224.0.1.1 ttl 4
enable auth monitor
driftfile /var/ntp/ntp.drift
statsdir /var/ntp/ntpstats/
filegen peerstats file peerstats type day enable
filegen loopstats file loopstats type day enable
filegen clockstats file clockstats type day enable
logconfig =syncstatus +sysevents
```


Obserwacja pracy serwera NTP — program `ntptrace` pokazuje ścieżkę serwerów NTP, według których bieżący serwer ustawiał aktualny czas:

```
shasta-694> ntptrace sierra
sierra: stratum 2, offset -0.001487, synch distance 0.04132
vega.cbk.poznan.pl: stratum 1, offset -0.001363, synch distance 0.00125, refid 'PPS'
```

Program `ntpq` służy do wydawania różnych zapytań i poleceń serwerowi NTP. Między innymi może wyświetlić konfigurację jego partnerów:

```
shasta-695> ntpq -p sierra
     remote           refid      st t when poll reach   delay   offset    disp
==============================================================================
 NTP.MCAST.NET   0.0.0.0         16 -    -   64    0     0.00    0.000 16000.0
 cyber.ict.pwr.w wask.wask.wroc.  3 u  921 1024  377    69.40  -10.061    4.85
+wask.wask.wroc. tik.cesnet.cz    2 u   40   64  377    71.04  -11.131   34.44
*vega.cbk.poznan .PPS.            1 u   58   64  377    62.26   -8.317   19.79
+anna.mat.uni.to ntps1-0.cs.tu-b  2 u   24   64  177    67.87   -9.495    6.97
-swisstime.ee.et swisstime2.ee.e  2 u   66  256  277    69.60    3.913   24.22
```

Znaczek w pierwszej kolumnie symbolizuje wykorzystanie informacji przez algorytm NTP: spacja i minus oznaczają odrzucenie, plus oznacza zakwalifikowanie źródła do ostatecznego zbioru serwerów, a gwiazdka wybór serwera do synchronizacji czasu.

System NFS — podstawowe koncepcje

- NFS jest sieciowym systemem plików — umożliwia współdzielenie systemów plików (albo kartotek) między komputerami.
- Schemat działania:
  - serwer NFS **eksportuje** strukturę dyskową (zwykle: system plików),
  - klient NFS **montuje** strukturę w wybranym katalogu, tak jakby to był system plików na własnym dysku fizycznym,
  - użytkownik klienta operuje na plikach, w ramach swoich uprawnień, związanych z plikiem na systemie serwera,
  - klient NFS odmontowuje strukturę, jeśli chce zakończyć użytkowanie.
- Prawa dostępu są oparte na identyfikatorach użytkownika, co wymaga jednolitego stosowania identyfikatorów w całej jednostce.
- Serwer NFS jest bezstanowy (sesja użytkownika systemu klienta może np. „przeżyć" restart serwera).
- Istnieją liczne parametry eksportu (serwera) i montowania (klienta) związane z niuansami transmisji sieciowej, praw dostępu, itp.

System NFS — podstawowe tryby pracy klienta

Klient NFS może montować system plików **hard mount** lub **soft mount**.

Tryb hard mount oznacza sztywne podłączenie systemu plików, który jest traktowany jak lokalny. Próba zamontowania musi zakończyć się pełnym sukcesem, a w czasie pracy nie mogą wystąpić żadne zakłócenia, bo może to spowodować zawieszenie programów klienta, albo w ogóle problem z wystartowaniem systemu klienta.

Tryb soft mount dopuszcza problemy z dostępnością zdalnego systemu plików, i jeśli takie wystąpią, to program otrzyma błąd operacji I/O i może kontynuować pracę, jeśli potrafi.

Jeśli chcemy, żeby system klienta działał niezależnie od problemów ze zdalnymi systemami plików, to właściwy jest tryb soft mount. Należy brać pod uwagę, że działanie programów może nie być poprawne. W pracy interakcyjnej często nie jest to problem, bo użytkownik widzi, że są problemy z siecią, i może postępować według uznania.

Jeśli chcemy zagwarantować, że system i wszystkie programy będą działały poprawnie, to musimy użyć trybu hard mount. Ten tryb jest właściwy w pracy autonomicznej (np. baza danych), gdzie nie ma komu zareagować na problemy sieciowe, i system powinien raczej zatrzymać się, niż działać dalej niepoprawnie.

System NFS — identyfikacja użytkowników i uprawnień

Identyfikacja użytkowników w systemie NFS odbywa się na podstawie numerów UID użytkowników. Numer UID jest z jednej strony wpisany jako identyfikator właściciela pliku w strukturze *i-node* tego pliku na serwerze, a z drugiej strony jest przekazywany serwerowi przez klienta wraz z operacjami dokonywanymi na pliku.

Ponadto, sprawdzanie uprawnień użytkownika do operacji wykonywanych na plikach importowanych przez NFS odbywa się w systemie klienta, tzn. serwer przyjmuje i wykonuje te operacje już bez sprawdzania.

W systemach korzystających z NFS, jak w ogóle w systemach uniksowych, numery UID i GID są głównymi mechanizmami kontroli praw dostępu do plików i katalogów, i opisane wyżej mechanizmy stosują się tak samo do numerów grup użytkowników GID. Jeśli system plików eksportowany z serwera posługuje się również listami praw dostępu ACL, a klient je rozumie i potrafi się nimi posługiwać, to są one również stosowane (również na poziomie numerycznych identyfikatorów użytkowników). Jednak format i znaczenie treści list ACL nie są ujednolicone między różnymi systemami — klient może nie rozumieć i opacznie interpretować uprawnienia wpisane w listach praw dostępu.

System NFS — konieczność wspólnej administracji

Z opisanych wyżej własności mechanizmu identyfikacji użytkowników i weryfikacji uprawnień wynikają istotne konsekwencje:

1. Musi istnieć zgodność pomiędzy serwerem a klientem co do numerów UID użytkowników korzystających z systemu NFS.
2. Musi istnieć zaufanie pomiędzy serwerem a klientem co do rzetelności numerów UID użytkowników przekazywanych razem z operacjami.
3. Dodatkowo, jeśli systemy korzystają z list ACL, to ich semantyka i interpretacja musi być zgodna między serwerem a klientem.

Poza tymi zasadniczymi właściwościami, wbudowanymi niejako w koncepcję systemu NFS, pojawia się szereg dodatkowych okoliczności, w większości kontrolowanych specjalnymi parametrami, jak np. parametry fizycznej transmisji danych, wpływające na efektywność działania systemu.

Ze względu na te okoliczności, system NFS ma zastosowanie głównie w silnie zintegrowanych jednostkach, gdzie ani wymagane zaufanie, ani ujednolicona administracja systemami, nie stanowią problemu.

System NFS — uprawnienia użytkownika `root`

Odrębną kwestią jest też, czy specjalne uprawnienia użytkownika root na jednym systemie powinny być respektowane na drugim w takim samym zakresie jak uprawnienia zwykłych użytkowników.

Po pierwsze, użytkownik `root` ma specjalne uprawnienia, i nie dotyczą go ograniczenia w prawach dostępu do plików wpisane w strukturze *i-node*. Zachodzi pytanie, czy te uprawnienia powinny również dotyczyć w systemie klienta NFS plików importowanych z innego systemu. We wszystkich nowoczesnych implementacjach identyfikator użytkownika `root` domyślnie jest odwzorowany na jakiś specjalny identyfikator, pozbawiony tych specjalnych uprawnień, ale można to ustawić parametrem `no_root_squash`.

Po drugie, w systemie mogą istnieć pliki z ustawionym bitem `set-uid`, gdzie specjalne uprawnienia są wpisane w systemie plików. Jeśli taki plik jest własnością `root`'a to stanowi obustronne zagrożenie — taki plik mógł stworzyć `root` na serwerze, i wtedy wykonanie pliku na kliencie może dać komuś specjalne uprawnienia. Jednak jeśli taki plik został utworzony na komputerze klienta, to może dać specjalne uprawnienia użytkownikowi w systemie serwera.

Konfiguracja systemu NFS — serwer

```
amargosa% cat /etc/exports
/var/spool/mail         sequoia(rw) tahoe(rw) mojave(rw)
/tmp                    sequoia(rw)


whitney% cat /etc/exports
/export/home       reksio(rw,sync) sequoia(rw,sync) tahoe(rw,sync) \
  inyo(rw,sync,no_root_squash,insecure_locks,insecure) diablo(rw,sync) \
  panamint(rw,sync) amargosa(rw,sync) becks(rw,sync) honolulu(rw,sync)
/export/home2       reksio(rw,sync) sequoia(rw,sync) tahoe(rw,sync) \
  inyo(rw,sync,no_root_squash,insecure_locks,insecure) diablo(rw,sync) \
  panamint(rw,sync) honolulu(rw,sync)
/export/mail    inyo(rw,sync,no_root_squash,insecure) diablo(rw,sync,no_root_squ
  panamint(rw,sync,no_root_squash) sequoia(rw,sync,no_root_squash)
/tftpboot/156.17.9.20          tioga.ict.pwr.wroc.pl(rw,sync,no_root_squash)
/usr                           tioga.ict.pwr.wroc.pl(ro)


diablo% cat /etc/dfs/dfstab
share -F nfs -o rw=panamint:inyo:carlsberg:miller:faxe:\
  tuborg:corona:grolsch:amstel:eb:lech:heineken:okocim:\
  zywiec,root=panamint:inyo   /var/mail
```

Monitorowanie pracy serwera NFS

Polecenie `exportfs` (Solaris: `share`) powoduje wyeksportowanie konkretnych (albo wszystkich) systemów plików zgodnie z plikiem konfiguracyjnym `/etc/exports` (Solaris: `/etc/dfs/sharetab`).

W braku opcji program wyświetla listę eksportowanych systemów plików, wraz z parametrami.

Polecenie `showmount` wyświetla listę klientów, którzy zamontowali dany system plików z serwera. Można również użyć tego polecenia z opcją `-e` aby odpytać dany serwer o listę jego eksportowanych systemów plików (od strony klienta).

Program `nfsstat` pokazuje statystyki pracy systemu NFS, części klienckiej (`-c`), albo serwerowej (`-s`). Oczywiście dany system może być jednocześnie serwerem i klientem NFS. Jednak dotyczy to różnych systemów plików. Nie można eksportować systemu plików, który dany system sam zaimportował przez NFS.

Konfiguracja systemu NFS — klient

- plik konfiguracyjny klienta NFS

```
panamint% cat /etc/fstab
...
whitney:/home      /home             nfs   defaults 0 0
diablo:/var/mail   /var/spool/mail   nfs   noexec 0 0
```

- klient może używać automountera

Automounter

Automounter jest serwerem, który zapewnia dostęp do systemów plików „na żądanie". Ma pod swoją kontrolą pewne drzewo katalogów, i próba dostępu do katalogów w tym drzewie powoduje próbę zamontowania odpowiednich systemów plików, zgodnie z konfiguracją automountera. Ponieważ najczęściej przydaje się to to współdzielenia dostępu do katalogów domowych użytkowników, oraz skrzynek pocztowych w systemach gdzie użytkownicy posiadają wspólne konto sieciowe na wielu komputerach, automounter korzysta z mechanizmu **map**, które pozwalają skonfigurować dostęp do katalogów domowych i skrzynek pocztowych w jednolity i przejrzysty sposób. Dodatkową standardową mapą automountera jest mapa **hosts**.

```
sequoia% cat /etc/auto_master
+auto_master
/net   -hosts   -nosuid,nobrowse,noexec,soft,intr,noquota
/home  auto_home  -nobrowse,nosuid
/mail  auto_mail  -nobrowse,soft,intr,noquota

sequoia% cat /etc/auto_home
witold sequoia:/export/home/witold
qc whitney:/home/qc
...
```


System NFS — wersja 4

Istnieje najnowsza wersja 4 protokołu NFS, która jest w większości niekompatybilna ze wcześniejszymi wersjami, ale dla której większość wymienionych ograniczeń i słabości systemu NFS zostało wyeliminowanych.

- implementacja oparta na połączeniach TCP na porcie 2049 zamiast systemie RPC (portmapper)
- istnieje możliwość mapowania nazw użytkowników i grup pomiędzy eksportowanymi systemami plików a identyfikatorami użytkowników
- protokół wersji 4 jest stanowy

Główną zaletą tego protokołu jest efektywność pracy przy dostępie do bardzo szybkich sieci i wydajnych serwerów. Jednak w chwili obecnej nie wszystkie systemy uniksowe obsługują NFS wersji 4.

Elementy systemu NFS

Demony (programy serwera) NFS:

- mountd — obsługuje żądania montowania systemów plików i zapewnia kontrolę dostępu. (Nieużywany w wersji NFSv4.)
- nfsd — obsługuje żądania montowania systemów plików (NFSv4)
- nfsmapid — obsługuje odwzorowanie nazw użytkowników i grup pomiędzy atrybutami plików i nazwami lokalnymi (NFSv4)
- lockd — obsługuje blokady plików
- statd — monitoruje stan sieci i wspomaga demona blokad w zwalnianiu martwych blokad po padnięciu lub restarcie klienta, i wznawianiu blokad po restarcie serwera
- nfslogd — obsługuje logowanie (NFSv2 i NFSv3)

Komunikacja z systemami Windows — Samba

Samba jest pakietem oprogramowania pozwalającym skonfigurować komputer uniksowy jako element sieci Microsoft Windows i następnie zarówno korzystać z zasobów (w terminologii windowsowej: *shares*) takich jak dyski i drukarki, udostępnianych z prawdziwych maszyn windowsowych, jak również udostępniać analogiczne własne zasoby, a także służyć sieci windowsowej jako serwer uwierzytelniania, pozwalający użytkownikom logować się z hasłem (lub bez).

Pakiet Samba składa się z następujących elementów:

smbd — demon obsługujący żądania połączeń z dyskami i drukarkami

nmbd — demon obsługujący rejestrację i żądania translacji nazw NetBIOS

smbclient — program kliencki pozwalający łączyć się z serwerami

smbpasswd,nmblookup — szereg pomocniczych programów klienckich

swat — program do konfiguracji Samby przez WWW

smbfs — moduł/driver jądra pozwalający włączać zdalnie udostępniane dyski do lokalnego systemu plików (raczej tylko na Linuksach)

Samba: konfiguracja

Istnieją narzędzia GUI do konfigurowania serwera Samby na komputerze uniksowym. Jednym z bardziej popularnych takich narzędzi jest SWAT (ang. *Samba Web Administration Tool*).

Jednak konfiguracja i administracja Sambą bezpośrednio przez plik konfiguracyjny jest dość łatwa, i ma sens nauczenie się tego i praca bezpośrednio na tym pliku. Wynika to z dwóch istotnych czynników:

1. Samba posiada pojedynczy plik konfiguracyjny (`smb.conf`), o logicznej, przejrzystej organizacji, i bardzo czytelnej treści
2. istnieje dobra dokumentacja

Główną trudnością jest znalezienie tego pliku konfiguracyjnego, jako że każda instalacja Samby, jak i każda dystrybucja Linuksa, ma inny pomysł gdzie ten plik powinien się znajdować (ale jedną z logicznych możliwości jest `/etc/samba/smb.conf`).

Samba: plik konfiguracyjny `smb.conf`

```
[global]
   workgroup = PALNET
; "security = user" requires a Unix account for every user accessing server
   security = user
   guest ok = yes
   guest account = nobody
   invalid users = root
   load printers = yes

[homes]
   browseable = no
   guest ok = no
   read only = no

[printers]
   browseable = no
   printable = yes
```


Samba: użytkownicy i uprawnienia

Pracując z systemem Samba można logować się na konkretną nazwę użytkownika, i autoryzować się przy pomocy hasła, bądź korzystać z użytkownika anonimowego (gościa). W pierwszym przypadku serwer Samby może autoryzować użytkowników zgodnie z ich identyfikatorem i hasłem uniksowym, bądź posiadać oddzielną bazę danych użytkowników i haseł. W drugim przypadku trzeba określić użytkownika systemu Unix, któremu będzie przypisany anonimowy użytkownik Samby.

Serwer Samby pozwala definiować uprawnienia (autoryzacje) na poziomie zasobów (*shares*), bądź na poziomie użytkowników. Pierwsze podejście jest gorsze, ale przydaje się we współpracy z niektórymi wersjami Windowsów, które nie potrafią podawać właściwej nazwy użytkownika i hasła. Autoryzację na poziomie użytkownika można zrealizować w prosty sposób przez nazwę użytkownika i hasło, a także korzystać z domen, albo z usługi *Active Directory*.

Samba: przykłady

```
shasta-753> smbclient -L piniek
creating lame upcase table
creating lame lowcase table
Password:

Domain=[PINIEK] OS=[Windows 5.1] Server=[Windows 2000 LAN Manager]

        Sharename       Type      Comment
        ---------       ----      -------
        OkiML321        Printer   Oki ML 321 Elite (EPSON)
        IPC$            IPC       Zdalne wywo?anie IPC
        SharedDocs      Disk
        print$          Disk      Sterowniki drukarek
        HP710C          Printer   HP DeskJet 710C
        temp            Disk
        Witold          Disk
        ADMIN$          Disk      Administracja zdalna
        C$              Disk      Domy?lny udzia?
        CD-DVD Combo    Disk
Domain=[PINIEK] OS=[Windows 5.1] Server=[Windows 2000 LAN Manager]
```



```
        Server               Comment
        ---------            -------

        Workgroup            Master
        ---------            -------

sierra-209> smbclient -L shasta
added interface ip=172.16.0.1 bcast=172.16.255.255 nmask=255.255.0.0
Password:
Domain=[PALNET] OS=[Unix] Server=[Samba 3.0.4]

        Sharename       Type      Comment
        ---------       ----      -------
        space           Disk      Public space on shasta
        IPC$            IPC       IPC Service (Samba Server)
        ADMIN$          IPC       IPC Service (Samba Server)
        hplj4k          Printer
        _default        Printer
        ps              Printer

        Server               Comment
        ---------            -------
```

```
        SHASTA               Samba Server

        Workgroup            Master
        ---------            -------
        PALNET               SHASTA

sierra-210> smbclient '\\shasta\space'
added interface ip=172.16.0.1 bcast=172.16.255.255 nmask=255.255.0.0
Password:
Domain=[PALNET] OS=[Unix] Server=[Samba 3.0.4]
smb: \> dir
  .                                   D        0  Fri Jan  4 20:06:46 2008
  ..                                  D        0  Tue Nov 20 19:39:17 2007
  lost+found                          D        0  Tue Apr  5 18:59:10 2005
...
                60492 blocks of size 524288. 1445 blocks available
smb: \> put Neostrada-parametry.ps
putting file Neostrada-parametry.ps as \Neostrada-parametry.ps (434.615 kb/s) (average
smb: \> cd Music\Misc
smb: \Music\Misc\> get aaa.mp3
getting file aaa.mp3 of size 4784796 as aaa.mp3 (433.818 kb/s) (average 433.818 kb/s)
smb: \Music\Misc\> quit
```


Samba: przykład — anonimowy serwer plików

Poniższy przykład pokazuje jak można udostępnić katalog dyskowy do zapisu i odczytu przez dowolnych użytkowników logujących się do naszego serwera.

```
[global]
  netbios name = SERWER
  security = share

[data]
  comment = All purpose disk space
  path = /space
  read only = no
  guest ok = yes
```

Typowe prawa dostępu katalogu `/tmp` są takie, że pliki i/lub podkatalogi może tam zakładać każdy, lecz usuwać może tylko użytkownik, który je założył. Na poziomie Uniksa zdalny (anonimowy) użytkownik Samby pojawi się jako użytkownik `nobody`, zatem każdy nowo logujący się użytkownik może skasować pliki wcześniej założone przez innego. Aby to zmienić musielibyśmy wprowadzić ustawienia `security = user` i `guest ok = no` oraz umożliwić autentykację użytkowników (przez nasz serwer lub inny).

Samba: przykład — serwer drukowania

```
[global]
; grupa uprawniona do administracji drukarkami, np. dodawania driverow
   printer admin = @winadmins

[printers]
; nastepujace ustawienia sa normalne w tej sekcji
   printable = yes
   browseable = no
   read only = yes
; chcemy rozliczac drukowanie wiec uzytkownicy musza sie logowac
   guest ok = no

; zapewnia drivery drukarek potrzebujacym tego klientom windowsowym
[print$]
; ogolny dostep tylko do odczytu
   read only = yes
   write list = @winadmins, root
```

Wgranie i zarejestrowanie drivera drukarki jest możliwe z klienta windowsowego (`Add Printer Wizard`), ale wymaga uprawnień użytkownika do zapisu zasobu `print$` w Sambie, oraz dostępu do używanych przezeń katalogów w Uniksie.

Samba: inne aplikacje

```
# sciaganie plikow w stylu wget
smbget -u witold 'smb://172.16.0.98/temp/ZUS_Z3-strona2.pdf'

# zarzadzanie uzytkownikami serwera Samba, lokalnie lub zdalnie
smbpasswd -r 172.16.0.98 -U witold

# skanowanie sieci Samba i wyswietlanie w postaci drzewa
smbtree

# wysylanie pliku na drukarke
smbspool {job} {user} {title} {copies} {options} [filename]
```


Samba: zabezpieczenia

- blokowanie/udostępnianie interfejsów
- blokowanie/udostępnianie określonych adresów
- blokowanie/udostępnianie określonym użytkownikom i/lub grupom
- blokowanie dostępu do całej sieci (porty: 137, 138, 139, 445)

Samba: rozwiązywanie nazw

Usługi katalogowe

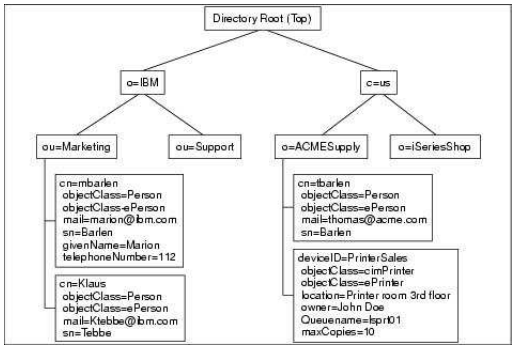
Czym różni się serwer katalogowy od serwera bazy danych:

- Dane w serwerze katalogowym są przeglądane dużo częściej, niż są modyfikowane, inaczej niż w „zwykłej" bazie danych. Typowo tysiące pobrań następują po wprowadzeniu danej.
- Dane w serwerze katalogowym są przydatne dla wielu innych użytkowników, poza ich właścicielem. Typowo istnieje użytkownik Anonymous umożliwiający niezarejestrowanym użytkownikom dostęp do bazy danych.
- Dane w serwerze katalogowym mogą być pobierane bardzo intensywnie. Serwery katalogowe są przystosowane do obsługiwania wielu (-set, lub nawet wielu tysięcy) pobrań na sekundę.
- Dla zwiększenia dostępności (i skalowalności systemów) w serwerach katalogowych stosowana jest replikacja danych. Oznacza to, że dane są duplikowane i są dostępne z alternatywnych serwerów.
- Dane w serwerze katalogowym są typowo zorganizowane w hierarchicznej strukturze drzewa, z dziedziczeniem jak w systemie obiektowym. W systemach relacyjnych dane typowo gromadzone są w tabelach.

Struktura katalogów LDAP

Katalog LDAP zawiera kolekcję obiektów ułożonych hierarchicznie, w postaci struktury drzewiastej. Katalog główny (*root*) jest punktem wyjściowym dla wszystkich danych przechowywanych w systemie.

System nazewnictwa LDAP określa zarówno sposób identyfikacji tych obiektów jak i budowę struktury drzewa. Podstawą tego systemu nazewnictwa jest **nazwa wyróżniona DN** (*distinguished name*). DN jest nazwą jednoznacznie określającą obiekt, zbudowaną jako uporządkowana sekwencja względnych nazw wyróżnionych (RDN), odpowiadających poszczególnym gałęziom drzewa katalogu, począwszy od korzenia, np.: cn=tbarlen,o=ACMESupply,c=us.



System nazw w katalogach LDAP

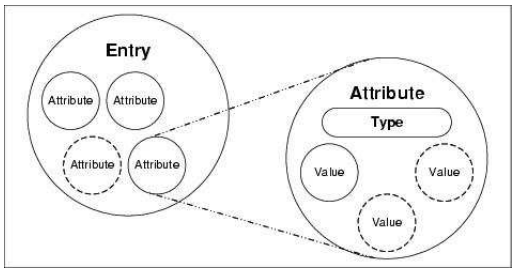
Pierwotnie idea katalogu X.500 zakładała, że w korzeniu drzewa katalogu będzie kraj. W tym systemie nazwą wyróżnioną przykładowej firmy zlokalizowanej w Polsce mogłaby być: `o=palnet, c=pl`

Rozwój Internetu spopularyzował nazwy domenowe wraz z całym systemem ich nadawania. W związku z tym pojawiła się tendencja do wykorzystania domenowych nazw organizacji w tworzeniu nazw wyróżnionych katalogów LDAP. Wtedy nazwa powyższej przykładowej firmy miałaby postać: `o=palnet.pl, c=pl`

Wadą tego systemu jest ... przywiązanie do nazwy kraju leżące u podstaw idei nazewnictwa X.500 ponieważ po pierwsze nazwa (kod) kraju się powtarza w nazwie wyróżnionej obiektu, a poza tym wiele organizacji ma charakter ponadkrajowy, i konieczność włączenia kraju do nazwy obiektu jest kłopotliwa i może być myląca. Na dodatek organizacje amerykańskie są uprzywilejowane w internetowym systemie domen (nie korzystają z kodu kraju), ale z kolei wiele organizacji nieamerykańskich korzysta z domen internetowych .com lub .org.

W związku z tym stosuje się również system nazw oparty ściśle na domenach internetowych, i tylko na nich, w postaci uporządkowanego ciągu względnych nazw wyróżnionych: `dc=palnet, dc=pl`

Model danych w katalogach LDAP




```
dn: o=palnet.pl, c=pl
o: palnet.pl
objectclass: top
objectclass: organization

dn: ou=People, o=palnet.pl, c=pl
ou: People
objectclass: top
objectclass: organizationalunit

dn: ou=Development, o=palnet.pl, c=pl
ou: Development
objectclass: top
objectclass: organizationalunit

dn: ou=Software, ou=Development, o=palnet.pl, c=pl
ou: Software
objectclass: top
objectclass: organizationalunit
```

```
dn: cn=Witold Paluszynski, ou=People, o=palnet.pl, c=pl
cn: Witold Paluszynski
sn: Paluszynski
givenname: Witold
uid: witold
mail: Witold.Paluszynski@pwr.wroc.pl
userpassword: lubudubu
title: The Boss
objectclass: top
objectclass: person
ou: Zarzad Spolki
ou: People
postalAddress:Plac Grunwaldzki 99, Wroclaw
postalCode:50-123
telephoneNumber:48 71 123457
```


Format zapisu danych LDIF

Istnieje uniwersalnie przyjęty format tekstowy zapisu obiektów katalogu LDAP. Format ten zwany jest LDIF (*LDAP Data Interchange Format*) i jest on przydatny do ładowania większej ilości danych, a także innych operacji wykonywanych np. automatycznie ze skryptów.

Struktura pojedynczego wpisu w pliku LDIF jest następująca:

```
dn: <distinguished name>
<attrtype> : <attrvalue>
<attrtype> : <attrvalue>
...
```

Wiersz w pliku LDIF może być kontynuowany przez rozpoczęcie kolejnego wiersza spacją lub tabem. Kolejne wpisy w pliku LDIF oddzielone są pojedynczym pustym wierszem (więcej niż jeden pusty wiersz traktowany jest jako logiczny koniec pliku).

Schematy danych w katalogach LDAP

Schematem danych nazywamy zestaw reguł określających:

- atrybuty dozwolone dla obiektów danej klasy
- atrybuty wymagane
- sposób porównywania wartości atrybutów, np. case-independent
- ograniczenia na wartości atrybutów, np. przedziały liczbowe

Korzystanie ze schematów pozwala na:

- utrzymanie jakości i spójności danych
- ograniczenie duplikacji danych
- atrybut klasy obiektu określa reguły schematu, do których obiekt musi sie stosować

Przeszukiwanie w katalogach LDAP

Operacja przeszukiwania katalogu LDAP wymaga określenia elementów:

baza — obiekt w drzewie katalogu od którego należy rozpocząć przeszukiwanie

zakres — część drzewa, które należy przeszukać: tylko obiekt bazowy, pojedynczy poziom pod obiektem, lub pełne poddrzewo pod obiektem

filtr przeszukiwania — warunek, który muszą spełniać obiekty wyszukiwane, zadawany jako wyrażenie logiczne na wartościach atrybutów

atrybuty zwracane — które wartości należy pokazać dla znalezionych obiektów; możliwe jest także zwrócenie samych nazw atrybutów, bez wartości

rozwiązywanie aliasów — określa czy w czasie przeszukiwania obiektu, który jest aliasem (alternatywną nazwą) innego obiektu, należy badać obiekt docelowy, czy sam obiekt aliasu

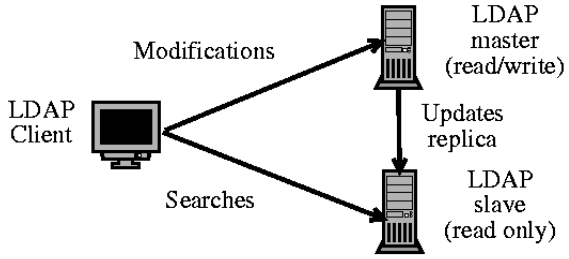
ograniczenia — można ograniczyć liczbę obiektów zwróconych w wyszukiwaniu, bądź zużyty czas; serwer może też wprowadzić swoje ograniczenia

Replikacja w katalogach LDAP

Ważnym elementem konfiguracji LDAP-a jest replikacja, dzięki której może istnieć wiele serwerów serwujących te same, zduplikowane dane. Poprawne funkcjonowanie systemu replikacji wymaga odpowiedniej konfiguracji serwerów, aby wszystkie posiadały tę samą aktualną wersję danych, oraz konfiguracji klientów, aby potrafiły odpowiednio kierować żądania do serwerów. Możliwe są różne warianty funkcjonowania takiego systemu.

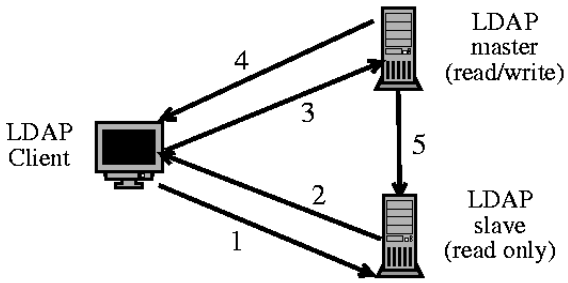
Replikacja w katalogach LDAP (2)

Model I — tylko master przyjmuje aktualizacje



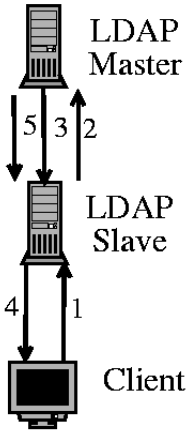
Replikacja w katalogach LDAP (3)

Model II — tylko master przyjmuje aktualizacje, ale slave odpowiada odsyłaczami



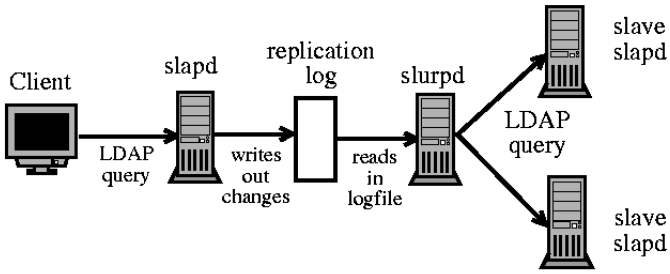
Replikacja w katalogach LDAP (4)

Model III — slave przyjmuje i poprawnie obsługuje aktualizacje



Replikacja w systemie OpenLDAP

Konfiguracja systemu replikacji — serwer slurpd:



LDAP i OpenLDAP — tutoriale internetowe

`http://quark.humbug.org.au/publications/ldap/ldap_tut.html`

`http://www.yolinux.com/TUTORIALS/LinuxTutorialLDAP.html`

`http://coewww.rutgers.edu/www1/linuxclass2012/lessons/LDAP/sec_1.php`

`http://www.redbooks.ibm.com/redbooks/SG244986/wwhelp/wwhimpl/java/html/wwhelp.htm`

Zarządzanie sieciami

W większych instytucjach istnieją rozbudowane instalacje sieciowe i pojawia się problem usprawnienia zarządzania nimi:

- wykrywanie błędów w sieciach, bramach, serwerach
- mechanizmy zawiadamiania administratora o problemach
- monitorowanie w celu podejmowania decyzji o wyrównywaniu obciążenia, rozbudowy, inwestycji
- dokumentowanie
- ułatwienie czynności administracyjnych z centralnego miejsca

Wykrywanie uszkodzeń sieci

- wynikają z: nieprawidłowych połączeń, awarii mediów lub urządzeń, przeciążenia określonych elementów
- można je skutecznie prowadzić za pomocą prostych, ogólnie dostępnych narzędzi: ping, traceroute, netstat
- ping: stan poszczególnych serwerów, drożność połączeń, opóźnienia transmisji
- traceroute: dynamiczny obraz połączeń, miejsce niedrożności sieci
- netstat: stan interfejsów pojedynczej maszyny, liczba błędów, liczba kolizji w segmencie sieci
- wartości uzyskane tymi metodami mają znaczenie względne, i aby były znaczące, trzeba wykonać pomiarów wielokrotnie, w nieregularnych odstępach czasu, i wyeliminować te spowodowane mniej istotnymi stanami chwilowymi
- pomiary te mogą być wykonywane ręcznie lub automatycznie (skrypty); w tym drugim przypadku trzeba jeszcze zorganizować mechanizm rejestracji i powiadamiania administratora

Protokoły zarządzania sieciami — SNMP

Zastosowanie protokołu zarządzania siecią wprowadza porządek i systematykę. Wszystkie urządzenia sieciowe posługują się jednym językiem, i — jeśli protokół zostanie konsekwentnie zaimplementowany — można je monitorować, konfigurować, i resetować zdalnie, z jednego miejsca.

Bardzo prostym, wprowadzonym już w latach 80-tych, ale jedynym ogólnie przyjętym protokołem zarządzania siecią jest *Simple Network Management Protocol*. Definiuje on hierarchiczną przestrzeń nazw zarządzanych danych, oraz sposób czytania i zapisu tych danych w każdym węźle.

SNMP wprowadza szereg terminów, którymi określa zarówno swoje własne elementy, jak również obiekty w zarządzanej sieci, i posiadane przez nie dane.

SNMP: podstawowe pojęcia

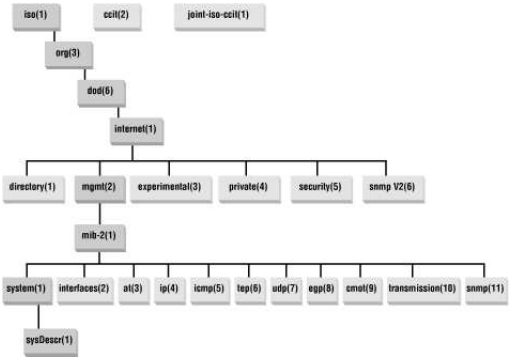
Urządzenia istniejące w zarządzanej sieci określane są jako jednostki (*network entities*), natomiast poszczególne dane w nich istniejące nazywane są obiektami. Natura poszczególnych obiektów może być różna: mogą być sprzętowymi przełącznikami na urządzeniu, programowymi rejestrami konfiguracyjnymi, licznikami, itp. SNMP jednolicie traktuje je jako obiekty i definiuje operacje, które mogą być na nich wykonywane. Każdy obiekt posiada nazwę i wartość, a więc może być uważany za zmienną (lub stałą).

Obiekty grupowane są w hierarchiczne, drzewiaste struktury, nazywane MIB (*Management Information Base*). SNMP wprowadził zestaw rozpoznawanych obiektów, na przykład, w MIB I: opis systemu, liczba interfejsów sieciowych, adresy IP poszczególnych interfejsów, liczniki pakietów przychodzących i wychodzących przez poszczególne interfejsy, oraz tablicę aktywnych połączeń TCP.

Wartością obiektu MIB może być: liczba całkowita, string, identyfikator obiektu, lub wartość pusta. Wartości mogą być grupowane w sekwencje wartości różnych typów, a sekwencje w tabele.

SNMP: drzewo MIB

Obiekty na każdym poziomie w hierarchicznym drzewie posiadają numery, jak również nazwy symboliczne. Konkretny obiekt na drzewie określony jest zatem ścieżką, którą zapisuje się z kropkami, podobnie jak domenowe adresy IP. Na przykład, jak widać poniżej, obiekt `sysDescr` zawierający tekstowy opis systemu danego urządzenia położony jest na ósmym poziomie w drzewie MIB i ma ścieżkę 1.3.6.1.2.1.1.1



Ścieżki obiektów mogą być względne lub bezwzględne. Bezwzględne ścieżki zaczynają się kropką, natomiast względne ścieżki interpretowane są tak, jakby zaczynały się od obiektu `mib-2`. Zatem poprawnymi i równoważnymi ścieżkami dla obiektu `sysDescr` są zarówno .1.3.6.1.2.1.1.1 jak i 1.1 a ponieważ możliwe jest również stosowanie nazw symbolicznych, zatem poprawną ścieżką tego samego obiektu jest też `.iso.org.dod.internet.mgmt.mib-2.system.sysDescr.0`.

SNMP: operacje

Operacje definiowane przez SNMP nazywają się w jego terminologii PDU (*Protocol Data Units*):

- `get-request` — zapytanie obiektu o wartość zmiennej
- `get-next-request` — zapytanie o wartość następnej zmiennej, np. przy sekwencyjnym listowaniu tablicy danych
- `set-request` — żądanie ustawienia wartości zmiennej
- `trap/snmpV2-trap` — konfiguruje urządzenie tak, aby zawiadamiało swoją jednostkę zarządzającą o jakimś zdarzeniu, na przykład takim jak restart urządzenia, lub przekroczenie przez jakiś licznik pewnej wartości progowej
- `response` — odpowiedź urządzenia, potwierdzenie, itp.

SNMP: przykłady

```
sequoia-475> snmpget -c public -v1 hp5 SNMPv2-SMI::mib-2.43.10.2.1.4.1.1
SNMPv2-SMI::mib-2.43.10.2.1.4.1.1 = Counter32: 4528

sequoia-476> snmpget -v 1 -c public 156.17.9.39 SNMPv2-MIB::sysContact.0
SNMPv2-MIB::sysContact.0 = STRING: Super Administrator

shasta-730> snmpget -c public -v1 printer SNMPv2-SMI::mib-2.43.11.1.1.6.1.1
SNMPv2-SMI::mib-2.43.11.1.1.6.1.1 = STRING: "Toner Cartridge HP C8061X"
shasta-731> snmpget -c public -v1 printer SNMPv2-SMI::mib-2.43.11.1.1.8.1.1
SNMPv2-SMI::mib-2.43.11.1.1.8.1.1 = INTEGER: 4500
shasta-732> snmpget -c public -v1 printer SNMPv2-SMI::mib-2.43.11.1.1.9.1.1
SNMPv2-SMI::mib-2.43.11.1.1.9.1.1 = INTEGER: 2340

sequoia-478> snmpwalk -v 1 -c public hp5
SNMPv2-MIB::sysDescr.0 = STRING: HP ETHERNET MULTI-ENVIRONMENT,ROM A.05.03,JETDIRECT,JD24,EEPROM
SNMPv2-MIB::sysObjectID.0 = OID: SNMPv2-SMI::enterprises.11.2.3.9.1
...

sequoia-481> snmpset -v 1 -c student_rw 156.17.9.39 SNMPv2-MIB::sysLocation.0 s "test"
SNMPv2-MIB::sysLocation.0 = STRING: test

sequoia-482> snmpwalk -v 2c -c public  156.17.9.56
SNMPv2-MIB::sysDescr.0 = STRING: Linux reksio 2.6.14.5 #8 PREEMPT Fri Jan 13 11:11:04 CET 2006 i
...

sequoia-483> snmpwalk -v 2c -c grupa_ro  156.17.9.56
...
```


Przykład użycia operacji `trap` (wersja 1):

```
> snmptrap -v 1 -c public  diablo:6666 ''  '' 1 2 ''

Komunikat daemona:
2006-05-24 19:56:33 10.0.0.12(via 84.40.238.85) TRAP, SNMP v1, community public
        SNMPv2-SMI::enterprises.3.1.1 Warm Start Trap (2) Uptime: 3:26:25.24
```

Operacje `trap` w wersji 2 mają inną postać w protokole i inna jest składnia wywołania polecenia `snmptrap`:

```
snmptrap -v 2c -c private diablo:6666 '' SNMPv2-MIB::warmStart

Komunikat daemona:
2006-05-24 20:02:27 xdsl-11093.wroclaw.dialog.net.pl [84.40.238.85]:
        DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (1273873) 3:32:18.73
              SNMPv2-MIB::snmpTrapOID.0 = OID: SNMPv2-MIB::warmStart
```


SNMP: uprawnienia

Pierwsze wersje SNMP (1 i 2) oparte były na prostym modelu przyznawania uprawnień, polegającym na wprowadzeniu grup zwanych *communities*.

Przynależność do grupy wymaga znajomości jej nazwy. Znajomość nazwy grupy jest również wystarczająca do udowodnienia przynależności do grupy, zatem nazwa grupy pełni rolę hasła.

Każdy obiekt w MIB posiada jeden z trzech trybów dostępu: `read-only`, `read-write`, i `none`.

W SNMP wersji 3 wprowadzono mechanizmy użytkowników, haseł i szyfrowania.

Materiały sieciowe na temat SNMP:

Strona CISCO poświęcona SNMP:
`http://www.cisco.com/en/US/tech/tk648/tk362/tk605/tsd_technology_support_sub-protocol_home.html`

Baza obiektów SNMP:
`http://tools.cisco.com/Support/SNMP/do/BrowseOID.do?local=en`