Securitatea rețelelor wireless

Prof. Dr. Adrian Atanasiu

October 7, 2014

Atac: o încercare a unei entități neautorizate de a pătrunde în interiorul unui anumit sistem.

Atac: o încercare a unei entități neautorizate de a pătrunde în interiorul unui anumit sistem.
Atacurile sunt lansate speculând vulnerabilități ale sistemului și având diverse scopuri (sublinierea unor erori de construcție, atacuri malițioase, utilizarea resurselor sistemului, pregătirea unor atacuri ulterioare).
Pentru atingerea acestor deziderate, un atacator poate efectua atacul de mai multe ori sau poate lansa atacuri combinate plecate de la surse diferite.

Tipuri de atacatori

Începători (Script Kiddie):
Tineri dornici de afirmare, pentru care tentația de a se lăuda cu spargerea unor sisteme este mai importantă decât riscurile asumate.
Știu foarte puțin despre rețelele atacate, dar inventivitatea și spiritul de observație îi fac să descopere vulnerabilități și astfel să posede un bagaj de cunoștințe mai larg decât utilizatorii obișnuiți.
Marea lor majoritate se opresc după câteva încercări.

Tipuri de atacatori

Începători (Script Kiddie):
Tineri dornici de afirmare, pentru care tentația de a se lăuda cu spargerea unor sisteme este mai importantă decât riscurile asumate.
Știu foarte puțin despre rețelele atacate, dar inventivitatea și spiritul de observație îi fac să descopere vulnerabilități și astfel să posede un bagaj de cunoștințe mai larg decât utilizatorii obișnuiți.
Marea lor majoritate se opresc după câteva încercări.
În general începătorii nu fac deosebire între sistemele pe care le atacă. Uzual, ei preferă să construiască o secvență de atac pe care o îmbunătățesc treptat și o aplică de mai multe ori pentru a vedea ce resurse sunt vulnerabile. Majoritatea lor se pot încadra în genul malițios: atacă ceva care există, fără a fi interesați să obțină foloase deosebite.

Tipuri de atacatori

Crackeri:

Sunt adversari mai experimentați și deci mai periculoși. Se disting de începători prin abilitatea de a genera atacuri cu strategii bine definite asupra unor ținte specifice. Uzual, după 1985 ei sunt înglobați sub termenul "*hackeri*".

Tipuri de atacatori

Crackeri:

Sunt adversari mai experimentați și deci mai periculoși. Se disting de începători prin abilitatea de a genera atacuri cu strategii bine definite asupra unor ținte specifice. Uzual, după 1985 ei sunt înglobați sub termenul "*hackeri*".

Definiție

Un **hacker** *este o persoană care explorează detaliile sistemelor de programare căutând slăbiciuni pe care să le exploateze (spre deosebire de utilizatorii obișnuiți care preferă să învețe numai minimul necesar).*
Un **cracker** *este o persoană care sparge securitatea unui sistem.*

Tipuri de atacatori

Elite:

Crema celor care pot sparge securitatea unui sistem.

În general aici se grupează specialiști guvernamentali, teroriști, radicali politici.

Tipuri de atacatori

Elite:

Crema celor care pot sparge securitatea unui sistem.

În general aici se grupează specialiști guvernamentali, teroriști, radicali politici.

Sunt persoane despre care nu se aude aproape deloc și sunt bine plătiți sau fanatici.

Vulnerabilități

Orice rețea conține în mod inerent vulnerabilități; important este să depistăm cum/când/ unde/ de ce apar ele și – eventual – cum le putem controla.

Vulnerabilități software

Diverse studii folosesc ca o măsură a acurateții unui soft rata de erori găsită la fiecare 1000 linii de cod.
Se consideră că majoritatea softului produs în prezent conține între 5 și 15 erori la 1000 linii de cod.

Vulnerabilități software

Diverse studii folosesc ca o măsură a acurateții unui soft rata de erori găsită la fiecare 1000 linii de cod.
Se consideră că majoritatea softului produs în prezent conține între 5 și 15 erori la 1000 linii de cod.
Codurile scrise pentru un software se modifică permanent. Uneori, repararea unei slăbiciuni poate genera probleme în altă parte.

Vulnerabilități software

Diverse studii folosesc ca o măsură a acurateții unui soft rata de erori găsită la fiecare 1000 linii de cod.
Se consideră că majoritatea softului produs în prezent conține între 5 și 15 erori la 1000 linii de cod.
Codurile scrise pentru un software se modifică permanent. Uneori, repararea unei slăbiciuni poate genera probleme în altă parte.
Două părți independente de soft pot fi sigure dacă sunt luate separat, dar prin rularea lor împreună în cadrul unui sistem pot cauza noi falii de securitate.

Vulnerabilități software

Diverse studii folosesc ca o măsură a acurateții unui soft rata de erori găsită la fiecare 1000 linii de cod.
Se consideră că majoritatea softului produs în prezent conține între 5 și 15 erori la 1000 linii de cod.
Codurile scrise pentru un software se modifică permanent. Uneori, repararea unei slăbiciuni poate genera probleme în altă parte.
Două părți independente de soft pot fi sigure dacă sunt luate separat, dar prin rularea lor împreună în cadrul unui sistem pot cauza noi falii de securitate.
Există liste de discuții dedicate semnalării unor astfel de probleme (de ex. *BugTraq*) care arată cât de multe slăbiciuni se află în softurile actuale și în cât de multe produse de pe piață se află acestea.

Vulnerabilități hardware

Tip de vulnerabilitate mai puțin obișnuit, dar importanța sa crește datorită numărului tot mai mare de hard programabil aflat pe piață (telefoane mobile, tablete etc).
Vulnerabilități în sistemul intrare/ieșire (*BIOS*), procesorul de rețea sau *CPU* pot crea pagube mari, deoarece o vulnerabilitate hardware este mult mai dificil de corectat decât una software.

Vulnerabilități hardware

Tip de vulnerabilitate mai puțin obișnuit, dar importanța sa crește datorită numărului tot mai mare de hard programabil aflat pe piață (telefoane mobile, tablete etc).
Vulnerabilități în sistemul intrare/ieșire (*BIOS*), procesorul de rețea sau *CPU* pot crea pagube mari, deoarece o vulnerabilitate hardware este mult mai dificil de corectat decât una software.

Exemplu

Deși nu a specificat nici o legătură cu securitatea, Intel a oferit utilizatorilor în 1994 înlocuirea gratuită a procesoarelor Pentium din cauza unei erori în calculul cu virgulă mobilă.

Vulnerabilități de configurare

Administratorii de rețea fac – cu cele mai bune intenții – numeroase microconfigurări în rețelele pe care le gestionează. Într-un firewall cu o politică de acces mai complexă pot există sute de comenzi care permit sau interzic diferite tipuri de trafic. Deci sunt șanse mari ca unele din ele să intre în conflict.

Vulnerabilități de configurare

Administratorii de rețea fac – cu cele mai bune intenții – numeroase microconfigurări în rețelele pe care le gestionează. Într-un firewall cu o politică de acces mai complexă pot există sute de comenzi care permit sau interzic diferite tipuri de trafic. Deci sunt șanse mari ca unele din ele să intre în conflict.

Problema RFTM: exprimă frustrarea administratorilor de a se confrunta frecvent cu întrebări și nelămuriri care sunt explicate detaliat în manualele de utilizare. Deci, dacă persoanele responsabile cu dezvoltarea unei tehnologii nu știu prea multe despre acea tehnologie, șansele ca ea să lucreze conform standardelor scade semnificativ.

Vulnerabilități de configurare

Administratorii de rețea fac – cu cele mai bune intenții – numeroase microconfigurări în rețelele pe care le gestionează. Într-un firewall cu o politică de acces mai complexă pot există sute de comenzi care permit sau interzic diferite tipuri de trafic. Deci sunt șanse mari ca unele din ele să intre în conflict.

Problema RFTM: exprimă frustrarea administratorilor de a se confrunta frecvent cu întrebări și nelămuriri care sunt explicate detaliat în manualele de utilizare. Deci, dacă persoanele responsabile cu dezvoltarea unei tehnologii nu știu prea multe despre acea tehnologie, șansele ca ea să lucreze conform standardelor scade semnificativ.

O metodă foarte simplă de a elimina posibile erori de configurare este de a adopta o tehnologie de securitate cât mai simplu de gestionat.

Vulnerabilități de politică

O alegere neinspirată a unei politici de securitate poate permite lansarea unui atac.
În general se spune că securitatea unei rețele este la fel de sigură ca politica de securitate pe care o adoptă. Din acest motiv există un proces iterativ prin care securitatea unui sistem este îmbunătățită în timp prin modificări ale sistemului și ale politicilor.

Vulnerabilități de politică

O alegere neinspirată a unei politici de securitate poate permite lansarea unui atac.
În general se spune că securitatea unei rețele este la fel de sigură ca politica de securitate pe care o adoptă. Din acest motiv există un proces iterativ prin care securitatea unui sistem este îmbunătățită în timp prin modificări ale sistemului și ale politicilor.
Atacurile *DDoS* sunt exemple de vulnerabilități ale politicilor de securitate. Deși o primă idee apare în 1983, până prin anul 2000, astfel de atacuri nu erau nici măcar imaginate. Astăzi frecvența lor a generat diverse standarde și indicații pentru a proteja rețelele în fața atacurilor *DDoS*.

Vulnerabilități de exploatare

Faptul că o rețea poate fi exploatată într-un mod sigur nu înseamnă că un utilizator o va utiliza în mod sigur.
Vulnerabilitățile de exploatare apar când un utilizator – de obicei începător – violează politica de securitate și crează o breșă.

Vulnerabilități de exploatare

Faptul că o rețea poate fi exploatată într-un mod sigur nu înseamnă că un utilizator o va utiliza în mod sigur.
Vulnerabilitățile de exploatare apar când un utilizator – de obicei începător – violează politica de securitate și crează o breșă.

Exemplu

Utilizatorul adaugă un modem la calculatorul său de servici și se poate conecta la el înafara orelor de program.
În mod normal el instalează personal pe calculator softul respectiv și deci nu stabilește o politică de securitate (la care nici nu are acces de altfel).
Așa că un intrus poate folosi acest modem ca punct de lansare al unui atac asupra întregii rețele.

Rezultate atac

Toate atacurile sunt lansate pentru obținerea unor anumite rezultate.

Inițial au fost distinse patru tipuri de rezultate posibile. Ulterior a fost adăugat un al cincilea.

Dezvăluirea informației

Diseminarea informației oricărei persoane neautorizate de a avea acces la informația respectivă.

Aici sunt incluse furtul de parole, citirea unor componente de pe hard unde accesul nu este permis, dezvăluirea de informații confidențiale etc.

Coruperea informației

Orice alterare neautorizată a fișierelor stocate sau a datelor transmise prin rețea constituie o corupere de informații.

Printre numeroasele atacuri de acest gen amintim man-in-the-middle, viruși care distrug datele, înlocuirea de site-uri web.

Respingerea serviciului

Denial of Service (DoS) este degradarea intenționată sau blocarea resurselor unei rețele sau calculator.

Multe tipuri de atacuri flood au ca obiectiv un atac *DoS*, ca și spargerea intenționată a unei rețele pentru a avea acces la resursele sale cu care ulterior se va lansa atacul *DDoS*.

Furtul serviciului

Este utilizarea neautorizată a calculatorului sau rețelei fără a degrada serviciile celorlaltor utilizatori.

Furtul unei parole și logarea cu aceasta la rețea este un exemplu tipic.

Mărirea accesului

Mărirea accesului (Increased access) este rezultatul unei extinderi a privilegiilor unui utilizator când apelează la serviciile unei rețele. El precede de fapt cele patru atacuri anterioare.

Exemplul tipic este *buffer flow attack* care duce la creșterea accesului neautorizat.

Taxonomia atacului

Sunt cuprinse principalele tipuri de atac contra rețelelor precum și consecințele pe care le pot avea acestea.
Principalele clase de atacuri sunt:

- **Citire**: Se poate obține acces la informația neautorizată.
- **Manipulare**: Se poate prelucra/manipula informația.
- **Spoof**: Se oferă sau se răspunde cu informații false.
- **Flood**: Se blochează resursele sistemului.
- **Redirect**: Se modifică fluxul de informații.
- **Composite**: Combinație între clasele anterioare.

Caracteristici

Pentru fiecare atac posibil vom lua în considerare următoarele:

- **Membru al clasei/subclasei**: Se referă la clasa/subclasa căruia îi aparține atacul respectiv.
- **Exemplu de implementări**: Oferă exemple ale atacului asupra unei rețele, website, computer etc.
- **Preliminarii**: Lista atacurilor care premerg sau urmează atacului în cauză.
- **Vulnerabilități**: Lista celor mai comune vulnerabilități pe care le urmărește atacul.
- **Utilizare tipică**: Descrie cea mai obișnuită apariție a atacului.

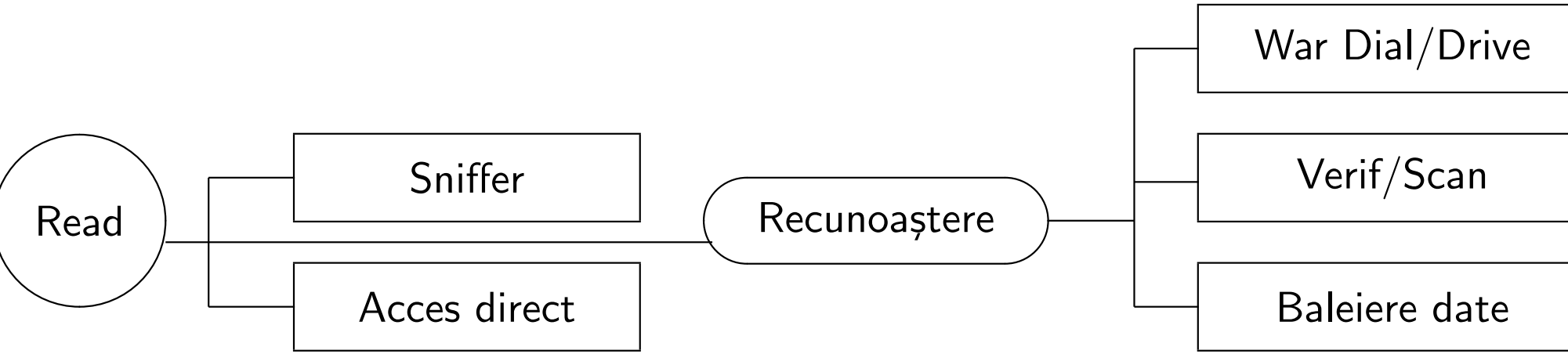
Caracteristici (2)

- **Rezultate atac**: Nominalizează cea mai utilizată clasă.
- **Posibile atacuri ulterioare**: Listează cele mai probabile atacuri care pot urma acestui tip de atac.
- *OSI* **Layers**: O listă a celor mai comune nivele din *OSI* (Open Systems Interconnection) folosite în atac.
- **Detectare**: O listă a tehnologiilor de securitate capabile să detecteze atacul.
- **Protecție**: O listă a tehnologiilor de securitate existente care pot opri atacul.

Clasa "Citire"

Sunt atacuri primare care au ca scop obținerea de informații despre potențiala victimă.
Ele merg de la aflarea modului de organizare al adreselor *IP* până la scanarea porturilor și a vulnerabilităților lor.
Sunt urmate de accesarea datelor (prin exploatarea slăbiciunilor detectate).

O descriere grafică a clasei de atac "**Citire" (Read)** este:



Recunoaștere (Recon)

Sunt primele atacuri, desemnate de atacator pentru a afla mai multe informații despre sistem.
Ele îi tatonează securitatea folosind metode active sau pasive.
În aproape toate cazurile informațiile obținute sunt exploatate iterativ, iar după ce se obțin suficiente date, sunt urmate de alte atacuri cu potențial crescut de virulență.

Baleierea datelor (Data Scavenging)

1. **Membru al clasei/subclasei**: Citire/Recunoaștere.
2. **Exemplu de implementări**: Utilități oferite de rețea: *Whois, Nslookup, Finger, Traceroute, Ping*. (*http://google.com*).
3. **Preliminarii**: Niciunul.
4. **Vulnerabilități**: Niciuna.
5. **Utilizare tipică**: Citește domeniile *IP*, serverele *DNS*, serverele de mail, sistemele publice, punctele de contact etc.
6. **Rezultate atac**: Dezvăluie informații.
7. **Posibile atacuri ulterioare**: Verificare și scanare.
8. *OSI* **Layers**: $3 - 7$
9. **Detectare**: Aproape imposibilă.
10. **Protecție**: Niciuna

Atacul de baleiere a datelor este în general primul pas în orice atac deliberat asupra unei rețele. Atacatorul combină utilități ale rețelei cu motoare de căutare pe Internet obținând cât mai multe informații primare despre victimă.

Atacul de baleiere a datelor este în general primul pas în orice atac deliberat asupra unei rețele. Atacatorul combină utilități ale rețelei cu motoare de căutare pe Internet obținând cât mai multe informații primare despre victimă.
Este aproape imposibil de detectat din două motive:

1. Dacă folosește utilități de rețea (*Ping*, *Traceroute* etc), volumul de trafic este atât de scăzut încât este imposibil de făcut diferența între utilizarea legitimă a protocoalelor și intenția de atac.
2. Informația obținută cu *Whois*, *Nslookup* sau motoarele de căutare Internet este de obicei publică.

Atacul de baleiere a datelor este în general primul pas în orice atac deliberat asupra unei rețele. Atacatorul combină utilități ale rețelei cu motoare de căutare pe Internet obținând cât mai multe informații primare despre victimă.
Este aproape imposibil de detectat din două motive:

1. Dacă folosește utilități de rețea (*Ping*, *Traceroute* etc), volumul de trafic este atât de scăzut încât este imposibil de făcut diferența între utilizarea legitimă a protocoalelor și intenția de atac.
2. Informația obținută cu *Whois*, *Nslookup* sau motoarele de căutare Internet este de obicei publică.

Adesea informația obținută vine de la servere diferite de cel al victimei (cazul căutărilor *Whois*).

După un atac de baleiere a datelor, atacatorul va putea dispune de

- Adresele *IP* ale sistemelor critice (WWW, DNS, mail);

După un atac de baleiere a datelor, atacatorul va putea dispune de

- Adresele *IP* ale sistemelor critice (WWW, DNS, mail);
- Domeniile *IP* asignate victimei;

După un atac de baleiere a datelor, atacatorul va putea dispune de

- Adresele *IP* ale sistemelor critice (WWW, DNS, mail);
- Domeniile *IP* asignate victimei;
- Providerul Internet (*ISP*) al victimei.

Verificare și scanare

Caracteristici generale ale atacului "**Verif/Scan**":

1. **Membru al clasei/subclasei**: Citire/Recunoaștere.
2. **Exemplu de implementări**: Nmap, Nessus.
3. **Preliminarii**: Baleiere date.
4. **Vulnerabilități**: Niciuna.
5. **Utilizare tipică**: Citește *IP*urile și aplicațiile accesibile din rețeaua victomei.
6. **Rezultate atac**: Dezvăluie informații.
7. **Posibile atacuri ulterioare**: Aproape toate.
8. ***OSI* Layers**: $3 - 7$
9. **Detectare**: *IDS* și firewall-uri (cu *log analysis*)..
10. **Protecție**: Niciuna (un firewall poate limita unele ținte de scanare, dar nu o scanare în curs pentru un serviciu free la o adresă *IP* accesibilă).

Acest atac este adesea numit și "*Scanarea porturilor*" sau "*Scanarea vulnerabilităților*".
Atacatorul folosește informația obținută prin atacul de baleiere pentru a afla mai multe informații despre rețeaua victimă. De obicei se face o scanare a porturilor, urmată de o scanare a vulnerabilităților.

Acest atac este adesea numit și "*Scanarea porturilor*" sau "*Scanarea vulnerabilităților*".
Atacatorul folosește informația obținută prin atacul de baleiere pentru a afla mai multe informații despre rețeaua victimă. De obicei se face o scanare a porturilor, urmată de o scanare a vulnerabilităților.
Folosind un tool ca *Nmap*, atacatorul poate afla:

- Toate adresele *IP* accesibile ale rețelei victimă;

Acest atac este adesea numit și "*Scanarea porturilor*" sau "*Scanarea vulnerabilităților*".
Atacatorul folosește informația obținută prin atacul de baleiere pentru a afla mai multe informații despre rețeaua victimă. De obicei se face o scanare a porturilor, urmată de o scanare a vulnerabilităților.
Folosind un tool ca *Nmap*, atacatorul poate afla:

- Toate adresele *IP* accesibile ale rețelei victimă;
- O șansă non-neglijabilă de a ghici care *OS* al fiecărui sistem accesibil este activ;

Acest atac este adesea numit și "*Scanarea porturilor*" sau "*Scanarea vulnerabilităților*".
Atacatorul folosește informația obținută prin atacul de baleiere pentru a afla mai multe informații despre rețeaua victimă. De obicei se face o scanare a porturilor, urmată de o scanare a vulnerabilităților.
Folosind un tool ca *Nmap*, atacatorul poate afla:

- Toate adresele *IP* accesibile ale rețelei victimă;
- O șansă non-neglijabilă de a ghici care *OS* al fiecărui sistem accesibil este activ;
- Dacă rețeaua este protejată de un firewall și – dacă da – de ce tip.

War Dialing și War Driving

1. **Membru al clasei/subclasei**: Citire/Recunoaștere.
2. **Exemplu de implementări**: War dialers: *Tone Loc*; War Driving: *Netstumbler*.
3. **Preliminarii**: Niciuna.
4. **Vulnerabilități**: De utilizare sau ale politicii folosite.
5. **Utilizare tipică**: Găsește modemurile nesigure sau *AP*-urile wireless conectate la rețeaua victimei.
6. **Rezultate atac**: Creșterea gradului de acces.
7. **Posibile atacuri ulterioare**: Sniffer.
8. ***OSI* Layers**: $1 - 2$
9. **Detectare**: Aproape imposibilă.
10. **Protecție**: Verificări pentru *AP*, controale regulate folosind tooluri war-driving. Pentru modeme se recomandă un audit periodic.

Atacurile War Dialing și War Driving permit atacatorilor să intre în rețeaua victimei fără a fi controlați (identificați).

Atacurile War Dialing și War Driving permit atacatorilor să intre în rețeaua victimei fără a fi controlați (identificați).
În primul caz atacatorul baleiează prin telefon prefixele numerelor asignate victimei sau zonei acesteia, căutând conexiuni de modem. Din lista astfel obținută, el poate ghici ce sisteme sunt conectate. Sunând apoi la aceste numere, un atacator poate trece ușor peste o mare parte din măsurile de securitate ale victimei, el fiind confundat cu un utilizator normal.

Atacurile War Dialing și War Driving permit atacatorilor să intre în rețeaua victimei fără a fi controlați (identificați).
În primul caz atacatorul baleiează prin telefon prefixele numerelor asignate victimei sau zonei acesteia, căutând conexiuni de modem. Din lista astfel obținută, el poate ghici ce sisteme sunt conectate. Sunând apoi la aceste numere, un atacator poate trece ușor peste o mare parte din măsurile de securitate ale victimei, el fiind confundat cu un utilizator normal.
Atacul *War Driving* operează similar, singura deosebire fiind aceea că atacatorul folosește o antenă montată pe mașină și se rotește în jurul locației fizice a victimei. Scopul este de a identifica punctele de acces *LAN* (*AP*-urile) slab securizate, prin care atacatorul se poate conecta direct la rețeaua victimei.

Sniffer

Se consideră atac *sniffer* orice situație când un intrus citește fără autorizație pachete de date de pe rețea, sau când acestea sunt deviate prin sistemul său.

Sniffer

Se consideră atac *sniffer* orice situație când un intrus citește fără autorizație pachete de date de pe rețea, sau când acestea sunt deviate prin sistemul său.
Scop: citirea într-un mod inteligent a informației utile atacatorului, cum ar fi:

- Informații de autentificare (parole);
- Patternuri tipice de utilizare din rețeaua victimei;
- Informații de gestiune a rețelei;
- Tranzacții confidențiale.

Sniffer

Se consideră atac *sniffer* orice situație când un intrus citește fără autorizație pachete de date de pe rețea, sau când acestea sunt deviate prin sistemul său.
Scop: citirea într-un mod inteligent a informației utile atacatorului, cum ar fi:

- Informații de autentificare (parole);
- Patternuri tipice de utilizare din rețeaua victimei;
- Informații de gestiune a rețelei;
- Tranzacții confidențiale.

Un virus sniffer (cum este *Ethereal*) poate decodifica informații până la nivele de aplicații foarte joase (protocoale *Gateway* sau *TLS/SSL*).

Caracteristici generale:

1. **Membru al clasei/subclasei**: Citire.
2. **Exemplu de implementări**: Ethereal.
3. **Preliminarii**: Redirecționare trafic sau *MAC* flooding.
4. **Vulnerabilități**: Niciuna.
5. **Utilizare tipică**: Citește traficul de pe rețea pentru care nu are permisiune. Obține parole.
6. **Rezultate atac**: Dezvăluie informații.
7. **Posibile atacuri ulterioare**: Acces direct.
8. ***OSI* Layers**: $2 - 7$
9. **Detectare**: Antisniff.
10. **Protecție**: Criptografică.

Un Sniffing nu este atac propriu zis, el având două puncte slabe:

1. Informația surprinsă trebuie să fie text clar. O criptare a datelor constituie un obstacol în citirea lor de către intruși.

Un Sniffing nu este atac propriu zis, el având două puncte slabe:

1. Informația surprinsă trebuie să fie text clar. O criptare a datelor constituie un obstacol în citirea lor de către intruși.
2. Pachetele citite trebuie trimise – sub o formă oarecare – atacatorului.

Un Sniffing nu este atac propriu zis, el având două puncte slabe:

1. Informația surprinsă trebuie să fie text clar. O criptare a datelor constituie un obstacol în citirea lor de către intruși.
2. Pachetele citite trebuie trimise – sub o formă oarecare – atacatorului.
   Dacă acesta este local și într-un mediu partajat (hub Ethernet, wireless), este suficient ca el să-și plaseze cardul de interfață la rețea (*NIC*).

Un Sniffing nu este atac propriu zis, el având două puncte slabe:

1. Informația surprinsă trebuie să fie text clar. O criptare a datelor constituie un obstacol în citirea lor de către intruși.
2. Pachetele citite trebuie trimise – sub o formă oarecare – atacatorului.
   Dacă acesta este local și într-un mediu partajat (hub Ethernet, wireless), este suficient ca el să-și plaseze cardul de interfață la rețea (*NIC*).
   Dacă mediul este protejat, atacatorul trebuie să realizeze în prealabil o deviere a traficului sau un gen de flooding Media Access Control (*MAC*).

Un Sniffing nu este atac propriu zis, el având două puncte slabe:

1. Informația surprinsă trebuie să fie text clar. O criptare a datelor constituie un obstacol în citirea lor de către intruși.
2. Pachetele citite trebuie trimise – sub o formă oarecare – atacatorului.
   Dacă acesta este local și într-un mediu partajat (hub Ethernet, wireless), este suficient ca el să-și plaseze cardul de interfață la rețea (*NIC*).
   Dacă mediul este protejat, atacatorul trebuie să realizeze în prealabil o deviere a traficului sau un gen de flooding Media Access Control (*MAC*).

Din acest motiv, unele firewall-uri nu consideră *Sniffer* drept virus, ci numai un instrument folosit de administratorii de rețea pentru a diagnostica a gamă largă de probleme de conexiune.

Acces direct

Caracteristici generale:

1. **Membru al clasei/subclasei**: Citire.
2. **Exemplu de implementări**: Logare la un server și furtul fișierului */etc/passwd*..
3. **Preliminarii**: Diverse.
4. **Vulnerabilități**: Niciuna.
5. **Utilizare tipică**: Acces neautorizat la setări de informații; furt de date.
6. **Rezultate atac**: Dezvăluie informații.
7. **Posibile atacuri ulterioare**: Manipulare.
8. *OSI* **Layers**: 7
9. **Detectare**: *IDS*.
10. **Protecție**: Firewall și aplicații de securitate.

În *Acces direct* sunt grupate toate atacurile în care intrusul încearcă să obțină un acces direct la resursele din rețea.

În *Acces direct* sunt grupate toate atacurile în care intrusul încearcă să obțină un acces direct la resursele din rețea.

Exemplu

Dacă intrusul a găsit o cale prin care să treacă prin firewall, el va utiliza un atac "Acces direct" pentru a se loga la sistemele protejate de acesta.
Apoi poate lansa o gamă largă de alte atacuri spre alte rețele, cel mai comun fiind un atac de prelucrare a informației.

În *Acces direct* sunt grupate toate atacurile în care intrusul încearcă să obțină un acces direct la resursele din rețea.

Exemplu

Dacă intrusul a găsit o cale prin care să treacă prin firewall, el va utiliza un atac "Acces direct" pentru a se loga la sistemele protejate de acesta.
Apoi poate lansa o gamă largă de alte atacuri spre alte rețele, cel mai comun fiind un atac de prelucrare a informației.

Deși *Acces Direct* este aproape întotdeauna lansat pe Nivelul 7, pentru unele aplicații el poate fi detectat și la nivele mai joase.

Clasa "Manipulare"

Această clasă conține toate atacurile a căror reușită duce la modificări sau transfer de date – indiferent de nivelul sistemului atacat.
Aici se află zeci de atacuri; cea mai mare parte din ele sunt cuprinse în două categorii: atacuri pe rețea și atacuri pe aplicație.

Clasa "Manipulare"

Această clasă conține toate atacurile a căror reușită duce la modificări sau transfer de date – indiferent de nivelul sistemului atacat.
Aici se află zeci de atacuri; cea mai mare parte din ele sunt cuprinse în două categorii: atacuri pe rețea și atacuri pe aplicație.
Intrușii folosesc aceste atacuri cu două scopuri:

- Manipulează pachetele de date extrăgând informație pe care o prelucrează în alte scopuri;

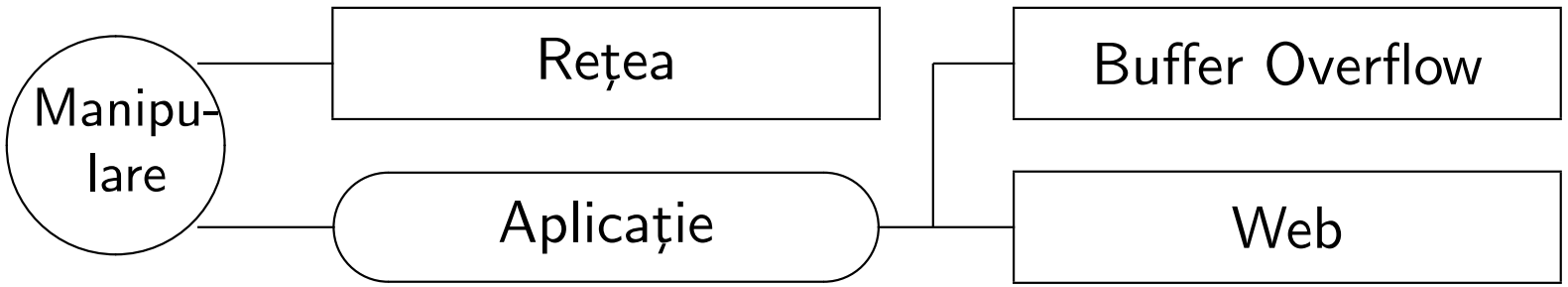
Clasa "Manipulare"

Această clasă conține toate atacurile a căror reușită duce la modificări sau transfer de date – indiferent de nivelul sistemului atacat.
Aici se află zeci de atacuri; cea mai mare parte din ele sunt cuprinse în două categorii: atacuri pe rețea și atacuri pe aplicație.
Intrușii folosesc aceste atacuri cu două scopuri:

- Manipulează pachetele de date extrăgând informație pe care o prelucrează în alte scopuri;
- Manipulează pachetele de date ale rețelei pentru a-i cauza diverse pagube (malițioase sau iremediabile).

Taxonomie

Structura clasei de atac "**Manipulare**" este



Atacuri pe rețea

Deși foarte numeroase, toate au însă aceleași caracteristici:

1. **Membru al clasei/subclasei**: Manipulare.
2. **Exemplu de implementări**: *Fragroute*.
3. **Preliminarii**: Diverse.
4. **Vulnerabilități**: Software.
5. **Utilizare tipică**: Tehnologii de evitare a securității.
6. **Rezultate atac**: Creșterea nivelului de acces.
7. **Posibile atacuri ulterioare**: Citire/Composite.
8. ***OSI* Layers**: $3 - 4$.
9. **Detectare**: *IDS*, routere.
10. **Protecție**: Firewall, aplicații de securitate, criptografie.

Cel mai frecvent atac de manipulare a rețelei este fragmentarea *IP*-ului.

Aici intrusul fragmentează intenționat traficul încercând să ocolească (bypass) controlul de securitate – care poate fi al rețelei (*IDS* sau firewall) sau al unei aplicații.

Programul tipic care lansează un atac de fragmentare este *Fragroute*.

Cel mai frecvent atac de manipulare a rețelei este fragmentarea *IP*-ului.
Aici intrusul fragmentează intenționat traficul încercând să ocolească (bypass) controlul de securitate – care poate fi al rețelei (*IDS* sau firewall) sau al unei aplicații.
Programul tipic care lansează un atac de fragmentare este *Fragroute*.
Înafara unui atac de fragmentare *IP*, atacatorul poate executa un atac la rădăcină (source route attack).
Acesta permite intrusului să transfere drumul atacului prin rețea.
Astfel de atacuri nu mai sunt primejdioase, ele fiind respinse automat de majoritatea routerelor.

Cel mai frecvent atac de manipulare a rețelei este fragmentarea *IP*-ului.
Aici intrusul fragmentează intenționat traficul încercând să ocolească (bypass) controlul de securitate – care poate fi al rețelei (*IDS* sau firewall) sau al unei aplicații.
Programul tipic care lansează un atac de fragmentare este *Fragroute*.
Înafara unui atac de fragmentare *IP*, atacatorul poate executa un atac la rădăcină (source route attack).
Acesta permite intrusului să transfere drumul atacului prin rețea.
Astfel de atacuri nu mai sunt primejdioase, ele fiind respinse automat de majoritatea routerelor.
Protocoalele *IP*, *TCP* (Transmission Control Protocol) și *UDP* (User Datagram Protocol) sunt deosebit de complexe. Aproape inevitabil, ele lasă posibilitatea atacatorilor să construiască softuri care obligă aceste protocoale să efectueze acțiuni neprevăzute de autorii lor.

Atacuri pe aplicație

Atacurile de manipulare a aplicației sunt atacuri îndreptate asupra softurilor pe nivele, având ca țintă exploatarea diverselor slăbiciuni în construirea sau implementarea aplicațiilor.
Cel mai cunoscut este atacul buffer overflow.

Atacuri pe aplicație

Atacurile de manipulare a aplicației sunt atacuri îndreptate asupra softurilor pe nivele, având ca țintă exploatarea diverselor slăbiciuni în construirea sau implementarea aplicațiilor.
Cel mai cunoscut este atacul buffer overflow.
Ulterior s-au dezvoltat și atacurile aplicațiilor web pentru rețele – de exemplu cross-site scripting sau insecure Common Gateway Interface (*CGI*).
Deoarece numărul și varietatea atacurilor pe aplicație este enormă, ne vom restrânge momentan la o analiză doar pentru aceste două tipuri de atac.

Buffer Overflow

Caracteristici:

1. **Membru al clasei/subclasei**: Manipulare/ Atacuri pe aplicație.
2. **Exemplu de implementări**: Vulnerabilități pe aplicații; *http://www.cert.org*.
3. **Preliminarii**: Acces direct.
4. **Vulnerabilități**: Software.
5. **Utilizare tipică**: Escaladarea privilegiilor pe mașina țintă.
6. **Rezultate atac**: Creșterea nivelului de acces.
7. **Posibile atacuri ulterioare**: Citire/Composite.
8. *OSI* **Layers**: 7
9. **Detectare**: *IDS*, aplicații de securitate.
10. **Protecție**: Aplicații de securitate.

Buffer Overflow este cea mai obișnuită formă de vulnerabilitate de aplicație. Ea apare când aplicația se dezvoltă neputând să-și limiteze resursele de memorie utilizate.

Buffer Overflow este cea mai obișnuită formă de vulnerabilitate de aplicație. Ea apare când aplicația se dezvoltă neputând să-și limiteze resursele de memorie utilizate.

Exemplu

Un program așteaptă la intrare 20 *octeți și primește de fapt* 300 *octeți. În mod normal, el ar trebui să ignore* 280 *din ei.*
Dacă însă aplicația are o eroare de cod, ea îi poate suprascrie în altă zonă de memorie și apoi să execute codul de acolo cu privilegiile aplicației originale. Dacă aplicația lucrează ca root, un atac reușit va obține pentru intrus privilegii de root.

Buffer Overflow este cea mai obișnuită formă de vulnerabilitate de aplicație. Ea apare când aplicația se dezvoltă neputând să-și limiteze resursele de memorie utilizate.

Exemplu

Un program așteaptă la intrare 20 octeți și primește de fapt 300 octeți. În mod normal, el ar trebui să ignore 280 din ei.
Dacă însă aplicația are o eroare de cod, ea îi poate suprascrie în altă zonă de memorie și apoi să execute codul de acolo cu privilegiile aplicației originale. Dacă aplicația lucrează ca root, un atac reușit va obține pentru intrus privilegii de root.

Din nefericire, atacurile buffer overflow, deși foarte vechi, sunt cele mai periculoase, în principal din cauza pagubelor pe care le provoacă precum și datorită marilor dificultăți de a le preveni.

Web

Caracteristici:

1. **Membru al clasei/subclasei**: Manipulare/ Atacuri pe aplicație.
2. **Exemplu de implementări**: *Cross-site scripting*, Aplicații *CGI* nesigure.
3. **Preliminarii**: Acces direct.
4. **Vulnerabilități**: Software.
5. **Utilizare tipică**: Variabilă.
6. **Rezultate atac**: Creșterea nivelului de acces și dezvăluirea informației protejate.
7. **Posibile atacuri ulterioare**: Citire/Composite.
8. *OSI* **Layers**: 7
9. **Detectare**: *IDS*, aplicații de securitate.
10. **Protecție**: Aplicații de securitate.

Atacurile pe aplicații Web sunt foarte diverse, cele două menționate fiind cele mai cunoscute atacuri de acest gen.
În Cross-site scripting, informația malițioasă este ascunsă într-un *URL* pe care victima va da click. Atacul poate afecta utilizatorii interni ai rețelei care folosesc această adresă.

Atacurile pe aplicații Web sunt foarte diverse, cele două menționate fiind cele mai cunoscute atacuri de acest gen.
În Cross-site scripting, informația malițioasă este ascunsă într-un *URL* pe care victima va da click. Atacul poate afecta utilizatorii interni ai rețelei care folosesc această adresă.
Caracteristic acestui atac web este faptul că nu există o modalitate unitară de a determina locul unde este problema; codul malițios poate fi pe browserul clientului, pe server, pe site-ul vizitat sau oriunde pe drumul spre acesta.
În general, cel mai bun mijloc de prevenire a lui este educarea utilizatorilor de a nu deschide site-uri decât dacă știu exact ce conțin.

Aplicații CGI nesigure

Similar atacurilor Citire, aplicațiile CGI nesigure pot constitui primul pas pentru un intrus care caută să compromită un server web.
Ori de câte ori se completează un formular sau se inserează adresa proprie pe un website există șanse de a fi victima unei forme de atac *CGI*.

Aplicații CGI nesigure

Similar atacurilor Citire, aplicațiile CGI nesigure pot constitui primul pas pentru un intrus care caută să compromită un server web.

Ori de câte ori se completează un formular sau se inserează adresa proprie pe un website există șanse de a fi victima unei forme de atac *CGI*.

Scripturile *CGI* curate nu acceptă – printre altele – decât tipuri de date de o formă specificată. De exemplu, dacă programul *CGI* solicită adresa unui utilizator, el trebuie să-i permită acestuia să folosească numai caractere alfanumerice, punctul și virgula. Pe de altă parte programul trebuie să permită și operarea cu caractere cum ar fi %, $ etc pentru alte scopuri.

Aplicații CGI nesigure

Similar atacurilor Citire, aplicațiile CGI nesigure pot constitui primul pas pentru un intrus care caută să compromită un server web.

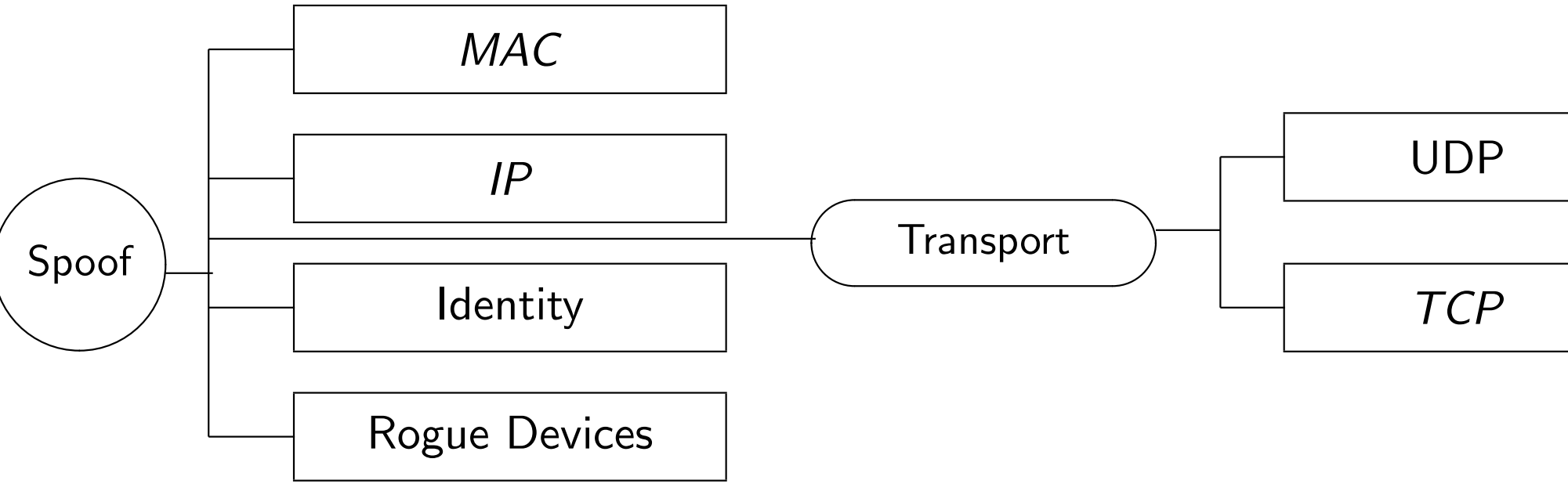
Ori de câte ori se completează un formular sau se inserează adresa proprie pe un website există șanse de a fi victima unei forme de atac *CGI*.

Scripturile *CGI* curate nu acceptă – printre altele – decât tipuri de date de o formă specificată. De exemplu, dacă programul *CGI* solicită adresa unui utilizator, el trebuie să-i permită acestuia să folosească numai caractere alfanumerice, punctul și virgula. Pe de altă parte programul trebuie să permită și operarea cu caractere cum ar fi %, $ etc pentru alte scopuri.

Practic, nu există însă programe *CGI* total sigure, singurul deziderat al dezvoltatorilor de soft fiind acela de a limita numărul de brese de securitate din aplicatiile construite

Clasa "Spoof"

Atacurile **Spoofing** apar atunci când un atacator este capabil să convingă un utilizator/sistem că o anumită informație provine de la o sursă care nu a creat-o.
Un atac spoofing poate fi lansat oriunde există protocoale slabe de autentificare.



MAC Spoofing

1. **Membru al clasei/subclasei**: Spoof.
2. **Exemplu de implementări**: Suporturi de rețea care suportă modificarea adresei *MAC* stocată pe cardul de acces.
3. **Preliminarii**: Acces direct (conectivitate *LAN* locală).
4. **Vulnerabilități**: Niciuna.
5. **Utilizare tipică**: Află o adresă *MAC* a unui sistem de încredere pentru a trimite sau primi date în numele acestuia.
6. **Rezultate atac**: Creșterea nivelului de acces și dezvăluirea informației protejate.
7. **Posibile atacuri ulterioare**: Variabilă.
8. ***OSI* Layers**: 2
9. **Detectare**: Niciuna.
10. **Protecție**: Memorie de adrese (*CAM* – Content Addressable Memory) statică.

MAC Spoofing (sau Layer 2 Spoofing) este un atac simplu care pune propria sa adresă *MAC* în locul aceleia a sistemului considerat de încredere.

MAC Spoofing (sau Layer 2 Spoofing) este un atac simplu care pune propria sa adresă *MAC* în locul aceleia a sistemului considerat de încredere.

În mediile Erthernet actuale, tabela *CAM* de switch-uri păstrează istoricul adreselor *MAC*, al *VLAN*-urilor și la ce port este conectată fiecare adresă *MAC*. Când un intrus schimbă o adresă *MAC* cu adresa altui sistem deja conectat la switch, tabela *CAM* se actualizează. Acest lucru apare clar imediat ce atacatorul sistemului trimite un frame pe canal. Tot traficul destinat acestei adrese *MAC* (și adresa *IP* deservită de ea) este returnată atacatorului până când sistemul real comunică din nou.

MAC Spoofing (sau Layer 2 Spoofing) este un atac simplu care pune propria sa adresă *MAC* în locul aceleia a sistemului considerat de încredere.

În mediile Erthernet actuale, tabela *CAM* de switch-uri păstrează istoricul adreselor *MAC*, al *VLAN*-urilor și la ce port este conectată fiecare adresă *MAC*. Când un intrus schimbă o adresă *MAC* cu adresa altui sistem deja conectat la switch, tabela *CAM* se actualizează. Acest lucru apare clar imediat ce atacatorul sistemului trimite un frame pe canal. Tot traficul destinat acestei adrese *MAC* (și adresa *IP* deservită de ea) este returnată atacatorului până când sistemul real comunică din nou.

Atacul lucrează bine pe sisteme care doar primesc date (de exemplu servere *Syslog*). Nu este un atac deosebit de periculos și stoparea lui este rezonabilă numai în cazul sistemelor critice, unde o linie *CAM* poate fi configurată astfel ca o adresă *MAC* dată să fie asociată întotdeauna unui anumit port.

IP Spoofing

Caracteristici:

1. **Membru al clasei/subclasei**: Spoof.
2. **Exemplu de implementări**: Orice atac capabil să acceseze sistemul de control al driverelor unui sistem.
3. **Preliminarii**: Niciunul.
4. **Vulnerabilități**: Niciuna.
5. **Utilizare tipică**: Ascunde sursa unui atac de nivel superior.
6. **Rezultate atac**: Creșterea nivelului de acces.
7. **Posibile atacuri ulterioare**: Variabilă.
8. *OSI* **Layers**: 3
9. **Detectare**: *IDS*.
10. **Protecție**: RFC 2827, RFC 1918, verificare *RPF* (pe router sau firewall), criptografie.

Un header *IP* are 20 octeți; niciunul din câmpurile sale nu este protejat pentru un atac spoofing.
Dacă intrusul poate accesa sistemul de control al driverelor (care de obicei este admis doar pentru administrator sau pentru *root*), el poate trimite un pachet cu orice header *IP*.

Un header *IP* are 20 octeți; niciunul din câmpurile sale nu este protejat pentru un atac spoofing.
Dacă intrusul poate accesa sistemul de control al driverelor (care de obicei este admis doar pentru administrator sau pentru *root*), el poate trimite un pachet cu orice header *IP*.
Ca și atacul Spoofing precedent, IP Spoofing este cauzat în principal de unele vulnerabilități software.
Factorul de impact este relativ redus dacă atacul este izolat.
Totuși, dacă el este o componentă a unui atac mai complicat, poate deveni extrem de periculos.

Un header *IP* are 20 octeți; niciunul din câmpurile sale nu este protejat pentru un atac spoofing.
Dacă intrusul poate accesa sistemul de control al driverelor (care de obicei este admis doar pentru administrator sau pentru *root*), el poate trimite un pachet cu orice header *IP*.
Ca și atacul Spoofing precedent, IP Spoofing este cauzat în principal de unele vulnerabilități software.
Factorul de impact este relativ redus dacă atacul este izolat.
Totuși, dacă el este o componentă a unui atac mai complicat, poate deveni extrem de periculos.
Criptografia este folosită ca mecanism de protecție numai atunci când se aplică unui sistem care solicită comunicații criptate pentru accesul la informațiile *IP*.
De exemplu, într-o aplicație financiară, cumpărătorul trebuie să afle *IP*-ul unui vânzător fără a-și dezvălui identitatea.

Transport Spoofing

Această subclasă de atacuri se referă la reușita unui spoofing pe nivelul de transport (Layer 4).
Sunt cuprinse aici două mari tipuri de atac: *UDP* Spoofing și *TCP* Spoofing.

UDP Spoofing

Caracteristici:

1. **Membru al clasei/subclasei**: Spoof/Transport Spoofing.
2. **Exemplu de implementări**: Orice atac capabil să acceseze sistemul de control al driverelor unui sistem.
3. **Preliminarii**: *IP* Spoofing.
4. **Vulnerabilități**: Niciuna.
5. **Utilizare tipică**: Introduce date neautorizate într-o aplicație care utilizează *UDP* ca mijloc de transport.
6. **Rezultate atac**: Coruperea sau dezvăluirea informației.
7. **Posibile atacuri ulterioare**: Variabilă.
8. ***OSI* Layers**: 4
9. **Detectare**: Niciuna (Trebuie stopat *IP* Spoofing).
10. **Protecție**: Se folosesc proprietăți din *TCP* sau se blochează atacul *IP* Spoofing.

Un Header *UDP* este mult mai simplu decât un header *IP*. El conține numai 8 octeți împărțiți în 4 câmpuri: numerele celor două porturi (sursă și destinație), lungimea câmpului de date, și o sumă de control (opțională la unele sisteme mai vechi). Nu există nici o noțiune de conectivitate. De aceea un *UDP* este mai simplu de atacat.

Un Header *UDP* este mult mai simplu decât un header *IP*. El conține numai 8 octeți împărțiți în 4 câmpuri: numerele celor două porturi (sursă și destinație), lungimea câmpului de date, și o sumă de control (opțională la unele sisteme mai vechi). Nu există nici o noțiune de conectivitate. De aceea un *UDP* este mai simplu de atacat.
Cum managementul multor aplicații cum ar fi *SNMP* (Network Management Protocol), *Syslog*, *TFTP* (Trivial File Transfer Protocol) folosește *UDP* ca mecanism de transport, securitatea administrării canalelor unei rețele este considerată ca fiind veriga slabă a securității wireless.

TCP Spoofing

Caracteristici:

1. **Membru al clasei/subclasei**: Spoof/Transport Spoofing.
2. **Exemplu de implementări**: Orice atac capabil să acceseze sistemul de control al driverelor unui sistem.
3. **Preliminarii**: *IP* Spoofing.
4. **Vulnerabilități**: Niciuna.
5. **Utilizare tipică**: Introduce date neautorizate într-o aplicație care utilizează *TCP* ca mijloc de transport.
6. **Rezultate atac**: Coruperea sau dezvăluirea informației.
7. **Posibile atacuri ulterioare**: Variabilă.
8. ***OSI* Layers**: 4
9. **Detectare**: Niciuna (Trebuie stopat *IP* Spoofing).
10. **Protecție**: Se blochează atacul *IP* Spoofing.

Structura unui header *TCP* este mult mai complexă decât a unui header *UDP*.
În particular, acesta conține un nonce – un număr aleator pe 32 biți – atașat conexiunii. Fără un acces direct la șirul de date printr-un atac sniffing, este imposibil de ghicit valoarea lui. Pentru a insera o comunicare validă în fluxul de date, atacatorul trebuie să știe această valoare și – simultan – să întrerupă legătura dintre client și server.

Structura unui header *TCP* este mult mai complexă decât a unui header *UDP*.
În particular, acesta conține un nonce – un număr aleator pe 32 biți – atașat conexiunii. Fără un acces direct la șirul de date printr-un atac sniffing, este imposibil de ghicit valoarea lui. Pentru a insera o comunicare validă în fluxul de date, atacatorul trebuie să știe această valoare și – simultan – să întrerupă legătura dintre client și server.
Dacă intrusul este exterior, atacul are foarte puține șanse de reușită.
El devine însă mult mai periculos dacă este lansat din interior, dintr-o locație aflată pe drumul dintre client și server. Fiind conectat la server, atacatorul își poate crea acces la toată informația necesară pentru lansarea atacului.

Identity Spoofing

Caracteristici:

1. **Membru al clasei/subclasei**: Spoof.
2. **Exemplu de implementări**: *LC4*; Jack Spintecătorul.
3. **Preliminarii**: *IP* Variabile.
4. **Vulnerabilități**: Utilizare.
5. **Utilizare tipică**: Convinge o rețea că este utilizator legal.
6. **Rezultate atac**: Creșterea nivelului de acces.
7. **Posibile atacuri ulterioare**: Citire/Manipulare.
8. *OSI* **Layers**: 7
9. **Detectare**: Niciuna.
10. **Protecție**: Aplicații de securitate (pentru detectare și protecție).

Prin Identity Spoofing se înțeleg diverse tipuri de atac, cum ar fi spargerea de parole, încercări de logare brute-force, furt de certificate digitale etc. Ținta o constituie mecanismele de identificare, aflate pe un nivel superior.
Dacă intrusul reușește să le compromită, el poate coborî ușor – prin intermediul mai multor biți de informație partajați – pe nivele mai joase, de aplicații.

Prin Identity Spoofing se înțeleg diverse tipuri de atac, cum ar fi spargerea de parole, încercări de logare brute-force, furt de certificate digitale etc. Ținta o constituie mecanismele de identificare, aflate pe un nivel superior.
Dacă intrusul reușește să le compromită, el poate coborî ușor – prin intermediul mai multor biți de informație partajați – pe nivele mai joase, de aplicații.
O listă a mecanismelor de identificare este (în ordinea crescătoare a securității):

1. Username și parolă în clar (de exemplu *Telnet*);
2. Chei prestabilite (de exemplu *WEP* – Wired Equivalent Privacy);
3. Parole One-Time;
4. Criptografia cu chei publice (de exemplu *PGP*, *IPSec*).

Jack Spintecătorul și *LC4* sunt atacuri de spargere de parole, care – în linii mari – încearcă să ghicească o parolă, apoi o criptează și o compară cu versiunea criptată a parolei victimei stocată pe server. Majoritatea parolelor sunt stocate sub formă de amprentă printr-o funcție de dispersie criptografică.

Jack Spintecătorul și *LC4* sunt atacuri de spargere de parole, care – în linii mari – încearcă să ghicească o parolă, apoi o criptează și o compară cu versiunea criptată a parolei victimei stocată pe server. Majoritatea parolelor sunt stocate sub formă de amprentă printr-o funcție de dispersie criptografică.

Identity spoofing este unul din cele mai periculoase atacuri, în primul rând deoarece administratorul de sistem nu poate obliga utilizatorii să aleagă parole dificile.
Cât timp un utilizator trebuie să țină minte o parolă, el nu va alege una dificilă, ci una legată intrisec de cunoștințele sale. O soluție alternativă ar fi identificările biometrice (cu alte slăbiciuni însă).

Aparate false (Rogue Devices)

Caracteristici:

1. **Membru al clasei/subclasei**: Spoof.
2. **Exemplu de implementări**: Orice device legitim de rețea. De exemplu *WLAN AP*, *DHCP*, server, router, host.
3. **Preliminarii**: Acces fizic la device.
4. **Vulnerabilități**: Utilizare sau control fizic al securității.
5. **Utilizare tipică**: Oferă servicii comunității; datele furate din cererile lor sunt trecute apoi spre rețeaua legitimă.
6. **Rezultate atac**: Dezvăluirea și coruperea informației.
7. **Posibile atacuri ulterioare**: Citire/Manipulare.
8. *OSI* **Layers**: Toate.
9. **Detectare**: Variază după tipul device-ului.
10. **Protecție**: Variază după tipul device-ului.

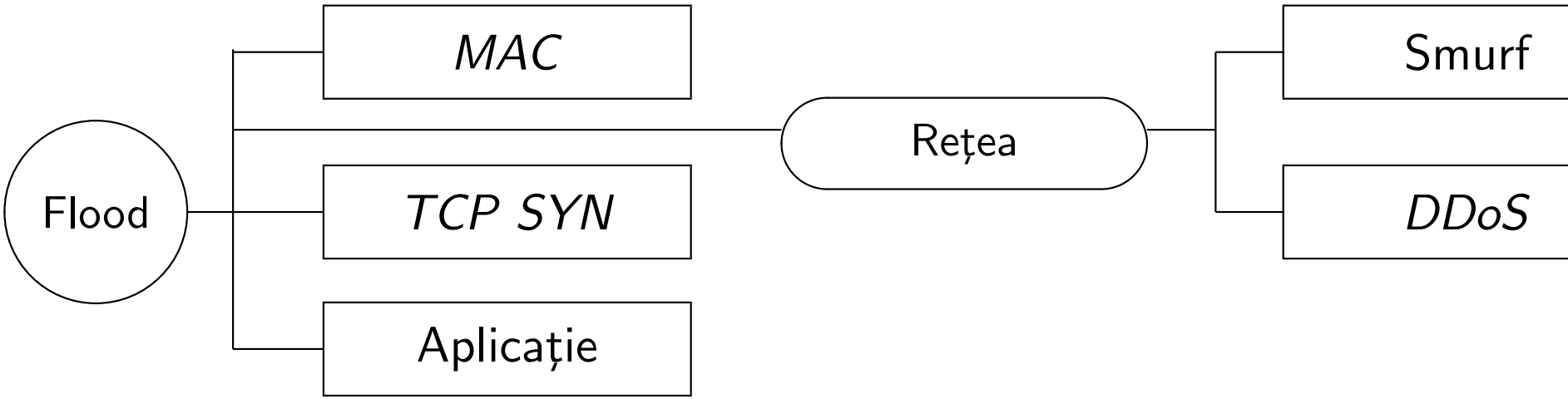
Spre deosebire de atacurile spoofing precedente – bazate exclusiv pe slăbiciuni software – acest atac introduce în rețea un device propriu (fals), sperând să convingă utilizatorii și device-urile din rețea că este legitim.
Cele mai comune astfel de atacuri sunt bancomatele false și cardurile de acces (bancare, telefonice etc).

Spre deosebire de atacurile spoofing precedente – bazate exclusiv pe slăbiciuni software – acest atac introduce în rețea un device propriu (fals), sperând să convingă utilizatorii și device-urile din rețea că este legitim.
Cele mai comune astfel de atacuri sunt bancomatele false și cardurile de acces (bancare, telefonice etc).
După ce acest aparat fals este introdus de intrus în rețea, el încearcă să determine adresa *IP* și prezența unui server proxy *HTTP*, după care crează o conexiune tunelată spre atacator. În continuare sunt posibile alte diverse atacuri, cum ar fi Redirecționare *ARP* sau *MAC* Flooding.

Spre deosebire de atacurile spoofing precedente – bazate exclusiv pe slăbiciuni software – acest atac introduce în rețea un device propriu (fals), sperând să convingă utilizatorii și device-urile din rețea că este legitim.
Cele mai comune astfel de atacuri sunt bancomatele false și cardurile de acces (bancare, telefonice etc).
După ce acest aparat fals este introdus de intrus în rețea, el încearcă să determine adresa *IP* și prezența unui server proxy *HTTP*, după care crează o conexiune tunelată spre atacator. În continuare sunt posibile alte diverse atacuri, cum ar fi Redirecționare *ARP* sau *MAC* Flooding.
Un atac cu aparat fals poate fi extrem de periculos, dar montarea aparatului este dificilă, deoarece intrusul trebuie să aibă acces fizic la rețeaua pe dorește să o compromită.

Clasa "Flood"

Un atac de tip Flood apare atunci când un atacator trimite un număr foarte mare de date care blochează resursele rețelei.



Sunt atacuri diverse, extrem de frecvente, cu grade diferite de periculozitate.
Scopul lor nu este aflarea sau coruperea informației, ci blocarea ei și a proceselor rețelei atacate.

MAC Flooding

Caracteristici:

1. **Membru al clasei/subclasei**: Flood.
2. **Exemplu de implementări**: *macof* .
3. **Preliminarii**: Acces *LAN* local.
4. **Vulnerabilități**: Politica de securitate.
5. **Utilizare tipică**: Umple o tabelă *CAM* și apoi respinge traficul legitim.
6. **Rezultate atac**: Dezvăluirea informației.
7. **Posibile atacuri ulterioare**: Citire/Manipulare.
8. *OSI* **Layers**: 2
9. **Detectare**: Schimbarea monitorizării (mărimea tabelei *CAM*).
10. **Protecție**: Securizarea portului.

Un atac *MAC flooding* trimite pachete cu sursă ascunsă și destinație adresele *MAC*, de la sistemul atacatorului spre rețeaua Ethernet.
Tabela *CAM* care păstrează urma adreselor *MAC* are mărime limitată. După ce ea s-a umplut, frame-urile destinate adreselor *MAC* care nu au o intrare în *CAM* sunt revărsate în rețeaua *VLAN* locală pentru a fi livrate destinatarului. În acest fel pachetele de informații sunt prelucrate similar unor informații partajate și pot fi accesate de intrus.

Umplere rețea

Atacurile de umplere a rețelei (Network flooding) au ca scop consumarea benzii de transmisie a unei rețele.
După aceea, șansele ca traficul legitim să circule între utilizatori va fi extrem de redus.

Umplere rețea

Atacurile de umplere a rețelei (Network flooding) au ca scop consumarea benzii de transmisie a unei rețele.
După aceea, șansele ca traficul legitim să circule între utilizatori va fi extrem de redus.

Atacurile sunt îndreptate în special asupra conexiunilor rețelelor Internet, caracterizate deja prin viteză mică de procesare.
Vom analiza două astfel de atacuri (cele mai frecvente): Smurf și DDoS.

Smurf

Atacul **smurf** (numit după personajele de desene animate) folosește ping-urile broadcast mascate prin Internet Control Message Protocol (*ICMP*).

Smurf

Atacul **smurf** (numit după personajele de desene animate) folosește ping-urile broadcast mascate prin Internet Control Message Protocol (*ICMP*).
Un *IP* conține informații utile pentru un *direct broadcast*. Un **direct broadcast** apare când o stație a unei rețele trimite un pachet broadcast spre altă rețea.
De exemplu, o stație din rețeaua 192.0.2.0/24 trimite un pachet spre 192.0.3.255. Dacă routerul este configurat să propage direct broadcast, rețeaua 192.0.3.0/24 primește acest pachet și-l retrimite tuturor stațiilor pe care le controlează. Acestea vor răspunde toate stației sursă.

Smurf

Atacul **smurf** (numit după personajele de desene animate) folosește ping-urile broadcast mascate prin Internet Control Message Protocol (*ICMP*).
Un *IP* conține informații utile pentru un *direct broadcast*. Un **direct broadcast** apare când o stație a unei rețele trimite un pachet broadcast spre altă rețea.
De exemplu, o stație din rețeaua 192.0.2.0/24 trimite un pachet spre 192.0.3.255. Dacă routerul este configurat să propage direct broadcast, rețeaua 192.0.3.0/24 primește acest pachet și-l retrimite tuturor stațiilor pe care le controlează. Acestea vor răspunde toate stației sursă.
Atacul *smurf* folosește acest comportament pentru a multiplica un pachet mic (*smurf*) de date într-unul mare, capabil să blocheze o stație.

Caracteristici:

1. **Membru al clasei/subclasei**: Flood/Umplere rețea.
2. **Exemplu de implementări**: Aproape orice program *ping*.
3. **Preliminarii**: Acces la rețea, abilitate de a "fura" o adresă a victimei.
4. **Vulnerabilități**: Politica de securitate.
5. **Utilizare tipică**: Umple o conexiune a unui site Internet cu *ICMP echo* dat de traficul de răspuns.
6. **Rezultate atac**: *DDoS*.
7. **Posibile atacuri ulterioare**: Niciunul.
8. *OSI* **Layers**: 3
9. **Detectare**: *IDS*, log analysis.
10. **Protecție**: Implementarea comenzii **no ip directed-broadcast**; Alegerea unor stive gazdă pentru *IP* care să nu răspundă la ping-urile broadcast; *CAR* (Commited access rate).

DDoS

Caracteristici:

1. **Membru al clasei/subclasei**: Flood/Umplere rețea.
2. **Exemplu de implementări**: *Tribe Flood Network 2000*, *Shaft*.
3. **Preliminarii**: Abilitatea de a infecta un număr mare de sisteme cu care se construiește o rețea de atacatori zombie.
4. **Vulnerabilități**: Politica de securitate.
5. **Utilizare tipică**: Sufocă conexiunea victimei cu o cantitate mare de mesaje Internet.
6. **Rezultate atac**: Respingerea serviciului Internet.
7. **Posibile atacuri ulterioare**: Niciunul
8. *OSI* **Layers**: $3 - 4$
9. **Detectare**: *IDS*, log analysis.
10. **Protecție**: *CAR*, filtre specifice, opțiuni *ISP*.

Au cea mai mare notorietate, mai ales datorită atacurilor recente asupra site-urilor guvernamentale din diverse țări.

Au cea mai mare notorietate, mai ales datorită atacurilor recente asupra site-urilor guvernamentale din diverse țări.
Forma standard de desfășurare (atacul Stachelraht):

1. Atacatorul infectează un număr de servere de pe Internet și injectează în ele un virus *DDosS*.

Au cea mai mare notorietate, mai ales datorită atacurilor recente asupra site-urilor guvernamentale din diverse țări.
Forma standard de desfășurare (atacul Stachelraht):

1. Atacatorul infectează un număr de servere de pe Internet și injectează în ele un virus *DDosS*.
2. Serverele infectate subjugă sistemele aflate sub control transformându-le în agenți proprii (*boți*). Atacul folosit poate fi de orice fel (de exemplu un *troian* prin e-mail) exploatând o vulnerabilitate de aplicație sau a sistemului de operare.

Au cea mai mare notorietate, mai ales datorită atacurilor recente asupra site-urilor guvernamentale din diverse țări.
Forma standard de desfășurare (atacul Stachelraht):

1. Atacatorul infectează un număr de servere de pe Internet și injectează în ele un virus *DDosS*.
2. Serverele infectate subjugă sistemele aflate sub control transformându-le în agenți proprii (*boți*). Atacul folosit poate fi de orice fel (de exemplu un *troian* prin e-mail) exploatând o vulnerabilitate de aplicație sau a sistemului de operare.
3. La un moment, atacatorul trimite ordinul de atac serverelor infectate, cu data atacului și adresa *IP* a victimei.

Au cea mai mare notorietate, mai ales datorită atacurilor recente asupra site-urilor guvernamentale din diverse țări.
Forma standard de desfășurare (atacul Stachelraht):

1. Atacatorul infectează un număr de servere de pe Internet și injectează în ele un virus *DDosS*.
2. Serverele infectate subjugă sistemele aflate sub control transformându-le în agenți proprii (*boți*). Atacul folosit poate fi de orice fel (de exemplu un *troian* prin e-mail) exploatând o vulnerabilitate de aplicație sau a sistemului de operare.
3. La un moment, atacatorul trimite ordinul de atac serverelor infectate, cu data atacului și adresa *IP* a victimei.
4. Toate sistemele infectate lansează simultan o avalanșă de mesaje de date spre calculatorul vizat, care este sufocat de trafic.

TCP SYN Flooding

Caracteristici:

1. **Membru al clasei/subclasei**: Flood.
2. **Exemplu de implementări**: *Apsend, Spastic*.
3. **Preliminarii**: Acces direct.
4. **Vulnerabilități**: Software.
5. **Utilizare tipică**: Sufocă o gazdă anumită cu cereri de conectare.
6. **Rezultate atac**: *DoS*.
7. **Posibile atacuri ulterioare**: de tip Spoof și Device-uri false.
8. *OSI* **Layers**: 4
9. **Detectare**: *IDS*, log analysis, aplicații de securitate.
10. **Protecție**: Cookie *TCP SYN*, interceptoare *TCP*.

TCP SYN este unul din primele forme de atac flooding. Atacatorul trimite un pachet *TCP SYN*, după care nu răspunde deloc la pachetul $SYN - ACK$ trimis ca răspuns. Serverul accesat păstrează cererea și retrimite de câteva ori pachetul $SYN - ACK$ înainte de a închide încercarea (falsă) de stabilire a sesiunii de comunicare.

TCP SYN este unul din primele forme de atac flooding. Atacatorul trimite un pachet *TCP SYN*, după care nu răspunde deloc la pachetul $SYN - ACK$ trimis ca răspuns. Serverul accesat păstrează cererea și retrimite de câteva ori pachetul $SYN - ACK$ înainte de a închide încercarea (falsă) de stabilire a sesiunii de comunicare. Când atacatorii lansează un atac *TCP SYN*, ei trimit sistemului mii de cereri de conectare. În încercarea acestuia de a răspunde la toate aceste cereri, sistemul își consumă toată memoria și cedează. În primii ani, acest atac era ușor de realizat deoarece mărimile cozilor de conectare ale sistemelor erau mici. Astăzi sistemele rezistă mult mai bine acestor atacuri, datorită atât îmbunătățirii aplicațiilor și sistemelor de operare, cât și a dezvoltării de tehnologii.

TCP SYN este unul din primele forme de atac flooding. Atacatorul trimite un pachet *TCP SYN*, după care nu răspunde deloc la pachetul $SYN - ACK$ trimis ca răspuns. Serverul accesat păstrează cererea și retrimite de câteva ori pachetul $SYN - ACK$ înainte de a închide încercarea (falsă) de stabilire a sesiunii de comunicare.
Când atacatorii lansează un atac *TCP SYN*, ei trimit sistemului mii de cereri de conectare. În încercarea acestuia de a răspunde la toate aceste cereri, sistemul își consumă toată memoria și cedează.
În primii ani, acest atac era ușor de realizat deoarece mărimile cozilor de conectare ale sistemelor erau mici. Astăzi sistemele rezistă mult mai bine acestor atacuri, datorită atât îmbunătățirii aplicațiilor și sistemelor de operare, cât și a dezvoltării de tehnologii.
Un atac de acest tip nu este posibil asupra *UDP*, deoarece *UDP* nu are nici o legătură cu operația de conectare. Asupra *UDP*-ului sunt mult mai puternice atacurile *DDoS*.

Flooding pe aplicație

Flooding-urile pe aplicație se referă la paleta de atacuri desemnate să scoată din circuit o aplicație sau un sistem prin consumarea resurselor sale. Exemplul tipic este Spamul. Deși spamul nu este abilitat pentru consumul de resurse, el are totuși acest efect asupra sistemului de poștă electronică.

Flooding pe aplicație

Flooding-urile pe aplicație se referă la paleta de atacuri desemnate să scoată din circuit o aplicație sau un sistem prin consumarea resurselor sale. Exemplul tipic este Spamul. Deși spamul nu este abilitat pentru consumul de resurse, el are totuși acest efect asupra sistemului de poștă electronică.
Alt tip de atac se referă la o rulare intensivă continuă a *CPU* sau un flooding pe server cu cereri de autentificare nefinalizate (atacatorul inițiază o conexiune, apoi nu mai răspunde din momentul stabilirii parolei); acesta seamănă cu atacul *SYN* flood, singura diferență fiind layerul de aplicație.

Flooding pe aplicație

Flooding-urile pe aplicație se referă la paleta de atacuri desemnate să scoată din circuit o aplicație sau un sistem prin consumarea resurselor sale. Exemplul tipic este Spamul. Deși spamul nu este abilitat pentru consumul de resurse, el are totuși acest efect asupra sistemului de poștă electronică.

Alt tip de atac se referă la o rulare intensivă continuă a *CPU* sau un flooding pe server cu cereri de autentificare nefinalizate (atacatorul inițiază o conexiune, apoi nu mai răspunde din momentul stabilirii parolei); acesta seamănă cu atacul *SYN* flood, singura diferență fiind layerul de aplicație.

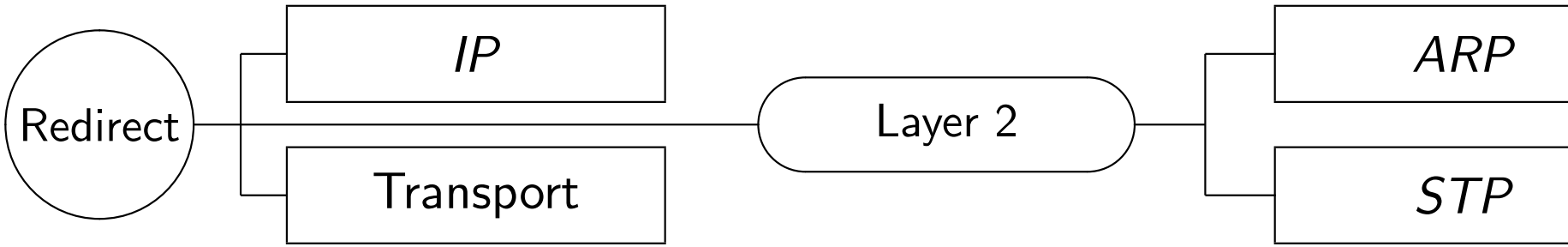
Alt atac interesant (numit și Flash crowds) este bazat pe efectul Slashdot. Când este dată o știre interesantă, de obicei sunt descrise doar câteva idei, după care se face trimitere la un link sau o locație *www* unde există mai multă informație.

Caracteristici:

1. **Membru al clasei/subclasei**: Flood.
2. **Exemplu de implementări**: Spam, Flooding pe autentificare, Abuz de procesare *CPU*.
3. **Preliminarii**: Acces direct.
4. **Vulnerabilități**: Oricare.
5. **Utilizare tipică**: Face inutilizabilă o aplicație sau sistem.
6. **Rezultate atac**: *DoS*.
7. **Posibile atacuri ulterioare**: de tip Spoof și Device-uri false.
8. *OSI* **Layers**: 7
9. **Detectare**: *IDS*, log analysis, aplicații de securitate.
10. **Protecție**: Aplicație de securitate.

Clasa "Redirectare"

Într-un atac *Redirectare* adversarul încearcă schimbarea fluxului de informație dintr-o rețea.
Aceasta poate apare la orice nivel, dar din perspectiva ideii de securitate pe rețea sunt importante numai redirectarea pentru *Layer* 2, *IP* și transport.



Redirectare *L2*

O redirectare pe Nivelul 2 poate fi realizată folosind:

- *ARP*;
- Spanning Tree Protocol (*STP*).

ARP Redirectare/Spoofing

Caracteristici:

1. **Membru al clasei/subclasei**: Redirectare/ Layer 2.
2. **Exemplu de implementări**: *arpsoof (part* sau *dsniff*).
3. **Preliminarii**: Acces direct (conectivitate *LAN* locală),
4. **Vulnerabilități**: Niciuna.
5. **Utilizare tipică**: Redirecționează traficul de ieșire din rețea prin sistemul atacatorului în loc de gateway-ul ales ca default.
6. **Rezultate atac**: Dezvăluirea informației.
7. **Posibile atacuri ulterioare**: Manipulare/Citire.
8. *OSI* **Layers**: 2
9. **Detectare**: *IDS*, *arpwatch*.
10. **Protecție**: Inspecții *ARP* și *ARP* static.

Atacul este cunoscut și sub numele *ARP* Spoofing.
Dar prima sa funcție este redirecționarea traficului; spoofing-ul este numai un mecanism utilizat pentru atac.

Atacul este cunoscut și sub numele *ARP* Spoofing.
Dar prima sa funcție este redirecționarea traficului; spoofing-ul este numai un mecanism utilizat pentru atac.
Atacul funcționează astfel: adversarul trimite o avalanșă de broadcast-uri *ARP* afirmând că adresa *MAC* a gateway-ului ales ca default s-a schimbat cu adresa *MAC* a atacatorului.
După ce mașina victimei updatează cache-ul *ARP*, toate cererile de ieșire ale sistemului victimei sunt îndreptate spre sistemul atacatorului, unde apoi mesajele pot fi citite, modificate sau șterse.

Redirectare *STP*

Atacurile *STP* pot fi utilizate ca o altă modalitate de redirecționare a traficului la Nivelul 2.

Atacatorul este capabil să convingă acest nivel al rețelei că el este o rădăcină de bridge *STP*.
Efectul este o modificare a topologiei nivelului 2 într-o structură avantajoasă intrusului.

Caracteristici:

1. **Membru al clasei/subclasei**: Redirectare/ Layer 2.
2. **Exemplu de implementări**: Orice device capabil să genereze mesaje *STP* (switch, host UNIX etc)
3. **Preliminarii**: Acces direct (conectivitate *LAN* locală),
4. **Vulnerabilități**: Politica de securitate.
5. **Utilizare tipică**: Schimbă drumul prin nivelul 2 al rețelei prin includerea sistemului atacator ca punct de comutare (switch).
6. **Rezultate atac**: Dezvăluirea informației.
7. **Posibile atacuri ulterioare**: Manipulare/Citire.
8. *OSI* **Layers**: 2
9. **Detectare**: Tooluri de management a rețelei.
10. **Protecție**: Control la *STP* root și/sau *BTPU*.

Redirectare *IP*

Caracteristici:

1. **Membru al clasei/subclasei**: Redirectare.
2. **Exemplu de implementări**: Orice device capabil să ruleze protocoale de routare.
3. **Preliminarii**: Acces direct.
4. **Vulnerabilități**: Configurare/Execuție.
5. **Utilizare tipică**: Introduce drumuri de routare preferențiale sau modifică configurația de router pentru a transfera traficul prin sistemul atacatorului.
6. **Rezultate atac**: Dezvăluirea informației.
7. **Posibile atacuri ulterioare**: Manipulare/Citire.
8. *OSI* **Layers**: 3
9. **Detectare**: Tooluri de management a rețelei.

Un atac prin redirectare *IP* poate schimba fluxul de informații în două moduri:

- Introducând un router fals cu advertizări nelegitime;

Un atac prin redirectare *IP* poate schimba fluxul de informații în două moduri:

- Introducând un router fals cu advertizări nelegitime;
- Reconfigurând routerele existente.

Un atac prin redirectare *IP* poate schimba fluxul de informații în două moduri:

- Introducând un router fals cu advertizări nelegitime;
- Reconfigurând routerele existente.

Traficul va trece acum prin sistemul atacatorului care are astfel acces direct la el și poate citi, modifica sau șterge mesajele.
O securizare atentă a protocolului de routare și a comenzilor de control pe routere poate reduce drastic rata de succes ale unor astfel de atacuri.

Redirectare transport

Caracteristici:

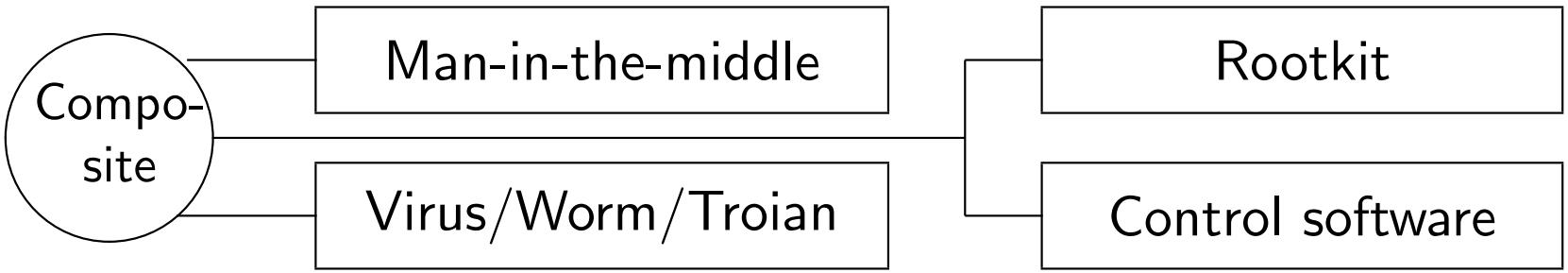
1. **Membru al clasei/subclasei**: Redirectare.
2. **Exemplu de implementări**: *Netcat*.
3. **Preliminarii**: Variabile.
4. **Vulnerabilități**: Variabile.
5. **Utilizare tipică**: Redirectează cererile la un număr de port și adresele *IP* la alt număr de port.
6. **Rezultate atac**: Creșterea accesulu.i
7. **Posibile atacuri ulterioare**: Manipulare/Citire.
8. *OSI* **Layers**: 4
9. **Detectare**: *IDS*
10. **Protecție**: Aplicații de securitate.

Redirectarea transportului – numit și *Redirectarea portului* – este un atac prin care intrusul poate intercepta traficul care este deviat pe alt drum.

Redirectarea transportului – numit și *Redirectarea portului* – este un atac prin care intrusul poate intercepta traficul care este deviat pe alt drum.
Atacatorul este capabil să seteze în sistemul compromis un soft care va redirecționa cererile de la un sistem și port (legitim) la alt sistem și alt port (ilegitim).
Cel mai bun exemplu de funcționare este *Netcat*.

Clasa Composite

Atacurile descrise folosesc o combinație din mai multe tipuri de atac. O structură generală a clasei este:



Man-in-the-middle

Caracteristici:

1. **Membru al clasei/subclasei**: Composite.
2. **Exemple de implementări**: *dsniff*, *Ettercap*.
3. **Preliminarii**: Variabile.
4. **Vulnerabilități**: Variabile.
5. **Utilizare tipică**: Urmărește traficul și atacă sesiunile.
6. **Rezultate atac**: Variabile (orice este posibil).
7. **Tipuri de atac folosite**: Citire, Manipulare, Spoof, Redirect.
8. *OSI* **Layers**: 7
9. **Detectare**: Variabilă.
10. **Protecție**: Criptografie.

Într-un atac *MITM* pe rețele wireless, atacatorul are control activ asupra celui mai înalt nivel relevant de conversație dintre două victime.
Acesta este de obicei Nivelul 7, dar uneori el poate apare și la nivelul 3 (când se utilizează *IPSec*).

Într-un atac *MITM* pe rețele wireless, atacatorul are control activ asupra celui mai înalt nivel relevant de conversație dintre două victime.
Acesta este de obicei Nivelul 7, dar uneori el poate apare și la nivelul 3 (când se utilizează *IPSec*).

Exemplu

Într-o comunicare între un client și bancă, clientul crede că este în legătură cu banca, iar banca – cu clientul. În realitate ambele conversații sunt routate prin sistemul atacatorului, care poate modifica datele personale, sumele transferate etc.

Într-un atac *MITM* pe rețele wireless, atacatorul are control activ asupra celui mai înalt nivel relevant de conversație dintre două victime.
Acesta este de obicei Nivelul 7, dar uneori el poate apare și la nivelul 3 (când se utilizează *IPSec*).

Exemplu

Într-o comunicare între un client și bancă, clientul crede că este în legătură cu banca, iar banca – cu clientul. În realitate ambele conversații sunt routate prin sistemul atacatorului, care poate modifica datele personale, sumele transferate etc.

Un sistem de criptare bine implementat constituie cel mai simplu mod de apărare contra unui atac *MITM*. Vom da ca exemplu două studii de caz: *dsniff* și *Ettercap*.

dsniff

Sub acest nume sunt adunate o serie de tool-uri construite de Dug Song. Fiecare element poate fi utilizat pentru a realiza propriul său atac.

dsniff

Sub acest nume sunt adunate o serie de tool-uri construite de Dug Song. Fiecare element poate fi utilizat pentru a realiza propriul său atac.
Astfel, *macof* poate lansa *MAC flooding*, *arpspoof* poate lansa *ARP Redirection* și *spoofing*.
Tool-ul *dsniff* poate activa ca un sniffer selectiv și afla username și parola unui utilizator. Lucrând împreună, toolurile din *dsniff* permit intrusului să lanseze un atac *MITM* complet.

Sunt posibile diverse tipuri de atac *MITM*. De exemplu, cu *arpspoof* , *dnsspoof* și *webmitm* se poate construi următorul atac:

1. Atacatorul începe prin a rula *arpspoof* pentru a redirecționa prin propria sa mașină traficul îndreptat spre principalul gateway.

Sunt posibile diverse tipuri de atac *MITM*. De exemplu, cu *arpspoof* , *dnsspoof* și *webmitm* se poate construi următorul atac:

1. Atacatorul începe prin a rula *arpspoof* pentru a redirecționa prin propria sa mașină traficul îndreptat spre principalul gateway.
2. După ce acest trafic traversează mașina atacatorului, *dnsspoof* returnează adresa *IP* a atacatorului cu datele *DNS* specifice hosturilor din spatele sistemului atacator.
   Deoarece un browser web arată un nume și nu o adresă *IP*, victima nu va ști că cererile sale sunt routate prin sistemul intrusului și nu spre adevărata destinație.

Sunt posibile diverse tipuri de atac *MITM*. De exemplu, cu *arpspoof* , *dnsspoof* și *webmitm* se poate construi următorul atac:

1. Atacatorul începe prin a rula *arpspoof* pentru a redirecționa prin propria sa mașină traficul îndreptat spre principalul gateway.
2. După ce acest trafic traversează mașina atacatorului, *dnsspoof* returnează adresa *IP* a atacatorului cu datele *DNS* specifice hosturilor din spatele sistemului atacator.
   Deoarece un browser web arată un nume și nu o adresă *IP*, victima nu va ști că cererile sale sunt routate prin sistemul intrusului și nu spre adevărata destinație.
3. La sosirea unei cereri web este lansat *webmitm*; acesta generează un certificat digital auto-semnat pe care îl prezintă victimei atunci când la un server *SSL* vine o cerere de conectare. Dacă victima nu observă certificatul fals, atunci atacatorul este capabil să citească toate pachetele și eventual să le modifice înainte de a le retrimite serverului real.

Ethercap

Este un tool similar cu *dsniff*. Principalele diferențe:

- Abilitatea atacatorului de a face *ARP spoof* la ambele capete ale canalului de comunicare, asigurând trecerea prin propria mașină atât a traficului trimis cât și a celui primit.

Ethercap

Este un tool similar cu *dsniff*. Principalele diferențe:

- Abilitatea atacatorului de a face *ARP spoof* la ambele capete ale canalului de comunicare, asigurând trecerea prin propria mașină atât a traficului trimis cât și a celui primit.
- Inserarea de comenzi în timp real în sesiuni *TCP*, care permite traficului ilegitim să fie trimis sau de la server sau de la client.

Ethercap

Este un tool similar cu *dsniff*. Principalele diferențe:

- Abilitatea atacatorului de a face *ARP spoof* la ambele capete ale canalului de comunicare, asigurând trecerea prin propria mașină atât a traficului trimis cât și a celui primit.
- Inserarea de comenzi în timp real în sesiuni *TCP*, care permite traficului ilegitim să fie trimis sau de la server sau de la client. De exemplu, un atacator care a lansat un atac *MITM* contra unui client care comunică prin Telnet cu un sistem UNIX poate face serverul să gândească că primește de la client comanda **rm-rf*** (comandă UNIX de a șterge toate fișierele), când de fapt acesta nu a lansat așa ceva.
  De asemenea, sesiunea poate fi încheiată de atacator.

Ethercap

Este un tool similar cu *dsniff*. Principalele diferențe:

- Abilitatea atacatorului de a face *ARP spoof* la ambele capete ale canalului de comunicare, asigurând trecerea prin propria mașină atât a traficului trimis cât și a celui primit.
- Inserarea de comenzi în timp real în sesiuni *TCP*, care permite traficului ilegitim să fie trimis sau de la server sau de la client. De exemplu, un atacator care a lansat un atac *MITM* contra unui client care comunică prin Telnet cu un sistem UNIX poate face serverul să gândească că primește de la client comanda **rm-rf*** (comandă UNIX de a șterge toate fișierele), când de fapt acesta nu a lansat așa ceva.
  De asemenea, sesiunea poate fi încheiată de atacator.
- Înlocuiește pachete având anumite secvențe de biți cu noi secvențe alese de atacator.

Viruși, Viermi și Cai Troieni

Un **virus** este o bucată de cod malițios care modifică altă piesă de software din sistem.

Viruși, Viermi și Cai Troieni

Un **virus** este o bucată de cod malițios care modifică altă piesă de software din sistem.
În general este nevoie de o anumită intervenție a utilizatorului (deschiderea unui attachment e-mail, citirea unui disc infectat etc).

Viruși, Viermi și Cai Troieni

Un **virus** este o bucată de cod malițios care modifică altă piesă de software din sistem.
În general este nevoie de o anumită intervenție a utilizatorului (deschiderea unui attachment e-mail, citirea unui disc infectat etc).

Exemplu

Virusul Melissa. Acesta se transmite prin documente word trimise victimei prin attachment. Documentul word conține un macro-cod malițios care face ca virusul să se propage la primele 50 adrese din Adress book.

Un **vierme (worm)** este un tool standard care infectează sistemele vulnerabile, iar acestea infectează la rândul lor alte sisteme.

Un **vierme (worm)** este un tool standard care infectează sistemele vulnerabile, iar acestea infectează la rândul lor alte sisteme.
Un vierme infectează în general în mod automat, deși uneori pentru lansare este nevoie de o acțiune a victimei (cum ar fi un click pe un atachment).

Un **vierme (worm)** este un tool standard care infectează sistemele vulnerabile, iar acestea infectează la rândul lor alte sisteme.
Un vierme infectează în general în mod automat, deși uneori pentru lansare este nevoie de o acțiune a victimei (cum ar fi un click pe un atachment).

Exemplu

Code Red *profită de o falie de securitate a serverului de indexare Microsoft (componentă a IIS – Internet Information Server) pentru a infecta sute de mii de sisteme.*
Din cauza propagării sale necontrolate, acest vierme poate avea și unele efecte de DoS în anumite zone de Internet.

Un **Cal Troian (Troian Horse)** este o aplicație care apare utilizatorului ca având o anumită funcție, dar în realitate acționând complet diferit.

Un **Cal Troian (Troian Horse)** este o aplicație care apare utilizatorului ca având o anumită funcție, dar în realitate acționând complet diferit.

Exemplu

NetBus/Whack-a-Mole. Acesta – distribuit frecvent prin e-mail – apare utilizatorului ca un joc din baza Microsoft Windows (ca și Solitaire de exemplu).
În timp ce utilizatorul joacă acest joc (simpatic), aplicația instalează o listă de comutare pe un port TCP de nivel înalt, permițând atacatorului să se conecteze la sistem și să lanseze diverse atacuri, cum ar fi resetarea sistemului și schimbarea proprietăților locale.

Caracteristici:

1. **Membru al clasei/subclasei**: Composite.
2. **Exemple de implementări**: *SQL Slammer* (vierme), *Code Red* (vierme), *Melissa* (virus), *NetBus/Whack-a-Mole* (Troian).
3. **Preliminarii**: Variabile.
4. **Vulnerabilități**: Software și utilizare.
5. **Utilizare tipică**: Variabilă
6. **Rezultate atac**: Variabilă (orice este posibil).
7. **Tipuri de atac folosite**: Citire, Manipulare, Spoof, Flood.
8. *OSI* **Layers**: 7
9. **Detectare**: *IDS*.
10. **Protecție**: Aplicații de securitate și software antivirus.

Rootkit

Caracteristici:

1. **Membru al clasei/subclasei**: Composite.
2. **Exemple de implementări**: *t0rn*.
3. **Preliminarii**: Acces la root.
4. **Vulnerabilități**: Software, configurare și utilizare.
5. **Utilizare tipică**: Ascunde victimei prezența unui atacator.
6. **Rezultate atac**: Variabilă (orice este posibil).
7. **Tipuri de atac folosite**: Citire, Manipulare.
8. ***OSI* Layers**: $3 - 7$
9. **Detectare**: *Chkrootkit* și *HIDS*.
10. **Protecție**: Aplicații de securitate.

Rootkit permite atacatorilor să-și ascundă prezența pe o mașină pe care deja au compromis-o.

Rootkit permite atacatorilor să-și ascundă prezența pe o mașină pe care deja au compromis-o.
De exemplu, să presupunem că un atacator a compromis un host Linux. Atunci *t0rn* îi va permite să facă următoarele acțiuni:

1. Să dezinstaleze *syslogd*.

Rootkit permite atacatorilor să-și ascundă prezența pe o mașină pe care deja au compromis-o.

De exemplu, să presupunem că un atacator a compromis un host Linux. Atunci *t0rn* îi va permite să facă următoarele acțiuni:

1. Să dezinstaleze *syslogd*.
2. Să-și stocheze parola pentru programe de tip Troian în */etc/ttyhash*.

Rootkit permite atacatorilor să-și ascundă prezența pe o mașină pe care deja au compromis-o.

De exemplu, să presupunem că un atacator a compromis un host Linux. Atunci *t0rn* îi va permite să facă următoarele acțiuni:

1. Să dezinstaleze *syslogd*.
2. Să-și stocheze parola pentru programe de tip Troian în */etc/ttyhash*.
3. Să instaleze o versiune troianizată a *sshd*, configurată pentru a urmări un anumit port al intrusului.

Rootkit permite atacatorilor să-și ascundă prezența pe o mașină pe care deja au compromis-o.

De exemplu, să presupunem că un atacator a compromis un host Linux. Atunci *t0rn* îi va permite să facă următoarele acțiuni:

1. Să dezinstaleze *syslogd*.
2. Să-și stocheze parola pentru programe de tip Troian în */etc/ttyhash*.
3. Să instaleze o versiune troianizată a *sshd*, configurată pentru a urmări un anumit port al intrusului.
4. Să ascundă nume de fișiere, de procese etc.

Rootkit permite atacatorilor să-și ascundă prezența pe o mașină pe care deja au compromis-o.

De exemplu, să presupunem că un atacator a compromis un host Linux. Atunci *t0rn* îi va permite să facă următoarele acțiuni:

1. Să dezinstaleze *syslogd*.
2. Să-și stocheze parola pentru programe de tip Troian în */etc/ttyhash*.
3. Să instaleze o versiune troianizată a *sshd*, configurată pentru a urmări un anumit port al intrusului.
4. Să ascundă nume de fișiere, de procese etc.
5. Să înlocuiască cu copii troianizate fișierele binare */bin/login*, */sbin/ifconfig*, */bin/ps*, */usr/bin/du*, */bin/ls*, */bin/netstat*, */usr/sbin/in.fingerd*, */usr/bin/find*, */usr/bin/top*.

Rootkit permite atacatorilor să-și ascundă prezența pe o mașină pe care deja au compromis-o.

De exemplu, să presupunem că un atacator a compromis un host Linux. Atunci *t0rn* îi va permite să facă următoarele acțiuni:

1. Să dezinstaleze *syslogd*.
2. Să-și stocheze parola pentru programe de tip Troian în */etc/ttyhash*.
3. Să instaleze o versiune troianizată a *sshd*, configurată pentru a urmări un anumit port al intrusului.
4. Să ascundă nume de fișiere, de procese etc.
5. Să înlocuiască cu copii troianizate fișierele binare */bin/login*, */sbin/ifconfig*, */bin/ps*, */usr/bin/du*, */bin/ls*, */bin/netstat*, */usr/sbin/in.fingerd*, */usr/bin/find*, */usr/bin/top*.
6. Să instaleze o parolă *sniffer*, *parser sniffer log file*, și *system log file*.

Rootkit permite atacatorilor să-și ascundă prezența pe o mașină pe care deja au compromis-o.

De exemplu, să presupunem că un atacator a compromis un host Linux. Atunci *t0rn* îi va permite să facă următoarele acțiuni:

1. Să dezinstaleze *syslogd*.
2. Să-și stocheze parola pentru programe de tip Troian în */etc/ttyhash*.
3. Să instaleze o versiune troianizată a *sshd*, configurată pentru a urmări un anumit port al intrusului.
4. Să ascundă nume de fișiere, de procese etc.
5. Să înlocuiască cu copii troianizate fișierele binare */bin/login*, */sbin/ifconfig*, */bin/ps*, */usr/bin/du*, */bin/ls*, */bin/netstat*, */usr/sbin/in.fingerd*, */usr/bin/find*, */usr/bin/top*.
6. Să instaleze o parolă *sniffer*, *parser sniffer log file*, și *system log file*.
7. Să restarteze */usr/bin/inetd*.

Rootkit permite atacatorilor să-și ascundă prezența pe o mașină pe care deja au compromis-o.

De exemplu, să presupunem că un atacator a compromis un host Linux. Atunci *t0rn* îi va permite să facă următoarele acțiuni:

1. Să dezinstaleze *syslogd*.
2. Să-și stocheze parola pentru programe de tip Troian în */etc/ttyhash*.
3. Să instaleze o versiune troianizată a *sshd*, configurată pentru a urmări un anumit port al intrusului.
4. Să ascundă nume de fișiere, de procese etc.
5. Să înlocuiască cu copii troianizate fișierele binare */bin/login*, */sbin/ifconfig*, */bin/ps*, */usr/bin/du*, */bin/ls*, */bin/netstat*, */usr/sbin/in.fingerd*, */usr/bin/find*, */usr/bin/top*.
6. Să instaleze o parolă *sniffer*, *parser sniffer log file*, și *system log file*.
7. Să restarteze */usr/bin/inetd*.
8. Să pornească *syslogd*.

După ce realizează aceste deziderate, atacatorul poate rula programe din sistem fără ca victima să aibă idee că sistemul ei este compromis.

După ce realizează aceste deziderate, atacatorul poate rula programe din sistem fără ca victima să aibă idee că sistemul ei este compromis.

Rootkit este o modalitate foarte eficientă pentru intruși de a-și ascunde prezența într-o rețea infectată.
Detectarea lui este extrem de dificilă. Există un utilitar, *Chkrootkit* care permite unui administrator să detecteze unele tooluri din rootkit în faza de execuție.

Remote Control Software

Controlul exterior de software nu este întotdeauna o slăbiciune a sistemului; el este utilizat adesea de multe ori în scopuri legitime. Multe sisteme de operare includ posibilitatea de a muta controlul înafara sistemului, ca o facilitate pentru inginerii *IT*.
Metoda poate fi folosită însă și de atacatori care doresc să preia de la distanță controlul unui sistem.

Remote Control Software

Controlul exterior de software nu este întotdeauna o slăbiciune a sistemului; el este utilizat adesea de multe ori în scopuri legitime. Multe sisteme de operare includ posibilitatea de a muta controlul înafara sistemului, ca o facilitate pentru inginerii *IT*.
Metoda poate fi folosită însă și de atacatori care doresc să preia de la distanță controlul unui sistem.
O metodă uzuală constă în trimiterea unui mesaj prin e-mail victimei, care conține în attach un soft de preluare exterioară a controlului (de ex. troianul *NetBus*). Odată rulat, procesul se auto-ascunde în sistem prin diverse mijloace – de exemplu redenumind procesul pe care îl execută.

Remote Control Software

Controlul exterior de software nu este întotdeauna o slăbiciune a sistemului; el este utilizat adesea de multe ori în scopuri legitime. Multe sisteme de operare includ posibilitatea de a muta controlul înafara sistemului, ca o facilitate pentru inginerii *IT*.
Metoda poate fi folosită însă și de atacatori care doresc să preia de la distanță controlul unui sistem.
O metodă uzuală constă în trimiterea unui mesaj prin e-mail victimei, care conține în attach un soft de preluare exterioară a controlului (de ex. troianul *NetBus*). Odată rulat, procesul se auto-ascunde în sistem prin diverse mijloace – de exemplu redenumind procesul pe care îl execută.
Atacul poate fi pasul inițial al unei palete largi de alte atacuri; lansarea unui atac *DDoS* poate începe cu preluarea controlului unui număr mare de sisteme.

Caracteristici:

1. **Membru al clasei/subclasei**: Composite.
2. **Exemple de implementări**: *Back Orifice 2000* (*BO2K*).
3. **Preliminarii**: Variabile.
4. **Vulnerabilități**: Software, configurare și utilizare.
5. **Utilizare tipică**: Controlează sistemul victimei dintr-o locație externă.
6. **Rezultate atac**: Variabilă (orice este posibil).
7. **Tipuri de atac folosite**: Citire, Manipulare, Spoof.
8. *OSI* **Layers**: $3 - 7$
9. **Detectare**: *IDS*.
10. **Protecție**: Aplicații de securitate, software antivirus.

Un tool de preluare a controlului foarte cunoscut este *Back Orifice 2000* sau *BO2K*. Acesta asigură comunicarea între client și server criptată cu sistemul *AES* și permite atacatorului să realizeze următoarele:

- Să blocheze mașina.

Un tool de preluare a controlului foarte cunoscut este *Back Orifice 2000* sau *BO2K*. Acesta asigură comunicarea între client și server criptată cu sistemul *AES* și permite atacatorului să realizeze următoarele:

- Să blocheze mașina.
- Să captureze toate cheile din sistem.

Un tool de preluare a controlului foarte cunoscut este *Back Orifice 2000* sau *BO2K*. Acesta asigură comunicarea între client și server criptată cu sistemul *AES* și permite atacatorului să realizeze următoarele:

- Să blocheze mașina.
- Să captureze toate cheile din sistem.
- Să ruleze un fișier *.wav* sau să arate un mesaj pe monitor.

Un tool de preluare a controlului foarte cunoscut este *Back Orifice 2000* sau *BO2K*. Acesta asigură comunicarea între client și server criptată cu sistemul *AES* și permite atacatorului să realizeze următoarele:

- Să blocheze mașina.
- Să captureze toate cheile din sistem.
- Să ruleze un fișier *.wav* sau să arate un mesaj pe monitor.
- Să aducă alte atacuri prin plug-in-uri.

Un tool de preluare a controlului foarte cunoscut este *Back Orifice 2000* sau *BO2K*. Acesta asigură comunicarea între client și server criptată cu sistemul *AES* și permite atacatorului să realizeze următoarele:

- Să blocheze mașina.
- Să captureze toate cheile din sistem.
- Să ruleze un fișier *.wav* sau să arate un mesaj pe monitor.
- Să aducă alte atacuri prin plug-in-uri.
- Să caute și să transfere sistemul de fișiere locale.

Un tool de preluare a controlului foarte cunoscut este *Back Orifice 2000* sau *BO2K*. Acesta asigură comunicarea între client și server criptată cu sistemul *AES* și permite atacatorului să realizeze următoarele:

- Să blocheze mașina.
- Să captureze toate cheile din sistem.
- Să ruleze un fișier *.wav* sau să arate un mesaj pe monitor.
- Să aducă alte atacuri prin plug-in-uri.
- Să caute și să transfere sistemul de fișiere locale.
- Să editeze regiștrii.

Un tool de preluare a controlului foarte cunoscut este *Back Orifice 2000* sau *BO2K*. Acesta asigură comunicarea între client și server criptată cu sistemul *AES* și permite atacatorului să realizeze următoarele:

- Să blocheze mașina.
- Să captureze toate cheile din sistem.
- Să ruleze un fișier *.wav* sau să arate un mesaj pe monitor.
- Să aducă alte atacuri prin plug-in-uri.
- Să caute și să transfere sistemul de fișiere locale.
- Să editeze regiștrii.

Toate acestea pot fi executate printr-un port specificat de utilizator folosind *UDP*, *TCP* sau chiar *ICMP*. Odată sistemul infectat, el devine "*zombie*" (**bot**): atacatorul poate face orice cu el, ca și cum ar fi în fața ecranului.

Sfârșit