Administrowanie systemami komputerowymi
Temat : Bezpieczeństwo systemu

Krzysztof Krawczyk krafcoo@icslab.agh.edu.pl
Mariusz Topka gandalf@icslab.agh.edu.pl
Tomasz Zatorski muzzy@icslab.agh.edu.pl

9 lipca 2001

1 Wstęp

1.1 Czy potrzebujemy zabezpieczeń?

Zawsze istnieje niebezpieczeństwo, że ktoś nieupoważniony do tego przejmie informacje przesyłane przez sieć, których nie chcielibyśmy nikomu pokazać. Nawet użytkownicy naszego systemu mogą zmienić coś czego nie zamierzaliśmy robić. Nieautoryzowany dostęp do naszego systemu może być wykorzystany przez intruzów zwanych crackerami. Mogą oni ukraść ważne informacje, lub nawet odciąć nam dostęp do naszych własnych zasobów.

Zabezpieczanie systemu musi prowadzić do zmniejszenia ryzyka nieupoważnionego zmienienia danych, czy to przesyłanych przez sieć czy to składowanych w naszym systemie. Jak? Będzie to powiedziane później.

Można zapytać się jak bezpieczny może być nasz system? Należy pamiętać, że żaden system nie będzie nigdy całkowicie bezpieczny. Wszystko co można zrobić, to ciągłe utrudnianie dostępu osobom trzecim. Dla przeciętnego użytkownika systemu unixowego wystarcza niewiele, natomiast dla użytkowników zaawansowanych (banki, kompanie telekomunikacyjne) zabezpieczenie systemu wymaga dużo więcej pracy. Kolejnym czynnikiem, który należy brać pod uwagę jest to, że im bardziej system jest bezpieczny, tym większe prawdopodobieństwo tego, że ktoś będzie próbował złamać zabezpieczenia (choćby dla własnej satysfakcji).

1.2 Czego chcemy bronić?

Zanim rozpocznie się zabezpieczanie systemu należy określić poziom zagrożenia, przed którym należy się bronić, jakie ryzyko można podjąć, a jakiego nie, oraz jak bardzo podatny na ataki będzie system który otrzymamy w wyniku naszych działań. Powinno się przeanalizować system pod kątem tego co chcemy bronić i dlaczego , jaką to ma wartość oraz kto ponosi odpowiedzialność za dane i inne wartościowe rzeczy.

- *Ryzyko* to możliwość tego, że włamywacz uzyska dostęp do naszego komputera. Czy będzie on mógł czytać lub zapisywać pliki, albo wykonywać programy które mogą spowodować szkody? Czy może skasować krytyczne dane? Czy może spowodować, że my lub nasza firma nie będzie mogła ukończyć ważnych prac projektowych? Nie należy zapominać o tym, że osoba niepożądana, która uzyskała dostęp do naszego konta lub systemu może podszywać się pod nas lub innego legalnego użytkownika systemu.

  Ponadto, posiadanie jednego niezabezpieczonego konta w systemie powoduje, że cała sieć staje się narażona na atak. Jeśli pozwoli się pojedynczemu użytkownikowi korzystać z pliku *.rhosts* lub niezabezpieczonych usług takich jak np. *tftp* ryzykuje się przejęcie konta przez intruza, który posiadając konto może próbować uzyskać dostęp do innych systemów lub kont.

- *Zagrożenie* najczęściej pochodzi od osób które są zmotywowane (czasami właśnie przez nasz system zabezpieczeń) do spróbowania włamania do naszego systemu lub naszej sieci. Należy zdecydować się komu z mających dostęp do systemu można zaufać, oraz jakie zagrożenia niesie to ze sobą.

Istnieje podział intruzów na kilka typów, które warto znać gdy zabezpieczamy sysytem :

- **The Curious** - zainteresowany podstawowymi informacjami; jakiego systemu używamy i jakie dane przechowujemy.
- **The Malicious** - ten typ próbuje bądź to wyłączyć system z użycia, lub zmienić stronę WWW, lub zmusić nas do poświęcenia czasu i pieniędzy na odzyskiwanie danych, które on uszkodził.
- **The High-Profile Intruder** - próbuje użyć nasz system do uzyskania popularności i złej chwały. Może też wykorzystać zabezpieczony system do zaanonsowania swoich umiejętności.
- **The Competition** - jest zainteresowany danymi w naszym systemie. Może to być ktoś, kto myśli, że będzie mógł na tym zyskać, finansowo lub w inny sposób.
- bf The Borrowers - zainteresowany tylko w tym, żeby urządzić sobie sklep w naszym systemie i używać naszych zasobów do swoich własnych celów. Zwykle taka osoba stawia serwery chat'owe lub irc'owe, strony pornograficzne, lub nawet serwery DNS.
- **Leapfrogger** - zainteresowany tylko tym żeby uzyskać dostęp do innych systemów. Jeśli nasz system ma dobre połączenia sieciowe lub jest ruterem do wewnętrznych hostów, można się spodziewać tego typu intruzów.

- *podatność* wskazuje jak dobrze zabezpieczony jest nasz komputer od innych sieci, oraz możliwość, że ktoś uzyska nieautoryzowany dostęp do systemu.

Odzyskanie danych utraconych w wyniku ataku może zabrać dziesięć razy więcej niż czas poświęcony na zabezpieczanie systemu, a czasami może okazać się po prostu niemożliwe.

1.3 Polityka bezpieczeństwa

W średnich i dużych systemach należy ustanowić politykę bezpieczeństwa. Taka polityka powinna być łatwa do zrozumienia dla i przestrzegania. Powinna chronić dane oraz prywatność użytkowników. Rzeczy które powinniśmy rozważyć to m.in.:

- kto ma dostęp do systemu (czy mój kolega może używać mojego konta?)
- kto może instalować programy w systemie
- kto jest właścicielem poszczególnych danych
- odzyskiwanie danych po katastrofie (disaster recovery)
- właściwe używanie systemu

Ogólnie przyjętą zasadą polityki bezpieczeństwa jest :

To co nie jest dozwolone jest zabronione

Oznacza to, że dopóki nie udostępnimy danej usługi użytkownikom powinna ona być dla nich niedostępna.
Przykładową politykę można znaleźć na stronie:
http://www.faqs.org/rfcs/rfc2196.htmlSposoby zabezpieczania systemu Co się stanie z reputacją administratora jeśli intruz skasuje jakieś dane użytkownika?; lub zmieni stronę WWW?; lub opublikuje projekty firmy na następny kwartał? Jeśli planowana jest instalacja sieciowa, istnieje wiele czynników, które należy wziąć pod uwagę zanim dodamy kolejny komputer do naszej sieci. Nawet jeśli mamy zwykłe połączenie PPP lub mały serwer WWW, nie znaczy to, że włamywacze nie zainteresują się naszym systemem. Duże serwisy wcale nie są jedynymi którymi intersują się intruzi. Wielu z nich chce po prostu złamać jak najwięcej stron. Oczywiście, że duże serwisy są bardziej narażone na ataki ze strony "profesjonalnych" włamywaczy, którzy czynią to niejednokrotnie dla podniesienia swojego wizerunku w swoim światku.

Atakujący mają zwykle dużo czasu i zamiast zgadywać jak system został zabezpieczony, po prostu próbują wszystkich możliwości.

1.3.1 Hosty

Jest to obszar nad który administratorzy zazwyczaj poświęcają najwięcej uwagi. Wybieranie dobrych haseł, zabezpieczanie lokalnych usług sieciowych, aktualizowanie programów które są podatne na ataki to rzeczy które leżą w obowiązku lokalnego administratora.

1.3.2 Sieć lokalna

Jeśli w tej samej sieci znajdują się dziesiątki czy setki komputerów nie można zakładać, że każdy z nich jest zabezpieczony. Zapewnienie tego, że tylko autoryzowani użytkownicy mogą używać sieci, budowanie firewalli, używanie dobrych algorytmów szyfrowania i doprowadzenie do tego, że w sieci nie ma niebezpiecznych maszyn należą do administratora sieci.

1.3.3 Zabezpieczanie przez zmienianie

Ten rodzaj zabezpieczenia polega na przenoszeniu usług, które mają znane dziury na niestandardowe porty wierząc, że atakujący nie zauważą tego i nie będą próbować włamania. Ten rodzaj zabezpieczenia nie jest żadnym zabezpieczeniem.

1.3.4 Rzeczy o których należy pamiętać

Należy:

- Znać swój system
- Sprawdzać logi systemowe (/var/log/*) i mieć oko na swój system
- Utrzymywać system jak najbardziej aktualnym, aktualizując używane programy do ich najnowszych wersji.

2 Bezpieczeństwo fizyczne

Pierwszą warstwą zabezpieczeń, którą należy wziąć pod uwagę jest bezpieczeństwo fizyczne naszych komputerów. Należy zapytać się kto ma mieć fizyczny dostęp do naszej maszyny? Czy powinni mieć?

Bezpieczeństwo fizyczne naszego systemu w dużej mierze zależy od sytuacji oraz od budżetu. Użytkownik domowy nie potrzebuje wiele (choć może chcieć chronić maszynę przed dostępem np. dzieci). W laboratorium zabezpieczenia powinny być lepsze,ale użytkownicy powinni mieć dostęp do komputera. W biurze można zabezpieczać systam na czas naszej nieobecności. W niektórych firmach pozostawienie niezabezpieczonej konsoli może okazać się bardzo niebezpieczne.

Oczywistymi sposobami zabezpieczeń są zamki w drzwiach, na kablach, zamykane szafki, monitoring.

Sposobami które można wykorzystać są też :

- blokowanie komputera - zamki w obudowie, zabezpieczenia przed zmianą klawiatury lub myszki (są to możliwości niektórych płyt głównych)
- zabezpieczenia BIOS-u (hasła)
- zabezpieczenia boot loadera - np. restricted oraz password w LILO. Hasła te pozwolą jedynie na spowolnienie działania włamywacza, ale nie powstrzymają go całkowicie.
- xlock i vlock - blokowanie dostępu do konsoli lub X-ów. xlock nie zapewnia jednak ochrony przed przełączaniem się z X-ów na konsolę.

Pierwszą rzeczą, którą zawsze należy sprawdzać jest to, kiedy maszyna była wyłączana. Ponieważ Linuks jest stabilnym systemem jedynymi reboot'ami powinny być te wykonane przez administratora dla zamiany sprzętu, upgrade'u systemu itp. Jeśli maszyna została zrestartowana bez naszej wiedzy może być to znak włamania.

Ponieważ włamywacze często potrafią wyczyścić ślady ich obecności z logów dobrym pomysłem jest składowanie logów w dobrze chronionej sieci lub po prostu bezpośrednie wysyłanie ich na drukarkę. Potrafi to zrobić demon syslog. Niektóre z demonów logowania potrafią szyfrować swoje dane.

Rzeczy które należy sprawdzać w logach :

- krótkie lub niekompletne logi
- logi zawierające dziwne znaki czasowe (Timestamps)
- logi z niewłaściwymi prawami dostępu lub własności
- zapisy resetowania i restartowania systemu
- brakujące logi
- logowania poprzez *su* lub z dziwnych miejsc

3 Bezpieczeństwo lokalne

3.1 Tworzenie nowych kont

Należy upewnić się, że konta użytkowników mają minimalne wymagania potrzebne do zadań, które użytkownicy wykonują. Kilka zasad użytecznych przy dawaniu innym dostępu do maszyny :

- Daj im minimalną ilość praw jaką potrzebują
- Miej świadomość tego kiedy i skąd się oni logują, lub powinni logować.
- Upewnij się, że w systemie nie ma martwych (nieaktywnych) kont. Konta nieużywane stają się idealnym źródłem ataku na system.
- Używanie tego samego *userid* dla użytkownika na wszystkich komputerach i w sieciach do których ma on dostęp, pozwala na łatwe zarządzanie kontem i ułatwia analizę logów.
- Tworzenie kont o *id* takim samym jak *id* jednej z istniejących grup jest absolutnie niedopuszczalne. W tym układzie nie ma możliwości wyegzekwowania odpowiedzialności takiego użytkownika.

3.2 Bezpieczeństwo root'a

Root jak wiadomo ma dostęp do całej maszyny, a czasem nawet do całej sieci. Dlatego należy pamiętać, aby używać tego konta jak najkrócej, do specyficznych zadań. Normalnie superużytkownik powinien działać na zwykłym koncie, gdyż nawet mały błąd w czasie pracy jako root może spowodować problemy. Im krócej pracuje się z uprawnieniami roota, tym się jest bezpieczniejszym.

Kilka sposobów zabezpieczenia się przed szkodami :

- ustawienie użytkownikom aliasu do komendy *rm* tak, aby zawsze pytała o potwierdzenie kasowania (chociaż aliasy można zmieniać).
- logowanie się na konto roota tylko przy specjalnych zadaniach. To co można wykonać jako użytkownik nie powinno być wykonywane jako root.
- ścieżka (PATH) dla roota jest bardzo ważna. Należy ograniczyć ścieżkę jak tylko można i nigdy nie umieszczać w niej katalogu bieżącego ".". Poza tym w ścieżce nie powinno być katalogów z prawami zapisu gdyż to pozwoli atakującym zmodyfikować lub podmienić binaria, pozwalając im tym samym na dostanie uprawnień roota kiedy tylko uruchomią ten program.
- nie należy używać r-narzędzi (rlogin/rsh/rexec) jako root. Używają one nieszyfrowanej transmisji i są łatwym celem ataku. Nigdy nie należy tworzyć pliku *.rhosts* jako root.
- Plik /etc/securetty zawiera listę termniali z których może się logować root. Dla bezpieczeństwa dobrze jest zostawić ten plik pustym. Znaczyć to będzie, że chcąc zalogować się na roota trzeba będzie zalogować się najpierw jako zwykły użytkownik i wydać komendę *su*.

Jeśli chcmy pozwolić na dostęp superużytkownika do maszyny innej osobie (najlepiej bardzo zaufanej) można użyć np. programu *sudo*, która pozwala na użytkownikm na użycie ich hasła dla uzyskania ograniczonego dostępu do zbioru komend jako root. Można to wykorzystać, aby pozwolić użytkownikowi montować dyski wymienialne w linuksie, bez innych praw roota. *Sudo* trzyma logi wszystkich udanych i nieudanych prób użycia programu.

Jednakże z używaniem *sudo* wiąże się kilka zagrożeń, których trzeba być świadomym. Minaowicie, każdy program oferujący shella da użytkownikowi prawa roota. Dotyczy to np. wszystkich edytorów. Także program taki jak */bin/cat* pozwoli użytkownikowi na zmienianie i nadpisywanie plików co może prowadzić do powstawania exploitów.

4 Bezpieczeństwo plików i systemów plików

Kilka minut przygotowań i planowania przed wystawieniem systemu online pomoże ochronić system oraz dane w nim składowane.

- Nigdy nie powinno się pozwalać na uruchamianie programów *SUID/SGID* z katalogów domowych. Należy używać opcji *nosuid* w */etc/fstab* dla partycji zapisywalnych przez kogokolwiek innego niż root. Można także użyć opcji *nodev i noexec* na partycji */home i /var* tym samym nie pozwalając na wykonywanie jakichkolwiek programów czy też tworzenia jakichkolwiek plików urządzeń. (według guru linuksowych na pl.comp.os.linux jest to mimo wszystko do obejścia)

- Jeżeli eksportujemy lub montujemy systemów plików typu NFS należy skonfigurować plik */etc/exports* w najbardziej możliwie restrykcyjny sposób. Oznacza to dla eksportowania : nie używanie wzorców, nie pozwalanie na zapis rootowi, oraz eksportowanie read-only gdzie tylko jest to możliwe, natomiast dla montowania : używanie opcji *nodev, nosuid,* oraz jeśli jest to pożądane *noexec*

- Należy skonfigurować maskę tworzenia plików (poprzez umask) dla użytkowników jak najbardziej restrykcyjnie.

- Ustawienie limitów na system plików jest wskazane. Można to uczynić (nawet dla poszczególnych użytkowników) używając modułu PAM-a *pam_limits.so* i pliku */etc/pam.d/limits.conf*

- pliki */var/tmp/wtmp i /var/run/utmp* zawierają zapisy logowania wszystkich użytkowników systemu. Ich integralność musi być utrzymana ponieważ mogą one wskazać skąd i kiedy użytkownik (lub potencjalny włamywacz) przeniknął do systemu. Pliki te powinny mieć prawa dostępu 644, bez wpływu na normalne funkcjonowanie systemu.

- bitu *immutable* można użyć aby zabezpieczyć przed skasowaniem lub zmianą ważnych plików systemowych. Zabezpiecza to również przed utworzeniem symbolicznego linku do pliku (takie linki były źródłem ataku kasującego pliki */etc/passwd* ;ub */etc/shadow*. (man chattr)

- pliki z ustawionymi bitami SUID lub SGID stanowią potencjalne ryzyko i powinny być dokładnie monitorowane. Ponieważ programy te dają specjalne prawa użytkownikowi je wykonującemu, konieczne jest sprawdzenie, że programy niebezpieczne nie

są zainstalowane. Ulubionym trikiem crackerów jest exploitowanie programów typu SUID-root, następnie pozostawienie ich jako tylne wejście do systemu (backdoor), żeby wejść do systemu następnym razem nawet jeśli oryginalna dziura została zamknięta.

Znalezienie wszystkich programów SUID/SGID i śledzenie jakie one są, pozwala na zauważenie każdej zmiany, która może wskazywać na potencjalnego intruza. Używając poniższej komendy można znaleźć wszystkie programy SUID/SGID w systemie:

```
root#  find / -type f \( -perm -04000 -o -perm -02000 \)
```

Można usunąć prawa SUID i SGID przy użyciu *chmod,* i później je przywrócić jeśli okażą się konieczne.

- pliki typu world-writable zwłaszcza pliki systemowe mogą być dziurą w systemie jeśli włamywacz dostanie się do naszego komputera i je pozmienia. Ponadto katalogi z prawami zapisu dla wszystkich są niebezpieczne, gdyż intruz może dodawać lub kasować pliki kiedy tylko chce.
- pliki które nie należą do żadnego użytkownika ani grupy mogą wskazywać na nieupoważniony dostęp. Można je znaleźć przy pomocy komendy:

```
root# find / -nouser -o -nogroup -print
```

- Pliki *.rhosts* nie powinny być dozwolone. Należy pamiętać że włamywacz potrzebuje tylko jednego niezabezpieczonego konta, żeby uzyskać dostęp do całej sieci.
- przed zmianą praw dostępu na jakimkolwiek systemie plików należy dobrze się zastanowić co się robi. Nigdy nie należy zmieniać praw ponieważ wydaje się to łatwym sposobem na uruchomienie czegoś. Zawsze należy się zastanowić dlaczego plik ma takie, a nie inne prawa dostępu przed ich zmienieniem.
- dodatkowym zabezpieczeniem może być np. szyfrowanie systemu plików.

4.1 Prawa dostępu

Ważne jest zapewnienie tego, że pliki systemowe nie zostaną wyedytowane przez użytkowników i grupy, które nie powinny mieć tego typu możliwości.

Systemy unixowe rozdzielają kontrolę dostępu do plików i katalogów zgodnie z układem: właściciel, grupa, inni, Zawsze istnieje dokładnie jeden właściciel, dowolna ilość członków grupy, oraz każda inna osoba. Prawa dostępu dla katalogu mogą mieć inne znaczenie niż takie same prawa dla pliku.

- Odczyt:
  - Możliwość wyświetlenia zawartości pliku
  - Możliwość odczytu katalogu
- Zapis:
  - Możliwość dodawania do lub zmieniania pliku

  - Możliwość kasowania lub przenoszenia plików w katalogu

- Wykonanie:
  - Możliwość uruchomienia programu binarnego lub skryptu
  - Możliwość szukania w katalogu połączona z prawami odczytu

- Save Text Attribute (dla katalogów) - "sticky bit" także ma inne znaczenie dla katalogów niż dla plików. Jeśli "sticky bit" jest ustawiony dla katalogu wtedy użytkownik może tylko kasować z tego katalogu pliki które należą do niego lub dla których ma explicite określone prawo zapisu, nawet jeśli ma prawa zapisu dla bieżącego katalogu. Zostało to zaprojektowane dla katalogów takich jak */tmp*, które są typu world-writable, ale w którym nie należy pozwalać na kasowanie dowolnych plików przez kogokolwiek. "Sticky bit" jest widoczny jako "t" przy listowaniu katalogu.

- Atrybut SUID (dla plików) - Opisuje on prawa set-user-id dla pliku. Kiedy ten tryb dostępu dla pliku jest ustawiony oraz plik jest plikiem wykonywalnym, proces który zostanie uruchomiony otrzyma prawa dostępu do systemu takie jak użytkownik, który jest właścicielem pliku, a nie takie jak ten, który go uruchomił. Dla przykładu, jeżeli właścicielem pliku jest root i ustawiony jest bit SUID wtedy program ten jest wykonywany z prawami superużytkownika.

- Atrybut SGID (dla plików) - Jeśli jest ustawiony dla praw dostępu dla grupy, bit ten kontroluje status "set-group-id" dla pliku. Proces uruchomiony przez taki plik zachowuje się tak samo jak plik z SUID-em, tyle że zamiast praw użytkownika, program otrzymuje prawa grupy.

- Atrybut SGID (dla katalogów) - Jeśli bit SGID zostanie ustawiony dla katalogu (chmod g+s katalog) pliki stworzone w tym katalogu będą miały grupę ustawioną na grupę do jakiej należy katalog.

Poniższe linie są przykładowymi określeniami minimalnego zbioru praw dostępu dla pliku, które są wymagane, aby wykonać akcję opisaną obok:

```
-r--------  pozwala na odczyt pliku przez właściciela
--w-------  pozwala użytkownikowi zmieniać lub kasować plik
            (Należy zauważyć, że ktokolwiek z prawami zapisu w
             katalogu w którym plik się znajduje może plik nadpisać i
             tym samym zmazać go)
---x------  właściciel może uruchomić ten program, ale nie może on być
            skryptem shellowym, ponieważ potrzebuje on praw odczytu
            pliku
---s------  program zostanie uruchomiony z effective User ID (EUID)
            ustawionym na właściciela pliku.
--------s-  program zostanie uruchomiony z effective Group ID (EGID)
            ustawionym na właściciela pliku.
-rw------T  nie będzie wykonywane uaktualnianie parametru ‘‘last
            modified time’’. Zwykle używane dla plików swapu
---t------  bez efektu
```

Natomiast te linie są przykładowymi określeniami minimalnego zbioru praw dostępu dla katalogu, które są wymagane, aby wykonać akcję opisaną obok:

```
dr--------  zawartość katalogu może być wyświetlona, ale atrybuty
            plików nie mogą zostać odczytane.
d--x------  można wejść do katalogu i używać go przy podaniu pełnej
            ścieżki wykonania
dr-x------  atrybuty plików mogą być odczytane przez użytkownika
d-wx------  pliki mogą być tworzone lub kasowane, nawet jeśli katalog
            nie jest katalogiem bieżącym
d------x-t  zabezpiecza pliki przed skasowaniem przez innych z prawami
            zapisu. Używane dla {\em /tmp}
d---s--s--  bez efektu
```

Pliki konfiguracyjne systemu zwykle w */etc* zwykle mają ustawiony tryb 640 (-rw-r—–) a ich właścicielem jest root. W zależności od wymagań bezpieczeństwa można te prawa zmieniać. Nigdy nie należy zostawiać plików systemowych zapisywalnych dla grupy lub każdego. Pliki takie jak */etc/shadow* powinny być tylko do odczytu dla roota, a katalogi w */etc* nie powinny być przynajmniej dostępne dla każdego.

Skrypty z bitem SUID

Skrypty z ustawionym bitem SUID są poważnym ryzykiem i dlatego kernel nie pozwala na ich uruchamianie. Nieważne jak bardzo bezpieczny wydawałby się skrypt, może on zostać wykorzystany przez crackera do uzyskania shella roota.

4.2 Integralność danych i sieci

4.2.1 Sprawdzanie integralności

Sprawdzanie integralności systemu plików to sprawdzanie, czy pliki nie zostały zmienione wbrew naszej woli. Przy czym nie chodzi tu o pliki uszkodzone, ale o pliki zainfekowane trojanami, albo backdoory. Istnieją wyrafinowane *rootkity*, które po uruchomieniu dają użytkownikowi prawa roota. Dodatkowo takie pakiety mogą posiadać, zmodyfikowane wersje binariów, które utrudniają wykrycie użytkownika z prawami roota. Np. istnieją specjalne wersje *ls, ps, netstat,* które ukrywają procesy atakującego, nie pokazując plików przy listingach katalogów, nie wyświetlając aktywnych połączeń sieciowych, itd. Doskonałym narzędziem do sprawdzania integralności, współpracującym z firewallami oraz innymi liniami obrony jest Tripwire (którego można znaleźć na stronie http://www.tripwiresecurity.com Inne programy tego typu to np. Aide lub Osiris. W tym miejscu opierać będziemy się na Tripwire. To co robi taki program to pobranie matematycznego odcisku palca (sumy kontrolnej) każdego pliku, który jest przechowywany w systemie, oraz własności plików określone przez administratora i zapisuje je w bazie danych. W każdym momencie administrator może uruchomić Tripwire, który porównuje sumy kontrolne liczone na bieżąco oraz własności plików z tymi które znajdują się w bazie danych. Jeśli sumy się zgadzają wszystko jest w porządku, natomiast jeśli nie oznacza to, że plik się zmienił. Może to wskazywać na włamanie do systemu i podmienienie pliku z programem.

Przykładowy raport z działalności Tripwire może wyglądać tak:

```
% /usr/TSS/bin/tripwire --check
Added Objects:
Rule Name: /etc (/etc)
Total number of added objects: 1
Added object name: /etc/passwd+
Property: Expected: Observed:
------------ ----------- -----------
* Device Number    ---    0
* Inode Number     ---    626645
* Mode             ---    -rw-r--r--
* Num Links        ---    1
* UID              ---    root (0)
* GID              ---    other (1)
* Size             ---    0
* Modify Time      ---    Thu Dec 10 16:53:42 1998
* Blocks           ---    0
* Object Type      ---    Regular File
* CRC32            ---    AAAAAA
* MD5              ---    DUHYzZjwCyBOmACZjs+EJ+
```

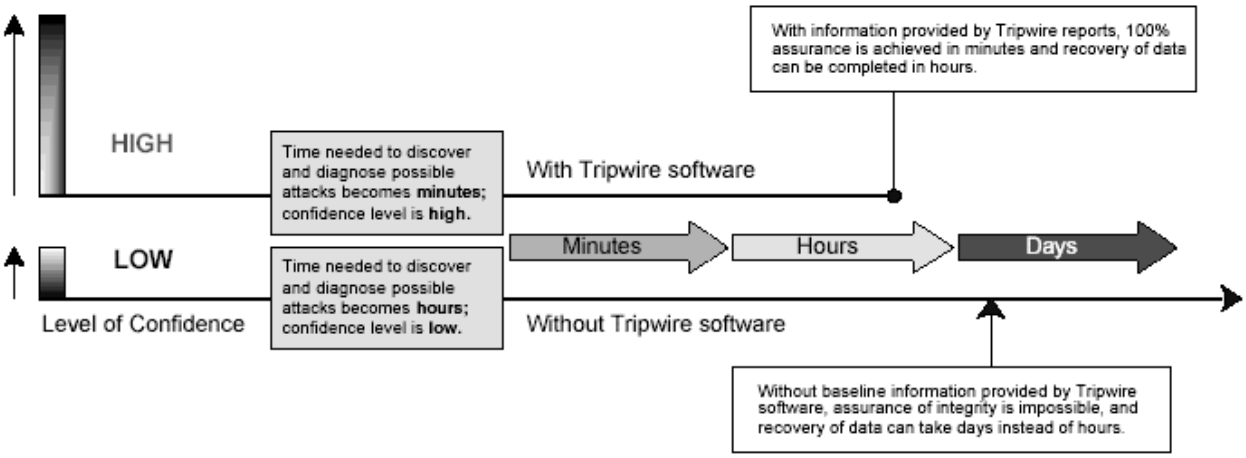
Widać tu, że zmieniony został plik */etc/passwd*,który w tym momencie jest po prostu pusty.

Dobrym pomysłem jest zainstalowanie binariów Tripwire na nośniku, który umożliwia fizyczne ustawienie ochrony przed zapisem. W ten sposób intruzi nie będą mogli zmienić samego programu sprawdzającego lub bazy danych. Można ustawić wywoływanie programu sprawdzającego w crontabie:

```
# set mailto
MAILTO=kevin
# run Tripwire
15 05 * * * root /usr/local/adm/tcheck/tripwire
```

Spowoduje to wysłanie maila z informacją codziennie rano o godzinie 5.15

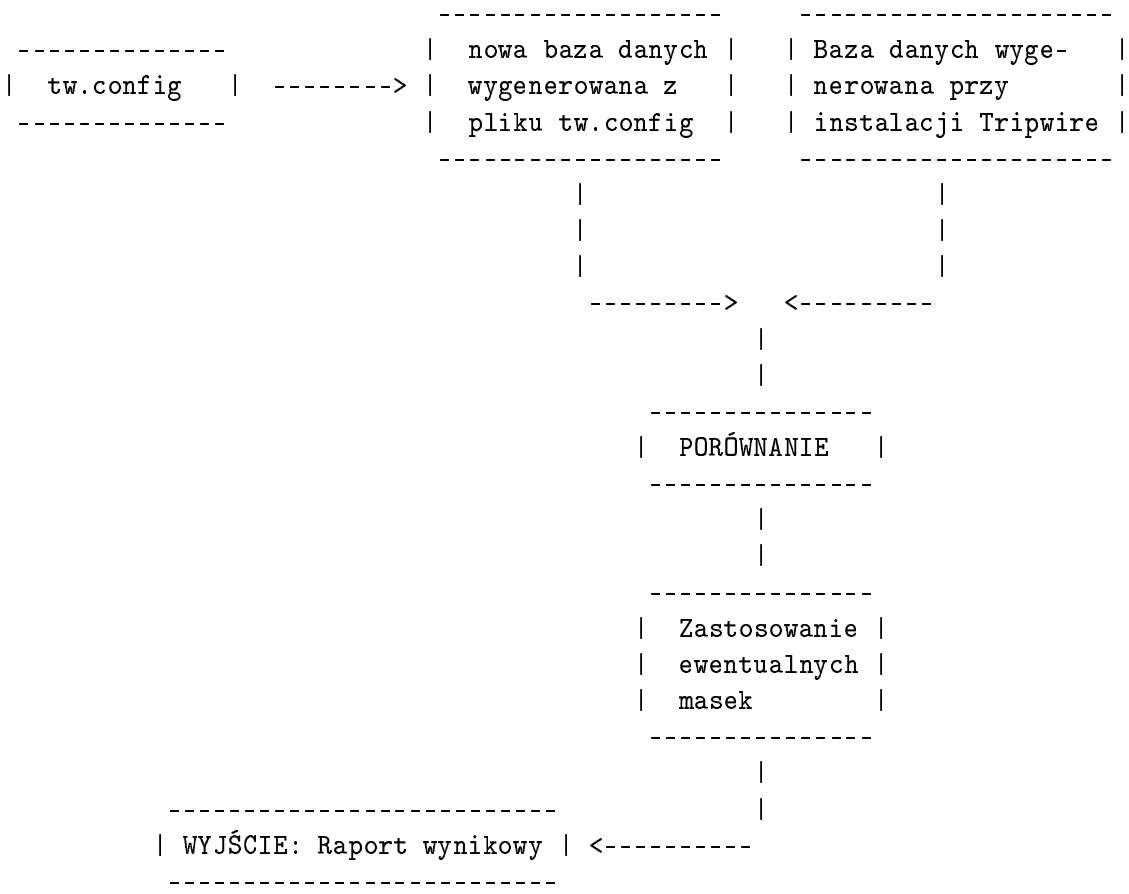
Programy sprawdzające mogą być bardzo przydatne do wykrywania włamywaczy zanim wykryjemy ich w inny sposób. Może on znacząco zaoszczędzić czas potrzebny na określenie strat i odtworzenie stanu systemu sprzed momentu włamania.



Rysunek 1: Wykrywanie włamań z użyciem Tripwire

Ponieważ wiele plików zmienia się w przeciętnym systemie należy uważać co jest działalnością crackera, a co naszą własną.

Oto schemat działania Tripwire:



Kiedy TripWire jest odpalany używane są tylko dwa pliki: tw.config, gdzie wpisane są pliki które mają być sprawdzone, oraz baza danych wygenerowana przy instalacji programu,

która posiada oryginalne odciski palców. Plik tw.config składa się z wpisów podobnych do tego:

```
/bin/mount +p+i+u+g+s+a
```

Ten wpis oznacza, że Tripwire ma sprawdzić bity praw, numer inode'a, ilość linków, ID użytkownika i grupy właściciela, oraz znacznik ostatniego dostępu. Są to flagi, które są sprawdzane w etapie aplikowania masek. Oczywiście istnieje dużo więcej flag, są one opisane w dokumentacji przychodzącej z Tripwire. Po zmienieniu lub dodaniu plików które chcemy sprawdzać można wykonać uaktualnienie bazy danych o nowe odciski.

Wygenerowany raport opisuje dokładnie każde złamanie polityki zabezpieczeń w szczegółach w zależnych od tego, czy dany plik w systemie został dodany, skasowany czy zmieniony. Każdy raport wyświetla własności, które obiekt obecnie posiada w systemie oraz, jeśli jest właściwa, starą wartość z bazy danych. Jeśli występują jakiekolwiek różnice pomiędzy bazą danych i obecnym systemem, administrator może albo naprawić problem, podmieniając plik na poprawny (np. włamywacz podmienił plik */bin/login*) albo zaktualizować bazę danych, żeby umieścić w niej informacje o nowym pliku.

4.2.2 Zalety integralności

Środowisko bez integralności doświadczy	Środowisko z integralnością doświadczy
Długich cyklów diagnozowania problemów	Szybkiego wykrywania problemów z dokładnym wskazaniem miejsca kłopotów
Nieudokumentowanych zmian w krytycznych plikach	Niższych kosztów Total Cost of Ownership (TCO) dzięki spójności danych
Trudności w sprawdzeniu, które pliki zostały zmienione po włamaniu	Szybkich oszacowań strat i oczyszczenia systemu po ataku
Niemożności śledzenia krytycznych danych związanych ze stabilnością infrastruktury	Stabilności infrastruktury dzięki mierzeniu integralności
Opóźnienia w uaktualnianiu lub wymienianiu komponentów infrastruktury	Szybszego wprowadzania nowych komponentów IT
Utraty kontroli nad elementami infrastruktury IT	Poprawionego bezpieczeństwa infrastruktury
Zwiększonych kosztów utrzymania akceptowalnego poziomu ryzyka	Obniżonych kosztów przez aktywne zarządzanie ryzykiem
Ciągłych cyklów włamanie-naprawa i niskiej produktywności pracowników IT	Mniej "gaszenia pożarów", a więcej czynności biznesowych
Nieprzewidywalnych scenariuszów na wypadek awarii/włamania	Niższych kosztów dzięki możliwości ograniczenia strat

Rysunek 2: Zalety integralności według IDC, 2001

4.3 Kopie zapasowe

```
Sprzęt jest nieokreślenie pewny.
Oprogramowanie jest określenie zawodne.
Ludzie są nieokreślenie zawodni.
Natura jest określenie pewna.
```

4.4 Potrzeba tworzenia kopii zapasowych

Twoje dane są cenne. Odtwarzanie ich wymaga czasu i wysiłku, a to kosztuje zdrowie i pieniądze; niektórych danych nie da się odtworzyć. Dlatego powinno się chronić swoje dane.

Zasadniczo istnieją cztery powody, z których możemy stracić dane: usterka sprzętu, pluskwa w oprogramowaniu, człowiek, naturalne katastrofy. Pomimo tego, że sprzęt staje się coraz bardziej niezawodny, zdarzają się usterki (czasami bardzo niespodziewanie). Najbardziej krytycznym medium przechowującym dane jest dysk twardy, jego zasada działania opiera się na malutkich polach magnetycznych stykających się z szumem magnetycznym świata. Nowoczesne oprogramowanie nie stara się być niezawodne; twardy jak skała program jest wyjątkiem, nie zasadą. Ludzie są zawodni, popełniają błędy, lub umyślnie niszczą dane. Natura płata figle. Wszystko wszystkim ale małym cudem jest działanie wszystkiego.

Kopie zapasowe są sposobem chronienia naszych danych. Posiadanie kilku kopii danych zapobiega przed ich utratą, do tego jest znacznie tańsze niż ich odtwarzanie.

Kopie zapasowe muszą być tworzone właściwie. Jak wszystko co fizyczne kopie z czasem zepsują się. Częścią tworzenia kopii zapasowej jest upewnienie się czy działa; nie chciałbyś się o tym dowiedzieć w razie potrzeby. Pamiętajmy, że w czasie tworzenia kopii również coś się może zdarzyć; wtedy zostaniemy z ręką w nocniku. Może się zdarzyć, że odtwarzając kopię zauważymy, że coś pominąłęliśmy, np. bazę użytkowników zawierającą 15000 wpisów. Oczywiście, może się zdarzyć, że wszystko działa wspaniale, jednak naszemu streamerowi coś się przydarzyło.

Paranoja pomaga przy tworzeniu kopii zapasowych.

4.5 Wybór medium na kopię zapasową

Najważniejszą decyzją jest wybór medium. Musimy rozważyć koszty, pewność, prędkość, dostępność i użyteczność.

Koszty są ważne, ponieważ pojemność wszystkich nośników, przechowujących kopie zapasowe, powinna być kilkakrotnie większa niż ilość danych. W większości wypadków wybiera się najtańsze medium.

Pewność działania jest bardzo ważna, zepsuta kopia zapasowa może doprowadzić dorosłego człowieka do płaczu. Medium musi być w stanie przechować dane przez lata. Sposób użycia jest również ważny i może znacząco wpłynąć na żywotność kopii. Dysk twardy pracuje pewnie, jednak jako medium do tworzenia kopii zapasowych nie jest zbyt przydatny, zwłaszcza gdy znajduje się w tym samym komputerze co zachowywane dane.

Prędkość zazwyczaj nie jest zbyt ważna, jeżeli można tworzyć kopie bez interakcji z użytkownikiem. Nie ma znaczenia ile tworzenie kopii zajmuje godzin dopóty, dopóki nie

trzeba przy tym siedzieć. Z drugiej strony jeżeli kopii zapasowej nie można wykonać w czasie bezczynności komputera prędkość zaczyna się liczyć.

Dostępność jest oczywiście niezbędna, nie można utworzyć kopii zapasowej jeżeli medium nie istnieje. Mniej oczywista jest potrzeba istnienia medium w przyszłości i to nie tylko na naszym komputerze. W innym przypadku nie będziemy mógli odtworzyć danych.

Użyteczność jest ważnym czynnikiem określającym częstotliwość wykonywania kopii. Czym łatwiej tworzyć kopie zapasowe tym lepiej. Proces ich tworzenia nie może być trudny ani nudny.

Aktualnie do wyboru mamy taśmy, płyty CD (lub pochodne). CD są dobrym rozwiązaniem gdy dane trzeba zarchiwizować, natomiast w świecie kopii zapasowych nadal królują streamery.

4.6 Wybór programu

Istnieje wiele programów do tworzenia kopii zapasowych. Tradycyjnymi programami do tego przeznaczonymi są: tar, cpio, dump. Wybór narzędzia może być podyktowany wybranym medium.

tar i cpio są podobne. Oba programy potrafią zapisywać i odczytywać dane z taśm, do tego potrafią obsłużyć każde medium, z którym potrafi sobie poradzić jądro. Niektóre Unixowe wersje tara i cpio mogą mieć problemy z dziwnymi plikami (dowiązaniami symbolicznymi, plikami urządzeń, itp.), jednak Linuxowe wersje poradzą sobie z każdym plikiem.

dump pracuje w sposób nieco odmienny niż powyższe programy - odczytuje pliki bezpośrednio z urządzenia, bez pośrednictwa systemu plików. Program został specjalnie napisany do tworzenia kopii zapasowych, tar i cpio powstały w celu archiwizowania danych.

Bezpośrednie odczytywanie systemu plików ma pewne zalety - umożliwia odczytywanie plików bez zmiany ich znaczników czasowych; używając tara, lub cpio należałoby zamontować system w trybie wyłącznie do odczytu. Bezpośrednie odczytywanie systemu plików jest wydajniejsze przy kopiowaniu wszystkiego. Główną wadą takiego podejścia jest ograniczenie programu do jednego systemu plików; Linuxowy dump rozumie tylko i wyłącznie ext2.

dump ma wbudowaną obsługę poziomów kopiowania (omówimy je w dalszej części); tar i cpio wymagają do tego dodatkowych narzędzi.

Ostatnio dużo szumu jest wokół graficznego programu do robienia backupów o nazwie amanda.

4.7 Proste kopie zapasowe

Tworzenie takich kopii sprowadza się do skopiowania wszystkiego za pierwszym razem, oraz późniejszych dopisywań. Pierwsza kopia nazywana jest pełną, późniejsze to kopie różnicowe. Odtwarzanie danych za pomocą kopii różnicowych wymaga odtworzenia kopii pełnej.

Jeżeli chcemy tworzyć kopie codziennie i posiadasz 6 taśm, możemy 1 wykorzystać na pełną kopię. Na taśmach 2 - 5 tworzyć kopie inkrementacyjne, po czym tworzymy pełną kopię na taśmie 6 i znów wykorzystujemy taśmy 2 - 5. Nie powinno się nadpisywać taśmy 1 do czasu stworzenia pełnej kopii (na wszelki wypadek). Po utworzeniu pełnej kopii na

taśmie 6 i przystępując do tworzenia następnej pełnej kopii wykorzystujemy taśmę nr 1, a 6 odkładasz na półkę, itd.

Jeżeli posiadasz więcej niż 6 taśm możemy dodatkowe wykorzystać do tworzenia pełnych kopii. Za każdym razem wykorzystujemy najstarszą z nich podczas tworzenia następnej pełnej kopii. Posiadając kilka pełnych kopii możemy znaleźć plik, który aktualnie nie istnieje, lub jego starszą wersję, itp.

4.8 Kopie wielopoziomowe

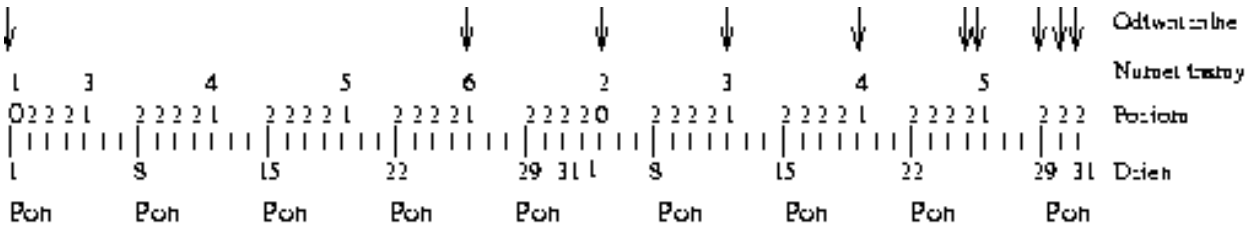
Prosta metoda tworzenia kopii zapasowych jest dobra dla osobistych komputerów i małych serwerów. Przy dużej ilości danych należy zastosować inne metody.

Prosta metoda ma dwa poziomy: pełny i różnicowa kopia. Można to uogólnić do kilku poziomów. Pełna kopia to poziom 0, kopie różnicowe to: 1, 2, 3, ... Na poszczególnym poziomie zachowujesz dane, które są zmienione względem niższego, lub równorzędnego poziomu.

W ten sposób uzyskujemy dłuższą żywotność danej kopii danych, umożliwiając w ten sposób dostęp do kilku wersji plików, lub danych, które zostały usunięte dawno temu.

Dla przykładu: posiadając 10 taśm, możemy 1 i 2 wykorzystywać na miesięczne kopie (robione w pierwszy piątek miesiąca), taśmy 3 - 6 będą zawierać tygodniowe kopie (inne piątki, pamiętaj, że w miesiącu może być 5 piątków, więc potrzebujemy 4 taśmy), taśmy 7 - 10 będą kopiami dziennymi (poniedziałek do czwartku). Za pomocą dodatkowych 4 taśm wydłużyliśmy sesję z 2 tygodni do dwóch miesięcy. Prawdą jest, iż nie możemy odzyskać wszystkich wersji wszystkich plików, jednak to co możemy odzyskać jest często wystarczające.

Rysunek 3 pokazuje, który poziom powinien być użyty w danym dniu, oraz z których kopii można odzyskać dane pod koniec miesiąca.



Rysunek 3: Przykładowy wielopoziomowy schemat tworzenia kopii zapasowych

Poziomy kopii mogą być również używane do utrzymywania czasu odzyskiwania danych do minimum. Jeżeli posiadamy wiele kopii różnicowych o coraz rosnącym jednostajnie poziomie, aby odzyskać wszystkie dane musimy je wszystkie odczytać. Możemy jednak tworzyć kopie o niekolejnych poziomach ograniczając liczbę taśm niezbędnych do odtworzenia systemu.

Aby uzyskać taki efekt, możemy używać niższego poziomu dla każdej następnej kopii różnicowej. Jednakże, w ten sposób wydłużamy również czas tworzenia kopii (każda kopia zawiera wszystko od poprzedniej pełnej kopii). Lepszy schemat jest sugerowany przez podręcznik programu dump. W skrócie wykonuj kolejno kopie o poziomach: 3, 2, 5, 4, 7, 6, 9, 8, 9, itd. W ten sposób tworzenie kopii i odzyskiwanie danych trwa krótko. Liczba niezbędnych taśm zależy od tego jak długie przerwy będziemy czynić między pełnymi kopiami, jednak i tak będzie niższa niż w przypadku prostego schematu.

Taśma	Poziom	Kopiowanie (dni)	Ilość taśm niezbędnych do odtworzenia
1	0	n/a	1
2	3	1	1, 2
3	2	2	1, 3
4	5	1	1, 2, 4
5	4	2	1, 2, 5
6	7	1	1, 2, 5, 6
7	6	2	1, 2, 5, 7
8	9	1	1, 2, 5, 7, 8
9	8	2	1, 2, 5, 7, 9
10	9	1	1, 2, 5, 7, 9, 10
11	9	1	1, 2, 5, 7, 9, 10, 11
...	9	1	1, 2, 5, 7, 9, 10, 11, ...

Tabela 1. Efektywny schemat tworzenia wielopoziomowych kopii zapasowych.

Wymyślny schemat może zredukować liczbę pracy, jednak może zwiększyć ilość czynników, na które trzeba zwracać uwagę. Musimy zadecydować czy warto.

dump ma wbudowaną obsługę poziomów, dla tara i cpio musimy to zaimplementować za pomocą odpowiednich skryptów.

4.9 Co zachowywać?

Powinniśmy zachowywać najwięcej jak tylko można, głównym wyjątkiem są programy, które mogą być łatwo reinstalowane, choć i one posiadają pliki konfiguracyjne, które należy zachować. Innym wyjątkiem jest system plików /proc, który jest automatycznie tworzony przez jądro i nie powinniśmy go zachowywać.

Zachowywanie newsów, logów i innych rzeczy z /var zależy od nas - musimy zdecydować na czym nam zależy.

Oczywistymi rzeczami do zachowywania są pliki użytkowników (/home), oraz konfiguracja systemu (/etc i inne pliki rozsiane po systemie).

4.10 Skompresowane kopie zapasowe

Kopie zapasowe zajmują dużo miejsca, które przecież kosztuje. Aby zredukować objętość kopii można je kompresować. Istnieje kilka sposobów. Niektóre programy mają obsługę kompresji wbudowaną, np. opcja –gzip (-z) tara przepuszcza archiwum przez gzipa.

Niestety skompresowane archiwa mogą czasami przysporzyć kłopotów. Z powodu działania kompresji uszkodzenie jednego bitu może spowodować utratę wszystkich danych. Niektóre programy posiadają możliwość naprawy, jednak metody te są zawodne przy dużej ilości błędów. Kopie zapasowe muszą być godne zaufania, a taka metoda kompresji naraża twoje dane.

Alternatywnie, można kompresować każdy plik z osobna. W najgorszym wypadku stracimy jeden plik, co przy nieskompresowanej kopii bezpieczeństwa jest również możliwe - to samo bezpieczeństwo. Program afio (odmiana cpio) potrafi tworzyć takie kopie zapasowe.

Kompresja zajmuje czas, przez co może spowodować zatrzymanie taśmy. Można to ominąć poprzez buforowanie wyjścia (wewnętrznie, jeżeli program jest na tyle sprytny, lub

zewnętrznie - za pomocą innych programów). Może to stanowić duże utrudnienie na wolnych komputerach. Jeżeli dane nie są zapisywane w odpowiednim tempie taśma zostaje zatrzymana, czyniąc proces kopiowania jeszcze wolniejszym.

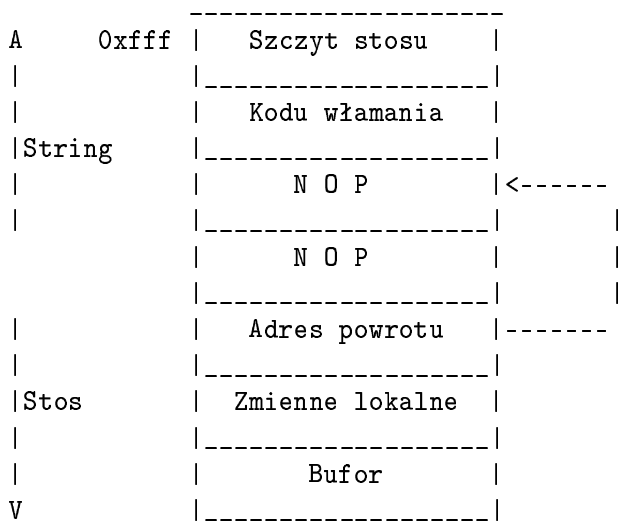
5 Przepełnienia bufora

Problem przepełnienia bufora został zauważony juz na początku lat 70. Jest to bardzo poważna konsekwencja, wprowadzonej w jezyku C, integralności danych programu. Samą sytuację rozwinęli jeszcze programiści. Jezyk C jako język dość niskiego poziomu (w porównaniu z TP, Ada, Modula) był wykorzystywany do pisania jak najbardziej efektywnego kodu. Niestety w takim kodzie nie ma miejsca na sprawdzanie poprawności danych. Po prostu tam gdzie tylko się da nie sprawdza się tego. A sam język, za programistę tego nie zrobi. Także kiedy zmienne są inicjalizowane, kopiowane, przesuwane nie są sprawdzane ich rozmiary co może spowodować przepełnienie bufora. Konsekwencją takiej sytuacji jest podatność dzisiejszych systemów typu UNIX na włamania. Pierwszym szeroko opisywanym włamaniem do systemu było wydarzenie z 1998 roku. Włamanie nastąpiło poprzez wykorzystanie funkcji gets(). Program rezerwował sobie 512 bajtów na dane zczytane. Jednocześnie wielkość wprowadzonych danych nie była sprawdzana. Jeżeli danych było więcej, były one nadpisywane w miejsce stosu, gdzie normalnie powinien się znajdować adres powrotu z funkcji. Tą sytuację wykorzystał włamywacz, żeby wykonać polecenie uruchomienia shella i później dostać prawa root'a.

5.1 Metodologia ataku

Ataki oparte na przepełnieniu bufora wykorzystują braki w sprawdzaniu granic, wielkości tablic jakie są wprowadzane do systemu przez użytkownika, a następnie przechowywne przez funkcję w buforze. Poprzez wpisywanie odpowiednich danych na koniec takiego bufora, użytkownik może zmienić stan programu. Wiele programów napisanych w C jest podatne na przepełnienie bufora. Z drugiej strony kultura programowania w języku C stawia przede wszystkim na wydajność. Jeżeli tylko nie trzeba to nie sprawdza się potencjalnych błędów. Wiele funkcji bibliotecznych jest włąsnie tak napisanych np.: gets(), strcpy() itd..

5.2 Przepełnienie bufora na stosie



Najpopularniejszą formą ataku na system komputerowy, wykorzystującą przepełnienie bufora jest atakowanie buforów na stosie. Ataki są tak przeprowadzane, żeby uzyskać dwa podstawowe cele:

- Wprowadzić swój kod Użytkownik wprowadza string zawierający wykonywalny kod (natywny dla danej maszyny). Jest to przeważnie zestaw instrukcji maszynowych wykonujących shella, co w efekcie daje powłokę z prawami root'a.
- Zmienić adres powrotu Dla aktualnie wykonywanej funkcji system ma przeznaczony pewien obszar stosu -> ramkę na stosie. Ten obszar jest nad buforem, który jest aktualnie atakowany. Przepełnienie bufora nadpisuje adres powrotu z aktualnej funckji w taki sposób, żeby został wykonany kod ataku. Kiedy funkcja się kończy sterowanie zamiast powrócić do miejsca, z którego funkcja została wywołana, przenosi się w miejsce gdzie znajduje się kod włamywacza. W wielu wypadkach taki kod jest poprzedzony dużą ilością NOP'ów. Redukuje to wymaganą dokładność z jaką trzeba odgadnąć adres miejsca, w którym znajduje się kod ataku.

Programy atakowane w ten sposób to przede wszystkim demony uruchamiane z prawami root'a. Kod wprowadzany przez hackera, daje mu shella z prawami root'a.

5.3 Przepełnienie bufora na stercie

Sterta to obszar pamięci zarezerwowny na zmienne alokowane dynamicznie przez aplikacje. W typowym ataku opartym na przepełnieniu bufora na stercie, zmienne takie jak hasła, nazwy plików, uid są nadpisywane. Taki sposób ataku nie jest równie popularny jak poprzedni, ale może prowadzić do otrzymania nieautoryzowanego dostępu z dużymi prawami. Można np. zmienić sobie uid na 0.

5.4 Różnice

W ataku na bufor umieszczony na stosie, kod maszynowy wprowadzony przez włamywacza jest wykonywany sekwencyjnie poprzez nadpisanie adresu powrotu. W atakach

opartych na przepełnieniu bufora na stercie, nadpisywane są dynamiczne zmienne programu (np. demona), aby w efekcie powiększyć swoje prawa w systemie.

6 SSH

Krótki wstęp - ssh czyli Secured Shell jest mechanizmem, który zapewnia bezpieczne połączenie terminalem przez potencjalnie niebezpieczną sieć . Za taką można przyjąć np. Internet.

6.1 Telnet i FTP

Zwykły telnet jest programem niebezpiecznym. Wszelakie dane (nazwa użytkownika, hasło, dane wstukiwane na klawiaturze) są przesyłane jawnie bez żadnego szyfrowania. Wystarczy zainstalować program monitorujący ruch w sieci, i przechwycić hasła oraz inne dane. Podobnie, niestety, rzecz ma się z FTP.

SSH działa na zasadzie szyfrowania danych kluczem publicznym. Metoda ta zakłada szyfrowanie danych przy pomocy jednego klucza i deszyfrowanie przy pomocy innego. Jedyną możliwością otrzymania oryginalnej wiadomości jest zastosowanie odpowiednich kluczy z tej samej pary. Autor szyfruje wiadomość za pomocą klucza publicznego adresata, ten zaś deszyfruje ją za pomocą swojego prywatnego klucza, który jest znany tylko jemu. Znając klucz publiczny nie można odtworzyć klucza prywatnego. (RSA - klucz publiczny to iloczyn dwóch dużych liczb pierwszych. Które z kolei tworzą klucz prywatny. Sam algorytm do znajdowania czynników pierwszych danej liczby, jest bardzo złożony)

6.2 Zabezpiecz się!

Jeżeli już obawiamy sie, że nasza sieć może ulec atakowi. To nie wystarczy zastąpić telenta SSH. Należy również wyłączyć wszystkie standardowe usługi UNIX'owe, które przesyłają hasła i inne dane bez szyfrowania ich. Do usług tych należą: rlogin, rsh, rcp. W sytuacji kiedy dostęp do serwera mają tylko użytkownicy z wewnętrznej sieci można pozostawić te usługi włączone, jednak gdy jesteśmy dostępni z zewnątrz należy je bezwzględnie wyłączyć. Konfiguracja tych usług odbywa się poprzez plik */etc/inetd.conf*.

6.3 Instalacja

Jeżeli w systemie nie ma zainstalowanego ssh należy je ściągnąć skompilować i skonfigurować. My skupimy się tutaj na samej konfiguracji. Mamy dwa standardy: ssh1 i ssh2. SSH2 jest nowsze, bezpieczniejsze ale niekompatybilne ze starszym bratem. Najlepiej mieć w systemie zainstalowane dwie wersje SSH. Pliki konfiguracyjne dla demonów tych usług to odpowiednio: */etc/sshd_config* i */etc/ssh2/sshd2_config*

6.4 Konfiguracja SSH1

Poniżej przedstawiamy przykład pliku konfiguracyjnego dla SSH1.

- **Port 22** domyślny port dla ssh to właśnie 22

- **ListenAddress 148.81.184.XXX** adres, gdzie jest zainstalowany serwer (demon) sshd obsługujący ssh1. Ustawienie jest istotne przy systemach o kilku kartach sieciowych (na przykład rutery lub bramki). Najwygodniej ustawić wtedy adres IP tej karty sieciowej, do której połączenia będą częstsze lub do karty od strony sieci zewnętrznej.

- **HostKey /etc/ssh_host_key**
  RandomSeed /etc/ssh_random_seed Położenie pliku z kluczem oraz parametrem losowym

- **KeyRegenerationInterval 3600** Co pewien czas następuje odnowienie klucza maszyny. Ten czas (w sekundach) ustawia się właśnie za pomocą tej opcji. Domyślnie - 1 godzina.

- **PermitRootLogin no** Czy root może się zalogować zdalnie za pomocą ssh. Patrz uwagi dalej.

- **IgnoreRhosts yes**
  IgnoreRootRhosts yes
  RhostsAuthentication no
  RhostsRSAAuthentication no
  Ignorowanie plików .rhosts, które wskazują źaufane"maszyny, skąd mógłby się zalogować użytkownik bez podawania hasła oraz zezwalanie na autentykację za pomocą mechanizmu rhosts.

- **PrintMotd yes** Czy wypisywać komunikat powitania (motd - najczęściej /etc/motd)

- **X11Forwarding yes**
  XAuthLocation /usr/bin/X11/xauth Czy przekazywać dane połączenia X11 (graficznego) i za pomocą którego programu dokonywać autentykacji użytkownika w środowisku graficznym X-Windows

- **(Allow,DenyUsers) I Groups** DenyGroups student Zezwolenie na logowanie się za pomocą ssh. Patrz dalej.

- **RSAAuthentication yes** Wybór metody autentykacji. Włączyć TYLKO RSA.

- **PasswordAuthentication yes**
  PermitEmptyPasswords no
  IdleTimeout 30m Autentykacja za pomocą haseł - włączanie, zezwolenie na puste hasła (ja bym to wyłączył!) oraz czas rozłączania podczas oczekiwania na podanie hasła (tutaj – 30 minut).

- `# AllowHosts *.our.com friend.other.com`

  DenyHosts home.pl *.algonet.se krakow.tpnet.pl opole.tpnet.pl
  Z których maszyn można się łączyć za pomocą ssh.

- **CheckMail no** Czy po zalogowaniu ma się odbywać sprawdzanie poczty.

- **AllowTcpForwarding no** Przy takim ustawieniu nie jest możliwe tunelowanie ftp, ale maszyna jest bezpieczniejsza.

- **AccountExpireWarningDays 30** Powiadamianie o kończeniu się ważności konta (dni)

- **PasswordExpireWarningDays 14** Powiadamianie o kończeniu się ważności hasła (dni)
- **ForcedPasswdChange yes** Wymuszanie zmiany już nieważnego hasła

Opcje konfiguracji występujące w ssh2 a nie występujące w ssh1

- **IdentifyFile identification**
  AuthorizationFile authorization
  PublicHostKeyFile hostkey.pub
  HostKeyFile hostkey
  RandomSeedFile random_seed
  Nazwa plików. Jeśli nie podano ścieżki - konfiguracja w */etc/ssh2*
- **Ciphers AnyStd** Wybór algorytmu szyfrowania (3DES, IDEA, Blowfish...) Tutaj każdy standardowy.
- **ForwardAgent yes**
  ForwardX11 yes Przekazywanie komunikacji za pomocą SSH2. Umożliwia to np szyfrowanie przez SSH2 połączenia graficznego X-Window.
- **PasswordAuthentication yes**
  PasswordGuess 3 Autentyfikacja za pomocą haseł i liczba dopuszczalnych prób podania hasła.
- **PubKeyAuthentication yes** Autentykacja za pomocą klucza publicznego/prywatnego (tak/nie)
- **ForcePTTYAllocation no** Czy podczas logowania wymusić przydzielenie konsoli (ttyv). Lepiej jest to wyłączyć, wtedy konsola zostanie przydzielona kiedy to będzie niezbędne.
- **UserConfigDirectory "%D.ssh2"** ścieżka do osobistego pliku konfiguracji ssh2
- **Ssh1Compatibility yes**
  Sshd1Path <set by configure> Kompatybilność z ssh1. Należy to włączyć. Sshd1Path nie trzeba robić bo zostało to zrobione przez skrypt *configure*.

Ważną opcją jest PermitRootLogin, jeżeli jest włączone, to umożliwia zdalne logowanie się użytkownika root. Wiekszość specjalistów radzi żeby tę opcję wyłączyć. Jeżeli administrator będzie chciał zrobić coś root'owego, to zawsze może zalogować się jako inny użytkownik a potem wykonać komendę su -l.

UserAllow, UserDeny, GroupAllow, GroupDeny - określa którzy użytkownicy (grupy) mają prawo do łączenia się za pomocą SSH. Jeżeli ustawimy UserAllow to połączą się tylko użytkownicy tam opisani. Jeżeli ustawimy UserDeny, to ci opisani nie będą mogli się połączyć. Jeżeli obydwie opcje zostaną użyte to łączyć się będą mogli tylko ci użytkownicy, którzy są opisani w UserAllow i jednocześnie nie są opisani w UserDeny. Podobnie ma się sprawa dla grup.

HostAllow i HostDeny określa maszyny z których można sie logować na dany komputer.

CompressionLevel decyduje o kompresji danych podczas transmisji.

6.5 Klient SSH

Razem z serwerem na maszynie instaluje się również klient ssh. Zmiena konfiguracji klienta odbywa się poprzez plik */etc/ssh_config* dla ssh1 i */etc/ssh2/ssh_config* dla ssh2. Zawartość tych plików prezentujemy poniżej (nie opisujemy opcji które już zostały opisane w części o konfiguracji serwera).

- **RSAAuthentication yes**
  TISAuthentication no Wybór metody autentykacji (należy wybrać tylko RSA)
- **PasswordPromptHost yes**
  PasswordPromptLogin yes Czy program ma pytać o hasła.
- **FallBackToRsh no**
  UseRsh no Możliwość użycia rsh w przypadku niepowodzenia połączenia za pomocą ssh. Możan włączyć, ale administrator zdalnej sieci na pewno wyłączył taką możliwość.
- **BatchMode no** Możliwość użycia ssh w trybie wsadowym
- **EscapeChar ~** Jaki znak powoduje wyjście z połączenia
- **Cipher 3DES** Algorytm stosowany do szyfrowania danych w połączeniu ze zdalną maszyną.
- **Compression yes**
  CompressionLevel 9 Kompresja - domyślnie jest włączona, poziom – 6. Dziewięć jest najwyższym, 0 wyłącza.
- **IdentifyFile ~/.ssh/identify** Położenie i nazwa pliku identyfikacji

Oczywiście konfiguracja klienta ssh2 trochę się różni:

- **VerboseMode no** Czy program ma podawać dodatkowe informacje.
- **PasswordPrompt "%'s password: "** Forma pytania o hasło użytkownika
- **Ssh1AgentCompatibility none**
  #Ssh1AgentCompatibility traditional
  #Ssh1AgentCompatibility ssh2 Wybór kompatybilności agenta ssh2. Najczęściej wybiera się gdyż w przypadku niepowodzenia identyfikacji powinno odpalać się klienta ssh1.
- **#LocalForward "110:pop3.ssh.fi:110"**
  #RemoteForward "3000:foobar:22" Przykładowe tunelowanie połączeń.
  Np.: 110:pop3.ssh.fi:110 przekaże połączenie klienta poczty pop3 za pomocą ssh. W tym przypadku wyłączone. Nie każdy użytkownik może przekazać połączenie na każdy port.

6.6 Secure Copy – scp i sftp

FTP podobnie jak telnet nie jest bezpieczną usługą. Hasła i w tym przypadku są przesyłane bez żadnego szyfrowania. Rozwiązaniem na to są serwery scp i sftp które instalują się razem z serwerem ssh.

7 Firewall

7.1 Idea firewalla

Najpierw określmy czym firewall nie jest: nie jest on zwykłym ruterem, hostem, czy też zbiorem systemów zapewniających bezpieczeństwo w sieci. Jest to raczej pewne podejście do bezpieczeństwa. Pomaga wprowadzić politykę bezpieczeństwa definiującą, które z usług oraz czyj dostęp jest dozwolony. Firewall to implementacja tej polityki w konfiguracji sieci, jednego lub więcej hostów i ruterów, oraz innych aspektów bezpieczeństwa takich jak np. zaawansowane uwierzytelnianie zamiast statycznych haseł. Głównym celem systemu ściany ogniowej jest kontrola dostępu do, lub z chronionej sieci. Implementuje ona politykę dostępu do sieci poprzez zmuszanie połączeń do przechodzenia przez firewall, gdzie mogą one być sprawdzone, i gdzie podejmowane są wobec nich określone akcje.

Firewall-em może być ruter, zwykły pecet, host lub zbiór hostów, ustawionych specjalnie tak, aby ochraniać sieć przed usługami które mogą być użyte w niecnych celach. Firewall zwykle jest umieszczany, na bramach wyższego rzędu, jak np. połączenie z Internetem, jakkolwiek może on być umieszczany też na niższych poziomach w celu zapewnienia ochrony przed pewnym mniejszym zbiorem hostów lub podsieci.

7.2 Dlaczego firewalle?

Głównym powodem zachęcającym do używania firewalli jest to, że bez nich systemy w podsieci są wystawione na atak na naturalnie niebezpieczne usługi takie jak NFS lub NIS, oraz próby ataku z hostów spoza swojej sieci. W systemach bez firewalli bezpieczeństwo sieci polega całkowicie na bezpieczeństwie hostów, i wszystkie hosty muszą współpracować, aby osiągnąć jadnakowo wysoki poziom bezpieczeństwa. Pomyłki i błędy w bezpieczeństwie zdarzają się coraz częściej, a włamania zdarzają się nie jako wynik złożonych ataków, ale po prostu poprzez błędy w konfiguracji i nieodpowiednie hasła.

Przy podejściu wykorzystującym firewalle, osiągamy następujące zyski:

1. Ochrona przed podatnymi na atak usługami - zwiększenie bezpieczeństwa i zredukowanie ryzyka dla hostów, poprzez filtrowanie niebezpiecznych usług. Dzięki temu tylko wybrane protokoły będą mogły przejść przez firewall. Ściana ogniowa może też chronić przed atakami związanymi z rutowaniem.

2. Kontrolowany dostęp do systemów - niektóre hosty mogą być osiągalne spoza firewalla, podczas gdy inne mogą być efektywnie chronione przed niechcianym dostępem. (np. dopuszczanie pakietów tylko do serwera mailowego). Jest to jedna z ważnych cech firewalla - nie udostępniać dostępu do hosta lub usług, które tego nie wymagają.

3. Skoncentrowane bezpieczeństwo - Firewall może być tańszy, niż inne sposoby zapewniania bezpieczeństwa poprzez możliwość skoncentrowania na systemie z firewallem dodatkowego oprogramowania związanego z bezpieczeństwem, w odróżnieniu np. od rozwiązań Kerberos, który wymaga modyfikacji na każdym z hostów z osobna (chociaż może on być bardziej odpowiedni niż firewall w określonych sytuacjach).

4. Ulepszona prywatność - możemy blokować takie usługi jak *finger* i DNS (dane o hostach wewnętrznych)

5. Logowanie i statystyki użytkowania sieci - jeśli cały ruch z, i do internetu przechodzi przez firewall możemy logować próby dostępu i otrzymać wartościowe statystyki dotyczące użytkowania sieci. Można także przy użyciu odpowiednich alarmów wykrywać próby skanowania firewalla lub ataku.

6. Przestrzeganie polityki bezpieczeństwa - być może najważniejsza rzecz, firewall dostarcza sposobów na zaimplementowanie i wymuszenie polityki dostępu.

7.3 Problemy z firewall-ami

Znając zalety firewalli nie należy zapominać, że mają one też swoje wady, że istnieje niemało rzeczy przed którymi one nie chronią. Firewall nie jest panaceum na wszystkie problemy bezpieczeństwa w Internecie.

Wiąże się z tym:

1. Ograniczony dostęp do usług pożądanych - najbardziej oczywiste to blokowanie dostępu do usług takich jak *telnet, ftp, X Windows, NFS*, z których użytkownicy często korzystają. Chociaż firewalle umożliwiają kontrolę na poziomie hostów, wszystko zależy od wprowadzonej polityki. Dobrze zaplanowana polityka, wyważona pomiędzy wymaganiami bezpieczeństwa, a potrzebami użytkowników może znacznie zredukować problem dostępu do usług.

2. Podatność na Backdoor-y - Firewalle nie chronią przed nimi. Jeśli mamy niechroniony dostęp modemowy do strony objętej firewallem, atakujący może go łatwo ominąć. Powstaje tylko pytanie: po co firewall, jeśli mamy dozwolony nieograniczony dostęp przez modem.

3. Małe zabezpieczenie przed atakami z wewnątrz - firewalle zwykle nie zapewniają ochrony przed atakami z sieci, które zabezpieczają. Firewall może być zaprojektowany do zapobiegania dostaniu się obób trzecich do ważnych danych, ale nie zapobiegnie zgraniu danych na taśmę i wyniesieniu ich poprzez osobę z wewnątrz.

4. inne problemy -

   - WWW, gopher - serwery i klienci WWW nie zostały zaprojektowane tak, aby współdziałać z firewallami, i można przez nie przekazać instrukcje do klienta, aby zmienić kontrolę dostępu i ważne pliki na hoście.
   - MBONE - transmisje multicastowe, wideo i audio enkapsulowane w innych pakietach. Firewalle zwykle przepuszczają pakiety bez kontroli zawartości.
   - wirusy - firewalle nie chronią uzytkowników przed ściąganiem zawirusowanych plików. Istnieje jednak możliwość skanowania przychodzącej poczty z wirusami, ale dzieje się to dopiero w późniejszym etapie nie związanym z firewallem.
   - przepustowość - firewall jest zwykle waskim gardłem, ponieważ wszystkie połączenia muszą przejść przez niego, i w niektórych przypadkach być sprawdzone. Chociaż dziś nie jest to już wielkim problemem, gdyż firewalle umieją przetwarzać dane w tempie ok 1.5 Megabita na sekundę.

- wszystko w jednym - system ściany ogniowej koncentruje bezpieczeństwo w jednym punkcie w odróżnieniu od innych technik. Włamanie do systemu z firewallem może być fatalne w skutkach dla słabiej chronionych hostów w podsieci.

Mimo tych wszystkich wad firewalle są mocno polecane do ochrony zasobów w połączeniu z innymi technikami i narzędziami.

7.4 Składniki firewalla

Głównymi częściami firewalla są:

- polityka bezpieczeństwa
- zaawansowane mechanizmy uwierzytelniania
- filtrowanie pakietów
- application gateways

7.4.1 Polityka bezpieczeństwa

Przy tworzeniu polityki należy zrozumieć i udokumentować następujące rzeczy:

- które usługi internetowe firma planuje używać, np. telnet, NFS
- gdzie usługi będą używane, np. tylko lokalnie, w Internecie, przy wdzwanianiu się z domu, przez zewnętrzne organizacje
- dodatkowe potrzeby, takie jak szyfrowanie czy obsługa wdzwaniania.
- jakie jest ryzyko związane z udostępnianiem tych usług i takiego dostępu
- jaki jest koszt kontrolowania, oraz jaki wpływ na użyteczność sieci ma wprowadzenie ochrony
- porównanie bezpieczeństwo *versus* użyteczność: czy bezpieczeństwo przeważy, jeśli usługa jest zbyt niebezpieczna lub zbyt kosztowna do ochrony.

7.4.2 Co powinien zapewniać firewall

Nie ma jednoznacznej odpowiedzi na tak postawione pytanie - można jedynie podać pewne wskazówki. Firewall powinien:

- obsługiwać politykę typu "zablokować wszystkie usługi oprócz tych które są dozwolone", nawet jeśli polityka tego nie przewiduje
- implementować naszą politykę bezpieczeństwa, a nie czyjąś.
- być elastyczny, powinno dać się go dostosować do nowych usług i potrzeb, jeśli polityka organizacji się zmieni. firewall powinien posiadać zaawansowane techniki uwierzytelniania, lub powinien umożliwiać obsługę takowych po ich zainstalowaniu.

- używać technik filtrowania pakietów do zezwalania lub odmowy usług dla określonego hosta jeśli jest to potrzebne.
- mieć elastyczny,"user friendly", język filtrowania IP, powinien filtrować jak najwięcej atrybutów, włączając w to adresy źródłowe i docelowe, typy protokołów, porty TCP/UDP, oraz interfejsy wejściowy i wyjściowy.
- korzystać z usług proxy dla *ftp* i *telnetu*, aby uwierzytelnianie było przeprowadzane centralnie na firewallu. Jeśli wymagane są usługi typu *nntp, X, http, gopher* firewall powinien posiadać odpowiednie serwery proxy.
- posiadać możliwość scentralizowania dostępu SMTP, aby zredukować bezpośrednie połączenia pomiędzy naszym serwisem a systemami zdalnymi. Dzięki temu mamy centralnie obsługiwany serwis mailowy.
- przystosowywać publiczny dostęp do serwisu, np poprzez oddzielenie publicznych serwerów informacji (które też mogą być chronione firewallem) od systemów, które nie potrzebują być dostępne z zewnątrz.
- jeśli firewall wymaga systemu operacyjnego (np. Uniksa), zabezpieczona wersja systemu powinna być częścią firewalla wraz z innymi narzędziami potrzebnymi do zapewnienia integralności hosta. System operacyjny powinien mieć poinstalowane patche.
- być rozwijany w sposób, którego siła i poprawność jest do zweryfikowania. Powinien być prosty w projektowaniu, aby mógł być zrozumiany i utrzymywany.
- powinien być uaktualniany regularnie przez zakładanie łat i poprawek.

7.5 Rodzaje firewalli

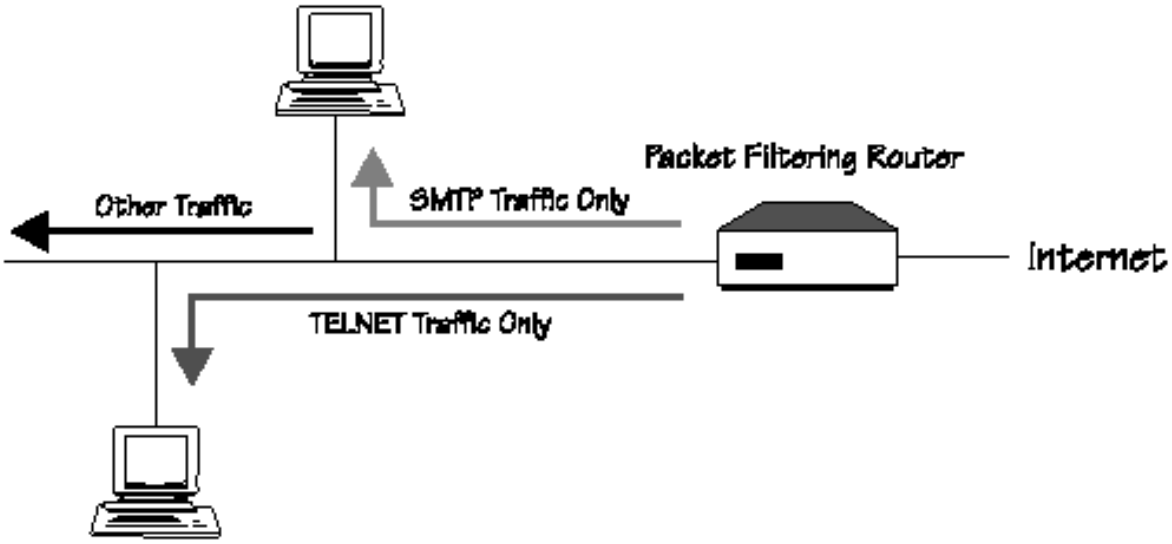
Istnieją dwa typy firewalli:

1. Firewalle filtrujące - blokujące wybrane pakiety sieciowe
2. Proxy serwery - które wykonują połączenia do nas.

7.5.1 Firewalle filtrujące

Filtrowanie pakietów działa na poziomie sieci. Dane mogą opuścić system jeśli reguły firewalla na to pozwalają. Kiedy pakiety nadchodzą, są filtrowane według typu, adresu źródłowego, docelowego, oraz informacji o porcie zawartej w każdym pakiecie.

Wiele ruterów posiada możliwość wykonywania usług firewalla. O firewallach filtrujących można myśleć jako o pewnego typu ruterach. Ponieważ niewiele danych jest analizowanych i logowanych, firewalle tego typu zużywają mniej czasu procesora i powodują mniejsze opóźnienia w naszej sieci. Firewalle filtrujące nie dostarczają możliwości wykorzystania uwierzytelniania przy pomocy haseł. Identyfikują one użytkownika tylko po adresie IP przypisanym do jego komputera. Może to być problemem gdy używamy DHCP (dynamicznego przydzielania adresów IP).



Rysunek 4: Przykładowy firewall filtrujący

Ściany ogniowe tego typu są przezroczyste dla użytkownika, nie musi on ustawiać żadnych specjalnych konfiguracji w swoich aplikacjach, aby używać Interetu. Przy serwerach proxy zwykle jest inaczej.

Które protokoły powinniśmy filtrować? Zależy to od polityki dostępu, ale następujące usługi są zwykle blokowane:

- tftp, port 69, trivial FTP, używany do bootowania stacji bezdyskowych, terminali serwerów i ruterów. Może być użyty do odczytania dowolnego pliku w systemie jeśli jest niepoprawnie skonfigurowany
- X Windows, OpenWindows, porty od 6000, port 2000, może wyciec tędy informacja o wpisywanych hasłach.
- RPC, port 111, usługi typu Remote Procedure Call (NFS, NIS), może być użyte do uzyskania haseł oraz pisania lub czytania z plików.
- rlogin, rsh, rexec, porty 513,514 i 512, usługi które niewłaściwie skonfigurowane mogą pozwolić na nieautoryzowany dostęp do konta i shella.

Inne usługi, bezpieczne czy też nie są zwykle filtrowane i ograniczane tylko do tych których system potrzebuje. Mogą to być np. telnet (port 23) i FTP (porty 20 i 21) - ograniczane tylko do określonych hostów, SMTP (port 25)- ograniczany tylko do centralnego serwera mailowego, RIP (port 520) - może być użyty do przekierowywania pakietów, oraz DNS (port 53) - zawierający nazwy hostów wewnętrznych, mogące być przydatne dla atakującego, UUCP (port 540), NNTP (port 119), gopher,http (porty 70 i 80).

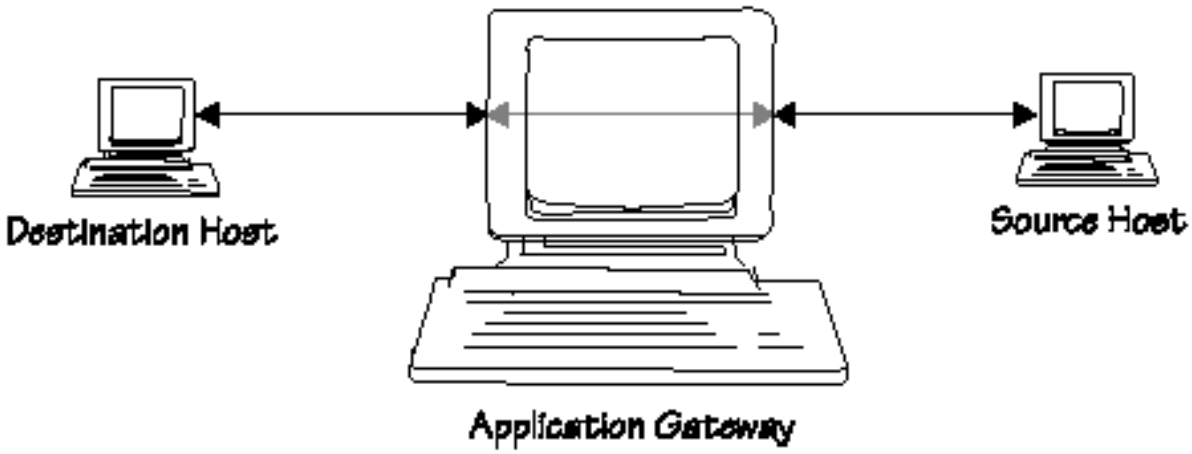
7.5.2 Serwery proxy

Proxy jest głównie używane do kontrolowania lub monitorowania ruchu wychodzącego. Niektóre aplikacje proxy cache'ują żądane dane. Zmniejsza to wymagania pasma i

zmniejsza czas dostępu do tych samych danych przez następnego użytkownika. Istnieją dwa typy serwerów proxy:

- Application Proxies - które wykonują pracę za nas
- Proxy SOCKS - które krosują porty

Application Proxy Najlepszym przykładem będzie osoba telnetująca się do innego komputera i dopiero stamtąd telnetująca się do świata zewnętrznego. Z application proxy proces ten jest zautomatyzowany. Gdy telnetujemy się na zewnątrz klient najpierw wysyła nas najpierw do proxy. Wtedy proxy łączy się z serwerem, z którym chcemy się połączyć i zwraca dane do nas.



Rysunek 5: Application Proxy

Używanie tego typu serwerów pozwala na:

- ukrywanie informacji, nazwy systemów wewnętrznych nie muszą zostać udostępnione poprzez DNS systemom zewnętrznym, ponieważ ten proxy serwer jest jedynym, który musi być widziany z zewnątrz.
- silne uwierzytelnianie i logowanie, zanim informacja osiągnie host docelowy
- efektywność cenowa - ponieważ oprogramowanie lub sprzęt do uwierzytelniania lub logowania powinno być umieszczone tylko w gateway-u.
- mniej złożone reguły filtrowania - będą one mniej złożone na ruterze filtrującym pakiety niż gdyby ruter musiał filtrować ruch aplikacji i kierować je do określonych systemów. Ruter powinien po prostu pozwalać na ruch aplikacji w stronę application proxy i odrzucać resztę.

Wadą serwerów tego typu to, iż w przypadku protokołów typu klient-serwer, jak np. telnet, trzeba wykonać dwa kroki, aby połączyć się na zewnątrz lub do wewnątrz.

Ponieważ serwery proxy obsługują całą komunikację, moga one logować dla na wszystko co chcemy. Dla proxy HTTP może to być np. URL, z którym się łączymy. Dla

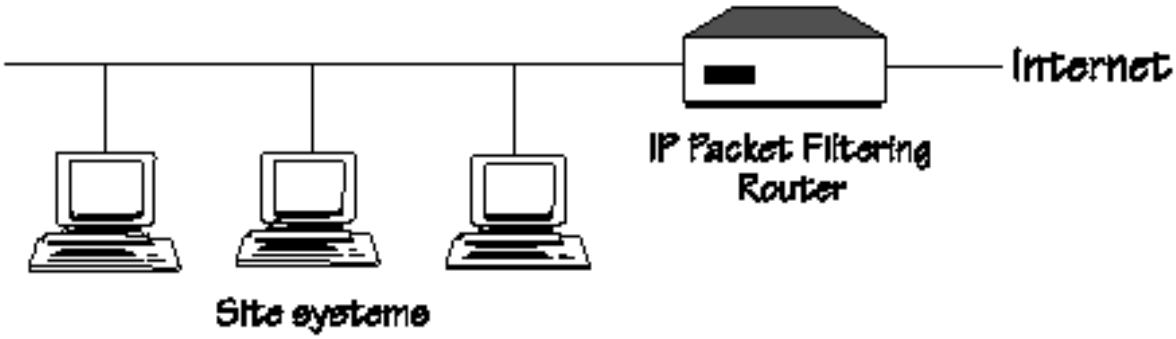
proxy FTP może to być każdy plik, który ściągamy. Proxy mogą nawet filtrować "nieodpowiednie" słowa ze stron, które odwiedzamy, lub szukać wirusów.

Proxy SOCKS Serwer SOCKS może przypominać tablicę z przełącznikami. Po prostu krosuje nasze połączenia przez system do innych połączeń wyjściowych na innych portach. Większość serwerów SOCKS obsługuje tylko połączenia TCP. I tak jak firewalle filtrujące nie umożliwiają uwierzytelniania użytkowników. Mogą jednakże zapisywać gdzie użytkownik chciał się połączyć.

7.6 Architektura Firewalli

Istnieje wiele sposobów ukształtowania naszej sieci do używania firewalli, np.:

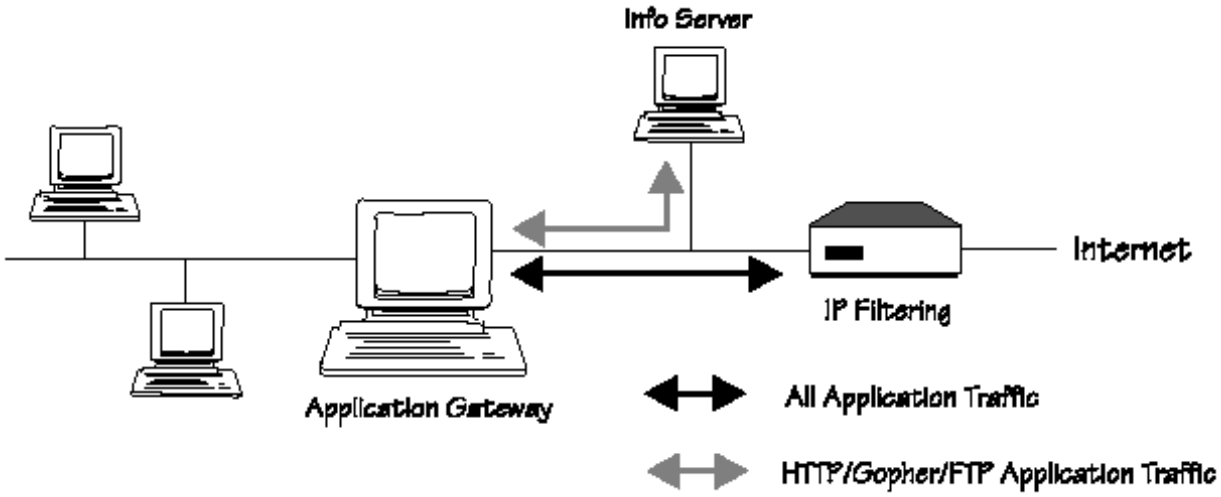
- *ruter ekranujący* - najprostsze z możliwych rozwiązań i najczęściej używane dla małych nieskomplikowanych sieci. Zasadniczo ten typ zabezpieczenia pozwala na separację ruchu pomiedzy sieciami lub konkretnymi urządzeniami na poziomie protokołów IP, IPX i MAC nadawcy lub odbiorcy. Stosuje się go jako rozszerzenie możliwości funkcjonującej już struktury połączenia sieci chronionej i publicznej. Zwykle pozwala się na bezpośredni dostęp do Internetu komputerom wewnątrz sieci chronionej, oraz zablokowanie dostępu z zewnątrz dla prawie wszystkich.



Rysunek 6: Ruter ekranujący

Wady takiego rozwiązania to słabe możliwości logowania, trudności w testowaniu wszystkich możliwości reguł filtrowania, trudność zarządzania przy złożonych regułach filtracji, oraz konieczność posiadania przez każdy z systemów z osobna mechanizmów autentykacji.

- *firewall z podwójną bramą* - ulepszona wersja firewalli z ruterem ekranującym. Składa się z hosta z dwoma interfejsami sieciowymi i z zablokowaną możliwością forward'owania pakietów IP. Ponadto ruter ekranujący umieszczony przy wyjściu do Internetu zapewnia dodatkową ochronę. Może to stworzy wewnętrzną ekranowaną podsieć, używaną przez specjalizowane systemy (np. WWW). Inaczej niż przy filtrowaniu pakietów, brama wewnętrzna całkowicie blokuje ruch IP pomiędzy Internetem, chronioną siecią. Jest to prosty firewall, i to dość bezpieczny. Usługi i dostęp są serwowane przez serwery proxy w bramie.



Rysunek 7: Firewall z podwójną bramą

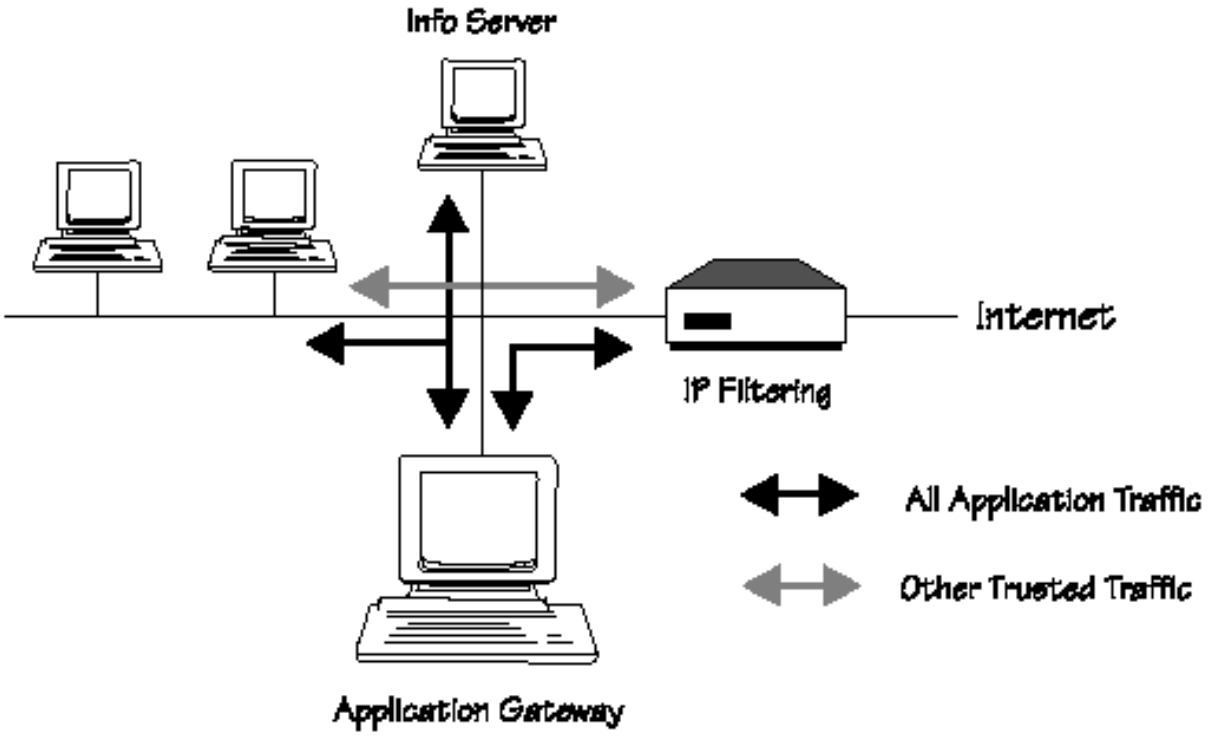
Typ ten implementuje zasadę polityki: blokuj wszystko, chyba, że jest to dozwolone. Ważne jest, aby host użyty jako firewall był bardzo dobrze zabezpieczony, gdyż użycie podatnych usług lub technik może prowadzić do włamań.

- *Brama ekranowana* - Rozwiązanie to jest często stosowaną konstrukcją składającą się ze współpracujących ze sobą rutera filtrującego pakiety i komputera-twierdzy. Powinien być to komputer posiadający trudny do złamania system haseł i zabezpieczeń, oraz monitorujący działania użytkowników. Zazwyczaj komputer ten pracuje w wewnętrznej, chronionej sieci, a ruter zapewnia że maszyna ta jest jedyną, która jest widziana z zewnątrz. Możliwe jest to dzięki filtrowaniu pakietów przechodzących przez ruter, i przesyłania do wewnątrz tych, które spełniają przyjete kryteria. Zazwyczaj ogranicza się też dostęp do wybranych usług sieciowych.

  Wada: teraz mamy dwa systemy, które należy ostrożnie skonfigurować. Jak wcześniej testowanie złożonych reguł filtrowania może być trudne i może prowadzić do pomyłek, i w konsekwencji dziur w ruterze.

- *Ekranowana podsieć* - Jest to rozwiązanie zbliżone do poprzedniego, jednakże wykorzystywanego dwukrotnie. Dla zapewnienia bezpieczeństwa dostępu wydzielono podsieć, która zazwyczaj jest widoczna i dostępna zarówno od strony sieci publicznej jak i chronionej. Jest to zapewnione przez odpowiednio skonfigurowane rutery filtrujące pakiety pod kątem adresata i nadawanych informacji.

  Dostęp do sieci publicznej z wewnętrznej sieci chronionej jest realizowana dwuetapowo poprzez komputer pracujący w dodatkowej podsieci. Dla zapewnienia bezpiecznej, lecz bezpośredniej pracy, w podsieci pośredniczącej komputery-twierdze, umożliwiające dzięki odpowiednim programom lub konfiguracji komunikację na poziomie niektórych usług sieciowych. Wykorzystanie dodatkowej podsieci chronionej, ma na celu taką konfigurację sieci, aby możliwe było bezpieczne udostępniane pewnych zasobów dla sieci publicznej, przy jednoczesnej blokadzie dostępu do sieci chronionej z zewnątrz. Jest to tzw. strefa zdemilitaryzowana (DMZ). Aby zrealizo-

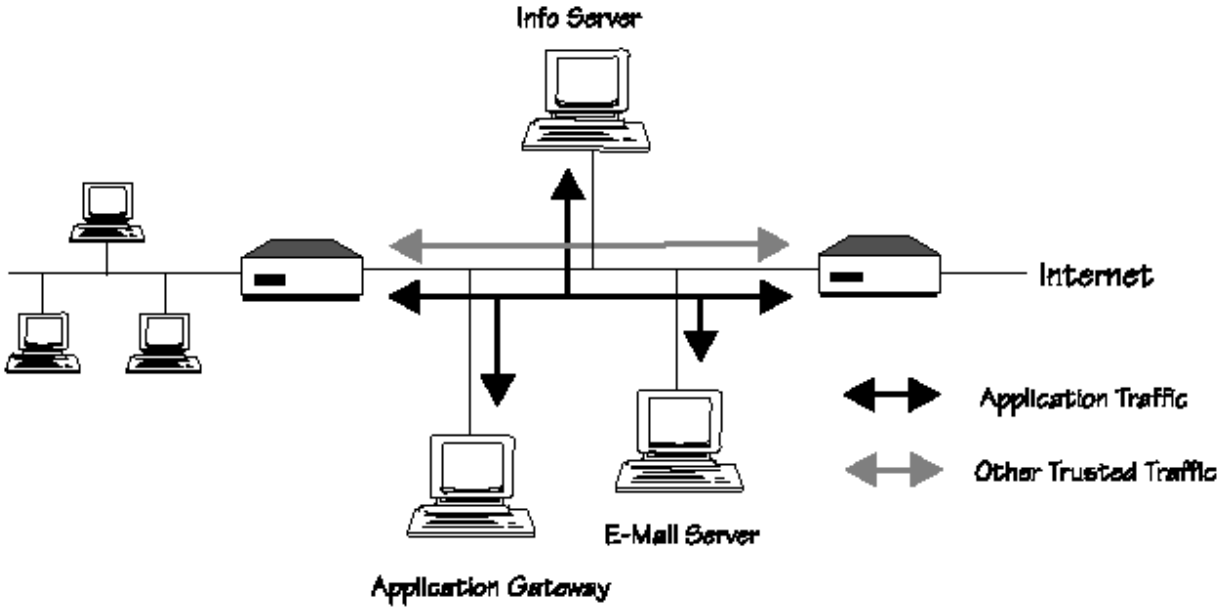


Rysunek 8: Brama ekranowana

wać to należy umieścić komputery i urządzenia, które chcemy udostępnić wszystkim, w tej podsieci. Do administratora należy właściwa konfiguracja tych urządzeń tak, aby nie stały się "bocznymi drzwiami" do chronionego systemu, oraz czuwanie nad pracą serwera pośredniczącego.

Ogólne wskazówki dotyczące bezpieczeństwa

1. Do not assume anything
2. Trust no-one,nothing
3. Nothing is secure
4. Security is a trade-off with usability
5. Paranoia is your friend



Rysunek 9: Podsieć ekranowana