

Securitatea LAN și atacuri de rețea

Firewall-ul de la marginea rețelei, oricât de bine configurat, nu vede absolut nimic din ce se întâmplă între două calculatoare din aceeași încăpere. Traficul lor nu îl traversează niciodată. Cursul acesta se ocupă exact de acest spațiu orb: atacurile care funcționează în interiorul rețelei locale, aproape toate exploatând mecanisme pe care le-am construit noi înșine, în cursurile 3 și 4, și care fac exact ce au fost proiectate să facă. Nu vom repara protocoalele - nu se poate. Vom adăuga în jurul lor verificările care le lipsesc.

Din capitolul precedent.

- Un switch **învață** adresele MAC din câmpul **sursă** al cadrelor primite.
- Tabela **CAM** are dimensiune finită; ce nu se găsește în ea se **inundă** pe toate porturile.
- **ARP** asociază o adresă IP cu o adresă MAC, întrebând prin broadcast.
- **VLAN-ul nativ** circulă **neetichetat** pe un trunk.
- **STP** alege ca rădăcină switch-ul cu cel mai mic Bridge ID.
- **DHCP** funcționează prin broadcast; clientul acceptă prima ofertă sosită.

OBIECTIVELE CAPITOLULUI

- Să clasificați un atac după scop și să spuneți ce mecanism exploatează
- Să explicați mecanica atacurilor CAM overflow, ARP poisoning și VLAN hopping
- Să descrieți cum poate un singur cablu opri o rețea întreagă prin STP
- Să configurați port security, DHCP snooping și Dynamic ARP Inspection
- Să închideți VLAN hopping-ul prin trei setări
- Să explicați ce face și, mai ales, ce **nu** face un salt
- Să aplicați o listă completă de securizare pe un switch de acces

Fiecare dintre aceste șase propoziții este, astăzi, punctul de plecare al unui atac. Nu pentru că ar fi implementate greșit, ci pentru că sunt implementate **exact așa cum**

au fost gândite - într-o epocă în care rețeaua locală era, implicit, un spațiu de încredere.

12.1 Trei familii de atacuri

De ce firewall-ul nu ajută aici. Un firewall stă **între** două rețele și vede doar traficul care îl traversează.

Dacă stația A și stația B sunt în același VLAN, pe același switch, traficul dintre ele nu ajunge niciodată la un ruter, deci nici la firewall. Switch-ul îl comută direct, la nivelul 2.

Toate atacurile din acest curs se poartă **sub** nivelul la care privește un firewall clasic. Apărarea trebuie să fie tot acolo - pe switch.

Tabelul 12.1 Trei familii de atacuri.

Familie	Scop	Exemple în LAN
Recunoaștere	aflarea a ce există și ce rulează	ping sweep, port scan, sniffing, interogări DNS
Blocare (DoS)	indisponibilizarea unui serviciu	SYN flood, smurf, CAM overflow, atac STP
Acces	obținerea de drepturi nemeritate	ARP poisoning, VLAN hopping, spargere de parole, inginerie socială

Observația care organizează tot cursul. Aproape fiecare atac de aici exploatează un mecanism **care funcționează exact așa cum a fost proiectat**.

ARP nu autentifică nimic - prin proiect. Switch-ul crede orice MAC sursă - prin proiect. STP acceptă orice BPDU - prin proiect. DHCP-ul acceptă prima ofertă - prin proiect.

Toate au fost gândite într-o epocă în care a fi conectat fizic la rețea însemna a fi de încredere. Contramăsurile nu repară protocoalele; adaugă în jurul lor verificările care lipsesc.

Analogie E ca într-o clădire de birouri unde toate ușile interioare sunt deschise, pentru că oricine a trecut de recepție este considerat angajat.

Firewall-ul este recepția. Măsurile din acest curs sunt încuietorile de pe ușile interioare - și motivul pentru care se numesc, împreună, **securitate în adâncime**.

12.2 Recunoașterea

Înainte oricărui atac serios vine culegerea de informații: ce adrese IP sunt active, ce servicii rulează pe ele, în ce versiune, ce sisteme de operare și în cine au încredere utilizatorii rețelei.

Aceleași unelte sunt folosite zilnic și de administratori - diferența este **autorizarea**, nu instrumentul.

Tabelul 12.2 Recunoașterea.

Utilitar	Ce face	Exemplu
nmap	descoperă stații active, porturi deschise, servicii, versiuni, sistem de operare	<code>nmap -sV 192.168.1.0/24</code>
tcpdump	captură de trafic din linia de comandă, cu filtre	<code>tcpdump -i eth0 -c 10 dst port 80</code>
Wireshark	captură și analiză grafică; cere interfața în mod promiscuu	format libpcap, comun cu tcpdump
whois	informații administrative despre un domeniu	<code>whois example.com</code>
host, dig	interogări DNS: servere de nume, de mail	<code>host -t MX example.com</code>

Regula de aur a acestui curs. Toate uneltele de mai sus se folosesc **exclusiv** pe rețeaua de laborator, pe propriile dumneavoastră dispozitive.

Scanarea sau interceptarea traficului altcuiva - chiar și în aceeași rețea, chiar și „doar ca să văd dacă merge” - este ilegală și se pedepsește. Scopul acestui curs este să înțelegeți ce vede un atacator ca să puteți construi apărarea, nu să exersați pe sisteme străine.



Figura 12.1 Terminal: ce vede o recunoaștere.

Priviți a treia comandă. Două adrese IP diferite cu **aceeași** adresă MAC este anomalia care trădează un atac de tip man-in-the-middle - și este verificarea cea mai ieftină pe care o puteți face pe o stație suspectă.

Ce poate face un firewall împotriva recunoașterii: să blocheze răspunsul la ICMP echo, să închidă porturile vulnerabile, să interzică inițierea conexiunilor din exterior, să limiteze rata cererilor. Ce **nu** poate face: să oprească un atacator aflat deja în rețeaua locală. De aici, restul cursului.

12.3 Atacuri de blocare a serviciului

Smurf attack



Figura 12.2 Smurf attack: amplificarea.

TCP SYN flood

L-am întâlnit în cursul 8: atacatorul trimite mii de segmente SYN cu adrese sursă falsificate, iar serverul alocă câte o intrare în coada de conexiuni **pe jumătate deschise** pentru fiecare. Când coada se umple, conexiunile legitime sunt refuzate.

Apărări: **SYN cookies**, care permit serverului să nu aloce nimic până la confirmare; limitarea numărului de conexiuni half-open per sursă; și, ca întotdeauna, monitorizarea traficului în condiții normale - o anomalie se observă doar dacă știți cum arată normalul.

12.4 CAM overflow

Acesta este atacul care exploatează direct mecanismul din cursul 3. Memoria unui switch este finită - tipic între 8 000 și 32 000 de intrări - deci tabela CAM se poate umple. Iar când se umple, switch-ul nu mai poate învăța adrese noi și **inundează** toate cadrele necunoscute pe toate porturile. Cu alte cuvinte: redevine un hub.



Figura 12.3 CAM overflow, pas cu pas.

Reluați simulatorul de tabelă CAM din cursul 3 și urmăriți ce se întâmplă la fiecare cadru: switch-ul învață **doar** din adresa sursă și inundă **orice** destinație necunoscută. Atacul nu face decât să împingă aceste două reguli la limită.



Figura 12.4 Simulator de tabelă CAM - învățare și inundare.

```
SW(config)# interface fastEthernet 0/5
SW(config-if)# switchport mode access
SW(config-if)# switchport port-security
SW(config-if)# switchport port-security maximum 2
SW(config-if)# switchport port-security mac-address sticky
SW(config-if)# switchport port-security violation restrict
SW(config-if)# switchport port-security aging time 60

SW# show port-security interface fastEthernet 0/5
SW# show mac address-table count
```

Listingul 12.1 contramăsura: port security copiază

Tabelul 12.3 CAM overflow.

Acțiune la violare	Ce face	Când o folosiți
protect	aruncă tăcut cadrele în exces	rar - nu aflați niciodată că s-a întâmplat ceva
restrict	aruncă, incrementează un contor, trimite SNMP și syslog	implicit recomandat
shutdown	închide portul (err-disable)	zone sensibile; cere intervenție manuală sau <code>errdisable recovery</code>

Ce înseamnă sticky. Adresele învățate dinamic sunt scrise automat în configurația activă, ca și cum le-ați fi introdus manual. Avantajul: nu trebuie să cunoașteți dinainte adresa fiecărui calculator.

Atenție însă: dacă nu salvați configurația, ele se pierd la repornire - iar dacă le salvați, portul rămâne legat de acel calculator anume, ceea ce devine o problemă la prima înlocuire de echipament.

12.5 Man-in-the-middle prin ARP poisoning

ARP, așa cum l-am studiat în cursul 5, are o proprietate care astăzi pare de neconceput: **nu autentifică nimic**. Mai mult, majoritatea implementărilor acceptă un răspuns ARP chiar dacă nu au trimis nicio cerere - mecanism numit *gratuitous ARP*, util pentru anunțarea unei schimbări de adresă, dar exploatabil.

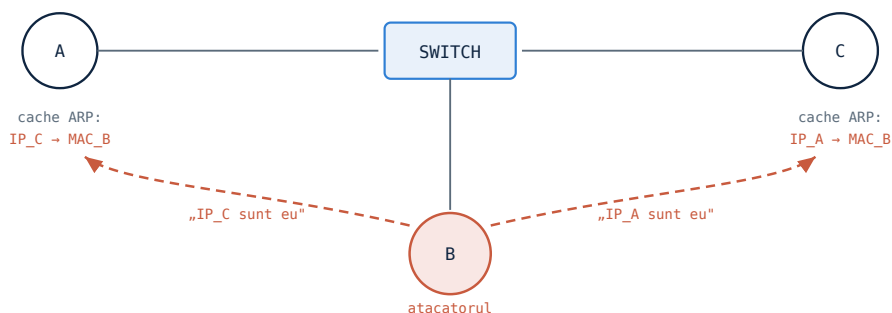


Figura 12.5 ARP poisoning. B trimite fiecăruia dintre cei doi un răspuns ARP nesolicitat. Ambele stații își actualizează cache-ul și încep să adreseze cadrele către MAC-ul lui B. Switch-ul le comută corect - către atacator.



Figura 12.6 ARP poisoning și cum se închide.

```
! 1. switch-ul invata ce adresa IP a primit fiecare port, prin DHCP
SW(config)# ip dhcp snooping
SW(config)# ip dhcp snooping vlan 10,20
SW(config)# interface gigabitEthernet 0/1
SW(config-if)# ip dhcp snooping trust

! 2. pe baza acestei evidente, se verifica fiecare mesaj ARP
SW(config)# ip arp inspection vlan 10,20
SW(config)# interface gigabitEthernet 0/1
SW(config-if)# ip arp inspection trust

! 3. pentru statiile cu adresa statica, evidenta se scrie manual
SW(config)# arp access-list SERVERE
SW(config-arp-nacl)# permit ip host 192.168.1.50 mac host 000c.2911.2233
SW(config)# ip arp inspection filter SERVERE vlan 10

SW# show ip dhcp snooping binding
SW# show ip arp inspection statistics
```

Listingul 12.2 contramăsura: DHCP snooping + Dynamic ARP Inspection copiază

Un beneficiu suplimentar: rogue DHCP. DHCP snooping oprește, în același timp, un atac complet diferit: cineva își pornește propriul server DHCP și se anunță drept gateway al tuturor. Clienții acceptă prima ofertă sosită - iar un server local, în aceeași încăpere, răspunde mai repede decât cel legitim.

Porturile *nettrusted* nu au voie să trimită mesaje de server (Offer, Ack), deci atacul nici nu pornește. Aceeași comandă, două atacuri închise.

12.6 VLAN hopping

Am construit VLAN-uri tocmai ca să izolăm departamente. Există două tehnici prin care izolarea poate fi ocolită - ambele exploatând comportamente **implicite** ale switch-urilor.

Switch spoofing

Multe switch-uri Cisco au implicit activat **DTP** (Dynamic Trunking Protocol), care negociază automat dacă un port devine acces sau trunk. Un atacator poate să se prezinte drept switch și să negocieze un trunk. Odată obținut, poate trimite și primi cadre etichetate 802.1Q pentru **orice** VLAN.

```
SW(config)# interface range fastEthernet 0/1 - 22
SW(config-if-range)# switchport mode access
SW(config-if-range)# switchport nonegotiate
SW(config-if-range)# switchport access vlan 10

! porturile nefolosite: inchise si mutate intr-un VLAN mort
SW(config)# interface range fastEthernet 0/23 - 24
SW(config-if-range)# switchport access vlan 999
SW(config-if-range)# shutdown
```

Listingul 12.3 contramăsura: nicio negociere pe porturile de acces copiază

Double tagging

Tehnica exploatează **VLAN-ul nativ**, despre care știm din cursul 4 că circulă **neetichetat** pe trunk. Nu necesită niciun protocol special pe atacator - doar posibilitatea de a construi un cadru cu două etichete.



Figura 12.7 Double tagging, pas cu pas.

Trei reguli care închid VLAN hopping-ul. 1. VLAN-ul nativ al trunk-urilor să fie un VLAN nefolosit, în care nu se află nicio stație.

2. Porturile de acces să fie `explicit switchport mode access` cu `switchport nonegotiate`.

3. Pe trunk-uri, lista de VLAN-uri permise să fie declarată explicit (`switchport trunk allowed vlan 10,20,30`), nu lăsată implicit la „toate”.

12.7 Atacuri împotriva STP

STP nu folosește nicio formă de autentificare: **orice** dispozitiv care trimite BPDU-uri participă la alegerea rădăcinii. Un atacator conectat oriunde în rețea poate anunța BPDU-uri cu un Bridge ID foarte mic și deveni root bridge - moment în care o parte însemnată a traficului începe să treacă prin el.



Figura 12.8 Alegerea rădăcinii și rolurile porturilor.

Coborâți prioritatea unui switch la 0 în simulatorul de mai sus și priviți cum se reorganizează **întreaga** topologie în jurul lui. Exact asta face atacatorul - doar că switch-ul cu prioritate 0 este laptopul lui, aflat sub birou.

1. Atacatorul se conectează la rețeaua de switch-uri, cu un singur cablu.
2. Anunță BPDU-uri cu prioritatea **0**, deci cu cel mai mic BID posibil.
3. STP reconverge; el devine rădăcină, iar rolurile porturilor se recalculează în jurul lui.
4. Traficul dintre segmente trece acum prin mașina lui, unde poate fi capturat.

Varianta de blocare a serviciului este și mai simplă: atacatorul își **schimbă continuu BID-ul**, forțând recalculări permanente. Porturile nu ajung niciodată în starea forwarding, iar rețeaua este, practic, oprită - cu un singur cablu și niciun exploit.

```

! porturile spre statii: trec direct in forwarding...
SW(config)# interface range fastEthernet 0/1 - 22
SW(config-if-range)# spanning-tree portfast

! ...si se inchid daca de acolo apare un BPDU
SW(config-if-range)# spanning-tree bpduguard enable

! radacina se stabileste explicit, pe switch-ul din centrul rețelei
SW_CORE(config)# spanning-tree vlan 1-100 root primary

! porturile spre switch-uri care nu au voie sa devina radacina
SW(config)# interface gigabitEthernet 0/2
SW(config-if)# spanning-tree guard root

! reactivarea automata a porturilor inchise, dupa 5 minute
SW(config)# errdisable recovery cause bpduguard
SW(config)# errdisable recovery interval 300

```

Listingul 12.4 contramăsurile STP copiază

Tabelul 12.4 Atacuri împotriva STP.

Mecanism	Ce face	Unde se pune
PortFast	sare peste listening și learning pe porturile de acces	porturi spre stații; niciodată spre switch-uri
BPDU Guard	închide portul dacă primește un BPDU	împreună cu PortFast, întotdeauna
Root Guard	împiedică un port să devină port rădăcină	pe porturile spre switch-uri care nu au voie să fie rădăcină
BPDU Filter	oprește complet trimiterea și primirea de BPDU-uri	rar; de folosit cu prudență, poate crea bucle

PortFast fără BPDU Guard este periculos. PortFast singur nu protejează pe nimeni: doar grăbește trecerea în forwarding. Dacă cineva conectează un switch pe acel port, se poate forma o buclă înainte ca STP să apuce să reacționeze.

Cele două se pun întotdeauna împreună. Regula mnemotehnică: *PortFast promite că nu e niciun switch acolo; BPDU Guard verifică promisiunea.*

12.8 Parole, hash-uri și salt

Tabelul 12.5 Parole, hash-uri și salt.

Metodă	Cum funcționează	Contramăsură
Sniffing	parolele trimise în clar (Telnet, FTP, HTTP) se citesc direct din trafic	SSH, HTTPS - orice cu criptare
Brute force	se încearcă toate combinațiile posibile	parole lungi; limitarea încercărilor
Atac de dicționar	se încearcă cuvinte uzuale și variații simple ale lor	parole care nu sunt cuvinte
Rainbow tables	lanțuri de hash-uri precalculate, care schimbă spațiu pe timp	salt

De ce nu se stochează parolele, ci hash-urile lor. Un **hash** este o funcție într-un singur sens: din parolă se calculează instantaneu, dar din rezultat nu se poate reveni la parolă.

Serverul stochează doar hash-ul. La autentificare, calculează hash-ul parolei introduse și compară. Dacă baza de date este furată, atacatorul nu are parolele - are doar hash-urile.

Problema: dacă doi utilizatori au aceeași parolă, au și **același hash**. Iar un atacator poate precalcuła hash-urile celor mai frecvente câteva milioane de parole și le poate căuta direct. Aici intră în scenă rainbow tables.

O **rainbow table** nu stochează toate hash-urile posibile - ar fi imposibil - ci punctele de pornire ale unor lanțuri de hash-uri, din care valorile intermediare se recalculează la nevoie. Este un compromis clasic memorie-timp, iar tabele publice de mii de gigaocteți sunt disponibile pe Internet.

Salt-ul

Un **salt** este un segment aleator, generat pentru fiecare utilizator și concatenat la parolă înainte de hashing. Este stocat alături de hash, în clar - nu este secret și nici nu trebuie să fie.

```
trudy:$6$/tKy92iM$/cIxbEX49qHpZt74D5L0W1vXO2fJuXjyXJnsT0.M...
#      ^      ^      ^
#      |      |      hash-ul propriu-zis
#      |      salt-ul, generat aleator
#      algoritmul: 6 = SHA-512
```

Listingul 12.5 o linie din `/etc/shadow`

Ce face și ce nu face salt-ul. Salt-ul **nu previne** atacurile cu rainbow tables și nu face parola mai greu de ghicit.

Ce face: obligă atacatorul să construiască o tabelă nouă **pentru fiecare utilizator în parte** - ceea ce transformă un atac de câteva minute asupra întregii baze de date într-un efort separat per cont. Și face ca doi utilizatori cu aceeași parolă să aibă hash-uri diferite.

Este o măsură de **scalare**, nu de imposibilitate: o parolă slabă rămâne slabă și cu salt. Pentru rezistență reală se folosesc funcții deliberat **lente** - bcrypt, scrypt, Argon2 - care fac fiecare încercare costisitoare.

12.9 Factorul uman și codul ostil

Ingineria socială

Cea mai eficientă cale de acces nu este tehnică. Atacatorul convinge o persoană că este de încredere - un coleg de la IT, un furnizor, un superior grăbit - și obține direct informația. Ținta nu are, de regulă, profil tehnic, vrea să ajute și nu conștientizează valoarea a ceea ce deține.

Ingineria socială **ocolțește** orice mecanism tehnic, oricât de bine configurat. Nu există port security pentru un telefon. Singura apărare reală este instruirea periodică a angajaților non-tehnici și existența unor **proceduri clare** pentru cererile neobișnuite - de exemplu, regula că o parolă nu se comunică niciodată telefonic, indiferent cine sună.

Phishing-ul este varianta industrializată a aceleiași idei: în loc de un apel, un milion de mesaje.

Cod executabil ostil

Tabelul 12.6 Cod executabil ostil.

Tip	Cum se propagă	Ce îl caracterizează
Virus	atașat unui program; are nevoie ca utilizatorul să îl ruleze	efect direct, adesea vizibil
Troian	ascuns într-o aplicație aparent utilă	efect subtil - de regulă deschide o porțiță de acces; greu de detectat
Vierme	exploatează vulnerabilități; nu are nevoie de utilizator	răspândire foarte rapidă; adesea construiește botnet-uri
Ransomware	oricare din căile de mai sus	criptează datele și cere răscumpărare; se propagă lateral în LAN

De ce segmentarea contează atât de mult azi. Un ransomware modern nu se mulțumește cu stația infectată: se propagă **lateral**, exploatănd exact încrederea implicită din rețeaua locală.

VLAN-urile, ACL-urile între ele și măsurile din acest curs nu împiedică infecția inițială - dar limitează dramatic cât de departe ajunge. Este diferența dintre un incident și un dezastru.

12.10 Lista scurtă de securizare a unui switch

Tot ce s-a discutat mai sus se rezumă la câteva configurații care se aplică oricărui switch de acces, în orice rețea de organizație.



Figura 12.9 Măsura și atacul corespunzător.

Tabelul 12.7 Lista scurtă de securizare a unui switch.

Măsură	Împotriva cui
port-security cu maximum și sticky	CAM overflow, dispozitive neautorizate
ip dhcp snooping	servere DHCP false
ip arp inspection	ARP poisoning, man-in-the-middle
switchport mode access + nonegotiate	switch spoofing
VLAN nativ nefolosit, VLAN-uri permise explicit	double tagging
portfast + bpduguard	atacuri STP
porturile nefolosite: shutdown, în VLAN mort	conectări neautorizate
SSH în loc de Telnet, cu ACL pe liniile VTY	interceptarea parolelor de administrare

```
SW(config)# hostname SW1
SW(config)# ip domain-name retele.local
SW(config)# crypto key generate rsa modulus 2048
SW(config)# ip ssh version 2
SW(config)# username admin privilege 15 secret ParolaPuternica
SW(config)# service password-encryption

SW(config)# access-list 20 permit 10.0.100.0 0.0.0.255
SW(config)# line vty 0 15
SW(config-line)# transport input ssh
SW(config-line)# login local
SW(config-line)# access-class 20 in
SW(config-line)# exec-timeout 5 0
```

Listingul 12.6 administrare prin SSH, nu prin Telnet copiază



Figura 12.10 Ordinea de securizare a unui switch nou.

12.11 Greșeli frecvente

Greșeli frecvente.

„Am pus portfast pe toate porturile, inclusiv spre switch-uri”

Se poate forma o buclă înainte ca STP să reacționeze - și o buclă de nivel 2 oprește rețeaua în câteva secunde.

PortFast doar spre stații, întotdeauna cu BPDU Guard alături.

„Am activat DAI, dar serverele cu adrese statice nu mai comunică”

DAI se bazează pe evidența construită de DHCP snooping. O stație care nu a folosit DHCP nu apare acolo.

Adăugați intrări manuale cu arp access-list pentru serverele cu adresă fixă.

„Am pus port-security maximum 1 peste tot”

Un telefon IP cu PC în spate are două adrese; o mașină virtuală în bridge adaugă altele.

Alegeți maximul după ce chiar se conectează acolo. 2 sau 3 e mai realist.

„Am lăsat VLAN 1 ca VLAN nativ”

Este valoarea implicită și, adesea, chiar VLAN-ul în care se află stațiile - precondiția exactă a atacului double tagging.

VLAN nativ dedicat, gol, diferit de 1 și de orice VLAN de utilizatori.

„Am configurat totul, dar nu am salvat”

Adresele sticky și întreaga configurație dispar la prima repornire.

copy running-config startup-config . Aceeași lecție ca în cursul 6.

„Port security cu violation shutdown, și acum jumătate din porturi sunt închise”

Un utilizator care își schimbă laptopul închide portul, iar reactivarea cere intervenție manuală.

restrict pentru zonele obișnuite; errdisable recovery acolo unde folosiți shutdown .

„Avem firewall, deci rețeaua internă e sigură”

Firewall-ul nu vede traficul dintre două stații din același VLAN.

Securitate în adâncime: segmentare, măsuri pe switch, criptare între capete.

12.12 Rezumat și glosar

Ce trebuie să rămână.

- Firewall-ul de margine **nu vede** traficul din interiorul unui VLAN.
- Aproape toate atacurile din LAN exploatează protocoale care funcționează **corect**.
- **CAM overflow** falsifică adrese MAC **sursă** → se încheie cu `port-security`.
- **ARP poisoning** exploatează lipsa autentificării în ARP → se încheie cu **DHCP snooping + DAI**.
- **Switch spoofing** exploatează DTP → `nonegotiate`.
- **Double tagging** exploatează VLAN-ul nativ → VLAN nativ nefolosit.
- **Atacul STP** exploatează lipsa autentificării BPDU → **PortFast + BPDU Guard**.
- **Salt-ul** nu întărește parola: obligă atacatorul la o tabelă per cont.
- **Ingineria socială** ocolește orice măsură tehnică. Se apără cu proceduri și instruire.
- Securitatea este **în adâncime**: nicio măsură singură nu ajunge.

Exerciții

1. Ce adrese MAC trebuie falsificate într-un atac de tip CAM overflow?
 - a. Adresele destinație
 - b. Adresele sursă, pentru ca acestea sunt cele invatate de switch
 - c. Adresa de broadcast
 - d. Adresa punctului de acces
2. Care este simptomul cel mai vizibil al unui CAM overflow reusit?
 - a. Porturile se inchid
 - b. Switch-ul incepe sa inunde traficul pe toate porturile, deci se comporta ca un hub
 - c. Adresele IP se schimba
 - d. Se pierde legatura cu gateway-ul
3. De ce functioneaza ARP poisoning?
 - a. Pentru ca switch-urile nu au tabela CAM
 - b. Pentru ca ARP nu autentifica nimic, iar statiile accepta raspunsuri pe care nu le-au cerut
 - c. Pentru ca IP-ul nu are checksum
 - d. Pentru ca VLAN-urile nu sunt configurate

4. Pe ce se bazează Dynamic ARP Inspection ca să decida dacă un mesaj ARP este fals?
 - a. Pe o listă de adrese MAC configurată manual
 - b. Pe evidența perechilor port-MAC-IP construită de DHCP snooping
 - c. Pe tabela de rutare
 - d. Pe numărul VLAN-ului

5. Ce condiție trebuie îndeplinită pentru un atac de tip double tagging?
 - a. Atacatorul trebuie să aibă acces fizic la trunk
 - b. VLAN-ul atacatorului trebuie să coincidă cu VLAN-ul nativ al trunk-ului
 - c. Switch-ul trebuie să ruleze RSTP
 - d. Toate VLAN-urile trebuie să aibă același ID

6. De ce atacul de tip double tagging este unidirecțional?
 - a. Pentru că switch-urile filtrează răspunsurile
 - b. Pentru că răspunsul ar pleca din VLAN-ul țintă și nu are cum să ajungă înapoi la atacator
 - c. Pentru că eticheta interioară se pierde
 - d. Pentru că STP blochează calea de întoarcere

7. Ce combinație de comenzi protejează cel mai bine un port de acces împotriva unui atac STP?
 - a. spanning-tree portfast, singur
 - b. spanning-tree portfast împreună cu spanning-tree bpduguard enable
 - c. spanning-tree bpdufilter
 - d. switchport nonegotiate

8. Ce realizează folosirea unui salt la stocarea parolelor?
 - a. Face imposibilă spargerea parolei
 - b. Obligă atacatorul să construiască o rainbow table separată pentru fiecare utilizator
 - c. Crijtează parola
 - d. Elimină nevoia de parole complexe

9. Care afirmație despre firewall-ul de margine este corectă?

- a. Protejează și traficul dintre două stații din același VLAN
- b. Nu vede traficul dintre două stații din același VLAN, pentru că acesta nu îl traversează
- c. Înlocuiește nevoia de port security
- d. Detectează automat ARP poisoning

Note bibliografice

Cu acest curs se încheie parcursul. Am pornit, în cursul 1, de la un semnal electric pe un fir și de la întrebarea cum se transformă un bit într-o tensiune. Am ajuns la o rețea locală segmentată în VLAN-uri, rutată dinamic, filtrată cu liste de acces, translatată la ieșire, extinsă fără fir și apărată împotriva atacurilor din interiorul ei.

Ce merită să rămână, dincolo de comenzi: aproape fiecare mecanism pe care l-am studiat a fost inventat ca **soluție** la o problemă concretă, iar aproape fiecare a devenit, ulterior, **problema** altcuiva. Switch-ul a rezolvat coliziunile și a creat CAM overflow. VLAN-ul a rezolvat difuzarea și a creat VLAN hopping. NAT-ul a salvat IPv4 și a rupt peer-to-peer. Nu există decizie de proiectare fără cost - există doar decizii luate în cunoștință de cauză și decizii luate din obișnuință.

Laboratorul 7 execută, controlat, trei dintre atacurile de aici și apoi le încheie, pe rând, ușile.

- Cisco - *Catalyst Switch Security Configuration Guide*: port security, DHCP snooping, DAI
- RFC 2827 / BCP 38 - filtrarea la intrarea în rețea împotriva falsificării adreselor
- RFC 3704 - filtrarea la intrare pentru rețele cu legături multiple
- OWASP - resurse despre parole, hashing și autentificare
- Documentația nmap și Wireshark - pentru uz strict autorizat

Extras din volumul „Rețele Locale - Note de curs și lucrări de laborator”. Varianta online a acestui capitol conține figurile în formă interactivă.