SIGURNOST

Informacijska sigurnost se svodi na tehničke mjere (bolja tehnička rješenja)

Sigurnost bi se trebala dodavati tokom razvoja proizvoda – povećava rizik propuštanja poslovnih prilika (duža proizvodnja).

Velika vrijednost u dolasku prvi na tržište: dominiranje, niski troškovi održavanja nakon uspostave, postavljanje prepreka (visoki troškovi prebacivanja)

STVARNI TROŠKOVI: vrijeme izrade i složenost

Korisnici bi platili više kad bi proizvodi bili sigurniji – tj. inkrementalni troškovi su u redu.

Troškovi propuštene prilike su preveliki za prodavača – to djeluje destimulativno na izgradnju sigurnosti od početka.

Vrijednost resursa se mijenja s brojem potrošača.

Metcalfov zakon – vrijednost mreže se povećava s kvadratom broja čvorova (N2).

Proizvod ima veću temeljnu vrijednost ako ima više korisnika.

Sigurnosne značajke OS-a ili mreže čine život težim za developere – npr. Record locking (nužno, ali primjenu čini složenijom).

Developeri su primarna meta za OS i mrežu distributera.

Nastaje implicitni sporazum za prebacivanje sigurnosnih trošaka na korisnike (nije apsolutno potrebno za aplikacije).

Dodavanje sigurnosnih značajki je u korist dobavljača, a ne korisnika:

* „Lock-in“ korisnici
* Maksimalno povećanje prihoda
* Zaštita „on-going“ prihoda
* Dobivanje podataka o tržištu

Lock-in korisnici – koristiti vlasničke mjere sigurnosti (vjerojatno smanjuje pouzdanost i stabilnost [?])

* Dobavljač može kontrolirati
* Može stvoriti prihod
* Blokiranje ili ometanje konkurencije
* Korisnici se bolje upoznaju sa proizvodom – teže prebacivanje na drugi proizvod

Maksimalno povećanje prihoda – koristiti sigurnost kao značajku čija je nadogradnja skupa

* Inkrementalni trošak pemali ili neprimjetni
* Mogu naplatiti mnogo za to
* Npr. Avionske cijene (Non-IT) ili osnovni proizvod u odnosu na „Gold“ verziju (IT)

Zaštita prihoda – korištenje sigurnosti kako bi se spriječio „reverse engineering“   
 – koristiti sigurnosne mjere kako bi se spriječio dodatak generičkih proizvoda (npr. printer cartridges)

Zaštita i skupljanje podataka

RFID (Radio-frequency identification) – pomaže u sprječavanju krađe   
 – stvara prihode (npr. toll tags)   
 – prati inventar i isporuke   
  
 – velika prijetnja privatnosti (može pratiti kretanje vozila i ljude=

Dobivanje podataka o tržištu – MS Passport (dobar primjer lošeg primjera)

* Navedena svrha je pružanje sigurnosti prilikom posjete raznim web stranicama
* Ali Passport prati vaše pretraživanje
* I dijeli vaše podatke
* I daje negativcima mogućnost napada na samo jednu točku

Danas u sigurnosnim pitanjima, napadači imaju prednost

* Lakše je pronaći jedan nedostatak nego naći i popraviti (patch) sve nedostatke

Model ulaganja u napad i obranu

* Procijeniti broj „bug-ova“ i ulaganja u nalaženje
* Napadačeva prednost je velika

Tko određuje kvalitetu sigurnosti?

* Međunarodni standardi za postojanje sigurnosti
* Standardi su više proces nego sadržaj
  + Nema apsolutnog standarda
  + Kupac kaže što želi u sigurnosti
  + Prodavač provjerava proizvod prema zahtjevima
* Trenutni radni standard se zove „Zajednički kriterij“ (Common Criteria)

Tko plaća procjenu?

* Trebao bi kupac, ali to je veliki trošak za svakog pojedinog kupca
* Trenutna praksa je da prodavač plaća procjenu (evaluator)
* Ovo dovodi do kupnje „lakšeg“ (jeftinijeg) evaluatora
* Proizvođač softvera može uzeti u obzir da proizvod koji je prošao „slabiju“ evaluaciju ima manju vrijednost
  + Ako proizvođač ugrađuje sigurnost u svoj proizvod i to ne uspije, proizvođač je odgovoran ako je sigurnost proizvoda certificirana.

Zašto IT prodavači ne pružaju sigurnost?   
Ekonomski razlog: kreiranje monopola, povećanje prihoda, smanjivanje rizika (ekonomija promiče nesigurnost)

NIST (National Institute od Standards and Technology) Priručnik računalne sigurnosti

* Zaštita dodijeljena automatiziranom informacijskom sustavu u svrhu postizanja primjenjivih ciljeva očuvanja cjelovitosti, raspoloživosti i povjerljivosti resursa informacijskog sustava  
  (uključuje hardware, software, firmware, informacije/podaci i telekomunikaciju)

Računalna sigurnost se temelji na CIA

* Confidentiality (povjerljivost)
* Integrity (cjelovitost)
* Availability (dostupnost)

Temeljni sigurnosni ciljevi za podatke i računalne usluge (npr. hardver, softver, podaci, telekomunikacije).

Točno tumačenje tri aspekta ovisi o kontekstu u kojem oni nastaju.

Povjerljivost

* Odnosi se na skrivanje informacija ili resursa
  + Samo ovlaštene osobe ili sustavi mogu pristupiti zaštičenim podacima
* Odnosi se i na postojanje podataka/resursa
  + Ponekad važnija od samih podataka (privatnost)
  + „Političar optužen za korupciju“ važniji je od „političar plaća 1000˘mita da sakrije slučaj „vožnje u pijanom stanju““
* Mehanizam kontrole pristupa podržava povjerljivost
  + Kontrola pristupa putem šifriranja (kriptografija)
  + Kontrola pristupa putem lozinke i dopuštenje pristupa

Integritet

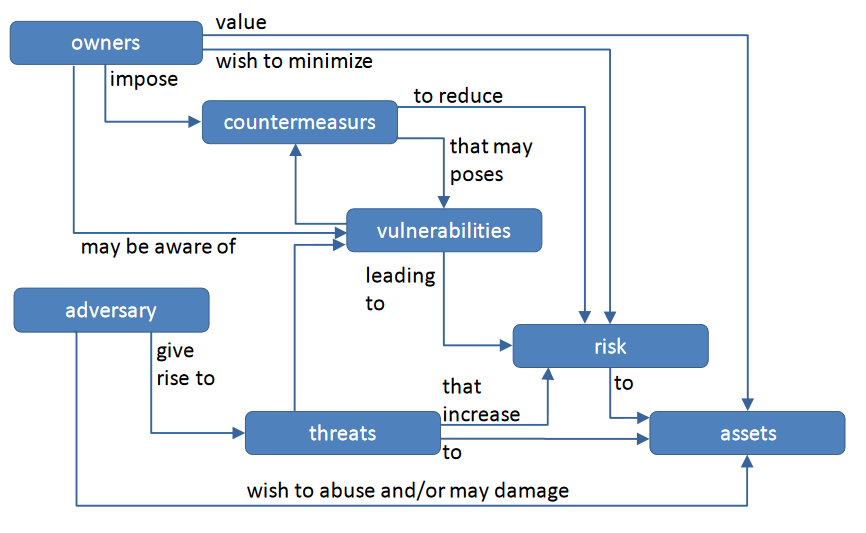
* Odnosi se na pouzdanost (trustworthiness) podataka ili resursa
  + Integritet podataka: osigurava da se informacije i programi mijenjaju samo u određenom i ovlaštenom načinu
  + Integritet sustava: osigurava da sustav obavlja svoju namijenjenu funkciju na slobodan način siguran od neovlaštenog manipuliranja
  + Integritet izvora (authentication): odnosi se na povjerenje u valjanost poruke i/ili izvor poruke
* Mehanizmi zaštite integriteta spadaju u dvije klase
  + Preventivni mehanizmi – blokira bilo koji neovlašten pokušaj promjene podataka ili mijenjanje podataka u neautoriziranom načinu (autentifikacija i kontrola pristupa)
  + Mehanizmi otkrivanja – ne pokušava spriječiti kršenje integriteta; jednostavno navode da integritet podataka nije više vjerodostojan ('crypto hash' i MAC funkcije, digitalni potpisi)

Dostupnost

* Odnosi se na sposobnost korištenja informacije ili željenog resursa
  + Nedostupnost sustava je barem jednako loša kao da ne postoji nikakav sustav
  + Aspekt raspoloživosti koji je relevantan za sigurnost jest činjenica da netko namjerno može onemogućiti pristup podacima ili uslugama, čineći ih nedostupnima
  + Denial-of-Service (DoS) napadi su pokušaji blokiranja dostupnosti (npr. SYN Flooding DoS attack)
* Pri dizajniranju/projektiranju novih sustava, ovaj aspekt je često zanemaren – vrlo loša praksa

Sigurnosna terminologija (RFC 2828)

* Sistemski resursi (sredstva koja se štite) – hardver, softver, podaci, komunikacija objekata i mreže
* Ranjivost (Vulnerability) – nedostatak ili slabost u dizajnu, implementaciji ili radu sustava koja se može iskoristiti za kršenje sigurnosne politike
* Sigurnosna politika (Security policy) – skup pravila koja navode što je dopušteno, a što nije
* Protivnik (Adversary) – subjekt koji napada ili prijeti sustavu
* Napad (Attack) – napad na sigurnost sustava od inteligentne strane
* Prijetnja (Threat) – potencijalno narušavanje sigurnosti (potencijalno iskorištava ranjivost)
* Rizik – očekivani gubitak izražen kao vjerojatnost da će određena prijetnja iskoristiti određenu ranjivost s određenim gubitkom
* Protumjera (Countermeasure) – akcija koja smanjuje prijetnju, ranjivost ili napad



Ranjivosti i napadi

* Slabosti sustava resursa (imovine)
  + Može biti oštećen (gubitak integriteta)
  + Postati propustan (gubitak povjerljivosti)
  + Postati nedostupan (gubitak dostupnosti)
* Napadi su prijetnje koje se obavljaju i mogu biti:
  + Pasivne (wiretapping, snooping)
  + Aktivne (Man-in-the-Middle, Man-in-the-Browser)
  + Unutrašnje (insider)
  + Vanjske (outsider)

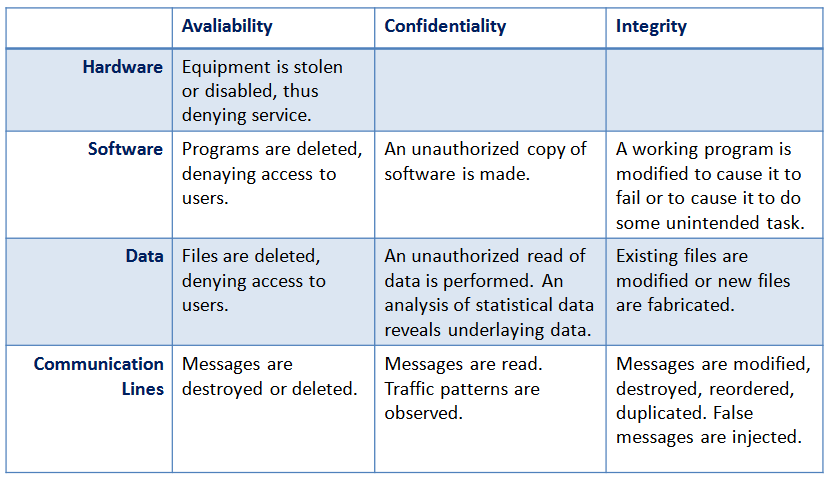
Man-in-the-Browser (MitB)

* Trojanski konj koji zarazi web preglednik i ima sposobnost mijenjanja stranice, sadržaja ili umetanje dodatnih transakcija
* SSL/PKI i/ili dva ili tri faktora autentičnosti ne pomažu
  + Jedini način protumjere na MitB napade je korištenje provjere transakcije

Prijetnje i napadi

* Neovlašteno objavljivanje – prijetnja na povjerljivost
  + Izloženost, presretanje, zaključci, provale
* Deception – prijetnja integritetu sustava i podataka
  + Zamaskiranje, falsificiranje, nepriznavanje
* Disruption – prijetnja dostupnosti ili cjelovitosti sustava
  + Onesposobljavanje (napad na dostupnost sustava), korupcija (integritet sustava), prepreke (smetnje u komunikaciji)
* Usurpation – prijetnja za integritet sustava
  + Misapproptiation (krađe usluga, tj. korištenje drugih uređaja za obavljanje DDoS), misuse (sigurnosne funkcije onemogućene od strane štetne logike ili hakera)

„Asset“ kategorije: hardver, softver, podaci i komunikacijske mreže



Strategije računalne sigurnosti

* Obuhvaća tri aspekta
  + Specifikacija/politika: Što bi sigurnost sustava trebala činiti?
    - Vrijednosti sredstava trebaju biti zastičene
    - Ranjivost sustava
    - Potencijalne prijetnje i vjerojatnost napada
    - Jednostavnost korištenja u odnosu na sigurnost
    - Troškovi osiguranja u odnosu na cijenu neuspjeha i oporavak
  + Provedbe/mehanizmi: Kako to učiniti?
    - Sprječavanje, otkrivanje, dogovor, oporavak
  + Ispravnost/jamstvo: Radi li uopće sigurnost?
    - Jamstvo – stupanj povjerenja da sustav ima sigurnosne mjere kakve smo zamislili
    - Testiranje

Računalna sigurnost pokušava osigurati povjerljivost, integritet i dostupnost sredstava računalnog sustava

Četiri važna načela:

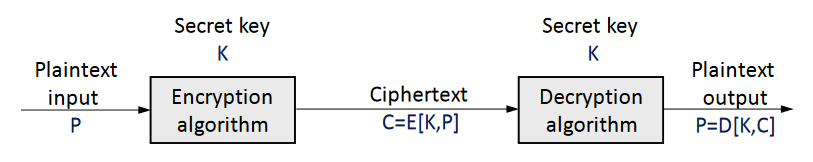
* Najlakši prodor – razmatra odjednom sve aspekte sustava
* Pravovremenost – sustav mora biti zaštićen protiv probojnosti onoliko dugo koliko proboj donosi neku korist onome tko ga vrši
* Efektivnost – mogućnost i korištenje kontrola zaštite
* Načelo najslabije karike – sigurnost je jaka koliko je jaka i najslabija točka

Protumjere (kontrole) se mogu primijeniti na razini podataka, programa, sustava, hardvera, komunikacijske veze, okoline i osoblja

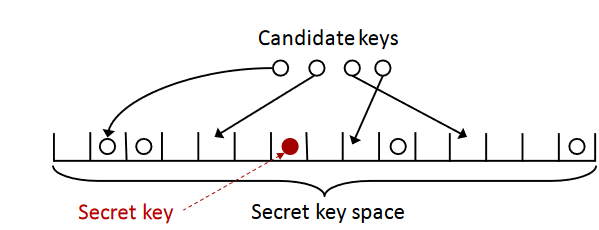
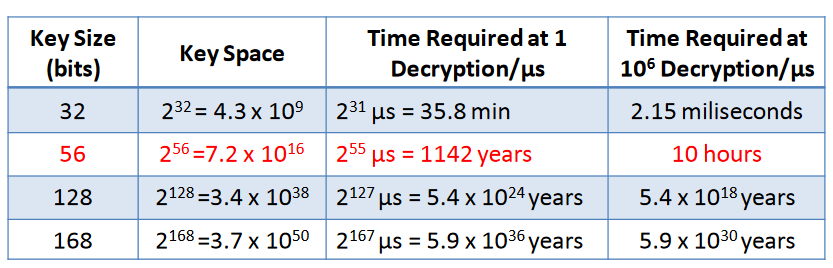
* Ponekad je potrebno nekoliko kontrola za pokrivanje jedne ranjivosti

Kriptografski algoritmi su važan element u provedbi sigurnosnih službi (npr. CIA)

Simetrična enkripcija

* Univerzalna tehnika za pružanje povjerljivosti za obje transmisije(komunikacije) i pohranjenih podataka (DBs, datoteke)
* Pet glavnih elemenata:
  + Plaintext (P) – izvorna poruka ili podatak koja je enkriptirana
  + Encryption algorithm (E[.]) – obavlja različite promjene (zamjene) na input plaintext
  + Secret key (K) – tajni ključ za enkriptiranje (algoritam za transformiranje plaintexta ovisi o njemu)
  + Ciphertext (C) – kodirana poruka proizvedena od šifrirane poruke (ovisi o plaintextu i secret key)
  + Decryption algorithm (D[.]) – uzima ciphertext i tajni ključ, i proizvodi originalni plaintext
* Dva uvjeta za sigurno korištenje:
  + Trebamo **jak algoritam šifriranja** (npr. napadač koji zna E[.] i C nije u mogućnosti dešifrirati C ili pronaći ključ K)
  + Pošiljatelj i primatelj moraju dobiti kopije tajnih ključeva na siguran način i moraju **držati ključ na sigurnome**

Napadač simetrične enkripcije:

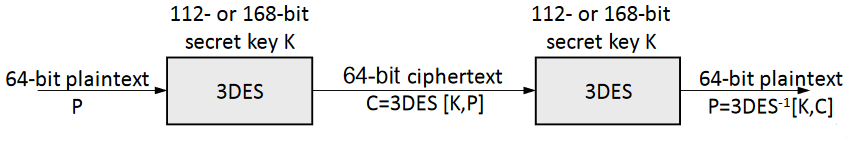
* Dva opća pristupa: kriptoanaliza i brute-force napad
* KRIPTOANALIZA
  + Iskoristiti prirodu algoritama šifriranja i opće značajke plaintext-ciphertext parova
  + Nastojanja da se nauče specifični plaintextovi ili tajni ključevi
  + Jednom kad je ključ ugrožen, sve buduće i prijašnje poruke enkriptirane tim ključem su također ugrožene
* BRUTE-FORCE ATTACK (izravan i jednostavan)
  + Nabavite jedan (plaintext, ciphertext) par
  + Isprobajte sve moguće kandidate za ključ K' i provjeriti da li je P == D[K',C]
  + U prosjeku polovica svih mogućih ključeva moraju biti isprobani

Data Encryption Standard (DES)

* Najraširenija korištena enkripcijska shema
  + Koristi 64 bit plaintext blok i 56 bit-ni ključ za izradu 64 bit ciphertext bloka
  + Prihvaćen 1977 od US 'National Institute od Standards and Technology' (NIST) kao standard
* Pokazao se nesigurnim, srpanj 1998 (brute-force attack)
  + Electronic Frontier Foundation namijenjen „DES cracker“ stroj vrijedan 250 000$
  + Otkrili su tajni ključ za 56 sati
  + 56 bitni enkripcijski ključ je prekratak!
  + Međutim, nikakve druge kobne slabosti nisu prijavljene za sada

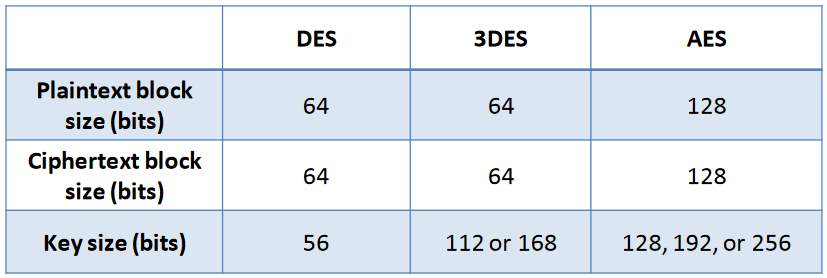
Triple DES (3DES)

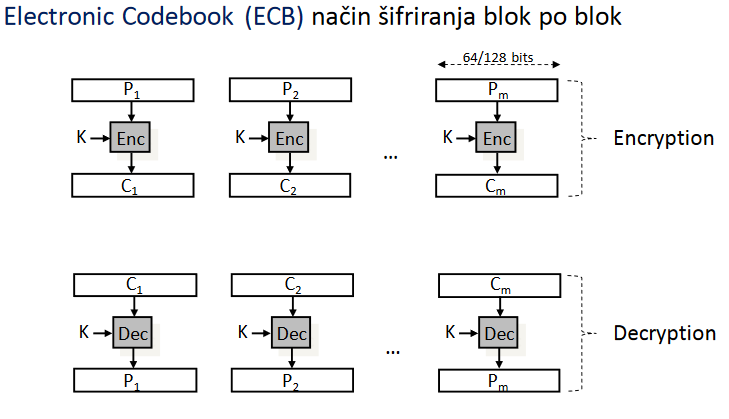
* Uključuje ponavljanje DES algoritma tri puta, koristeći dva ili tri jedinstvena tajna ključa
  + Tajni ključ 56 x 2 =122 or 56 x 3 =168 bita
  + Standardiziran za uporabu u financijskim aplikacijama 1985
  + 3DES sa 160-bitnim ključem je vrlo siguran i još uvijek u uporabi (npr. IPSec, Windows XP EFS)
* Nedostaci:
  + Neučinkovit, kada se provede u softver (učinkovit u hardveru)
  + DES i 3DES koriste kratku 64-bitnu veličinu bloka



Advanced Encryption Standard (AES)

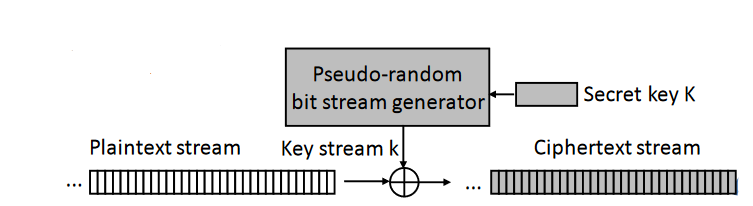
* 3DES nije dovoljno dobar kandidat za dugotrajno korištenje
* 1997. NIST je izdao novi AES sa zahtjevima
  + Jakost sigurnosti jednaka ili bolja od 3DES
  + Značajno više učinkovitija
  + Poveća duljina bloka (128 bita)
  + 'Support key' dužine 128, 192 i 256 bita
* U studenom 2001 NIST je odabrao Rijnael kao AES
* Danas se najviše koristi u komercijalnim uslugama



Sigurnosni problemi u praksi

* Obično, plaintext poruka P je veća   
  od 64 ili 128-bitnog bloka
* Postoje sigurniji načini: CBC, OFB, CFB, CTR…

Block vs. Stream Cipher

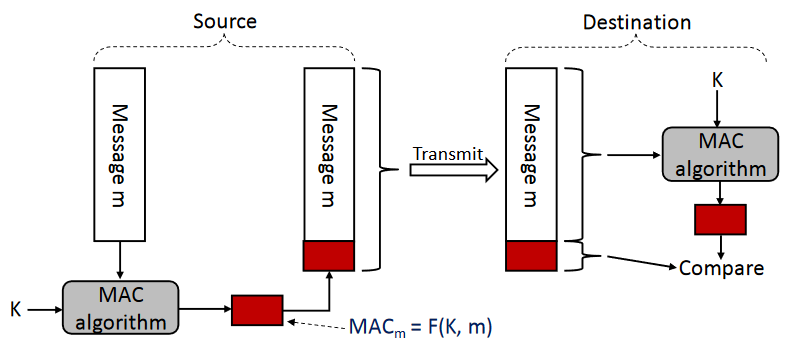
* 'Block cipher' procesi plaintext-ove unose u blokove fiksne veličine i proizvodi blokove ciphertext-a jednakih veličina za svaki blok plaintext-a
* 'Stream cipher' procesi unose elemente kontinuirano, stvarajući na izlaz jedan element u isto vrijeme  
  (Ci = Pi XOR ki (K) )  
  + Brza enkripcija i učinkovita
  + Ne smije ponavljati ključeve!

Autentičnost poruka

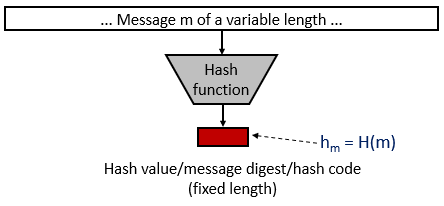
* enkripcija štiti protiv pasivnih napada (snooping-njuškanje)
* autentičnost poruka štiti protiv aktivnih napada (krivotvorenje podataka i transakcija)
  + potvrđivanjem da su primljene poruke autentične (nepromijenjene)
  + 'source/origin' poruke je autentičan
  + Također je moguće osiguranje isporuke poruka u vremenu i slijedu (1,2,3…)
* Opći pristup
  + Autentične oznake su generirane i dodane u svakoj poruci u prijenosu
  + Poruka sama po sebi ne mora biti šifrirana

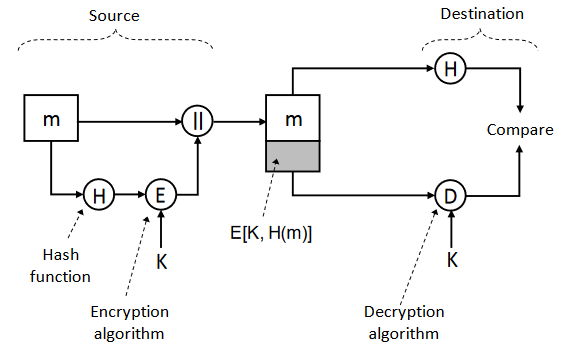
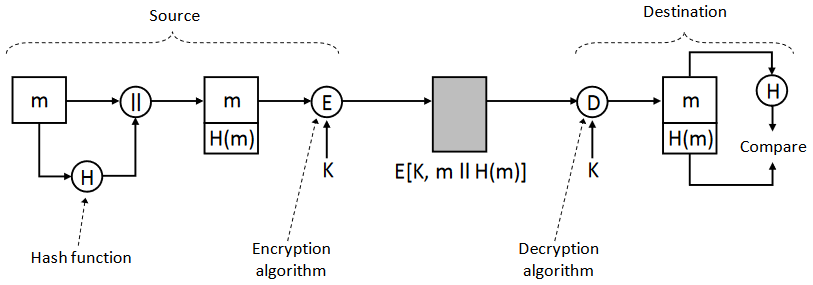
Kodovi autentičnosti poruka

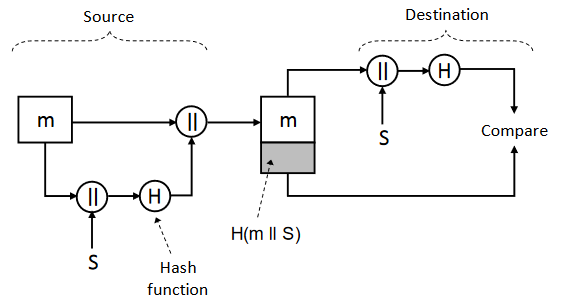
* Izvor i odredište dijele tajni ključ K
  + Izvor izračunava kod autentičnosti poruke (fiksne duljine) kao funkcija pruke m i zajedničkog ključa K: MACm = F(K, m)
  + Primatelj prima MACm i M i također računa MACm = F(K, m)
  + Ako su oba ista i ključ je tajan, primatelj prihvaća poruku m kao autentičnu



Sigurne (Kriptografske) Hash Funckije

* Kao i kod MAC, hash funkcija prihvaća promjenjive veličine m za unos poruka i proizvodi fiksnu veličinu sažete poruke (hash vrijednost)
* Za razliku od MAC, hash funkcija ne uzima tajni ključ kao ulazni
* Za autenticirat poruku, ovjerena hash vrijednost je poslana sa porukom
* Osiguravanje autentičnosti i povjerljivosti (desna slika)
  + H(.) detektira bilo kakvu promjenu u „hashed“ poruci m
  + Promjena u bilo kojem dijelu m rezultira u različitoj hash vrijednosti





* Autentičnost poruke bez enkripcije
  + H(.) općenito je mnogo učinkovitija (brža) od enkripcije
  + Pretpostavka je da izvor i odredište dijele tajnu S
* Hash funkcija H(.) se često koristi sa javnim kljulem na temelju kripto sustava za izradu digitalnih potpisa (npr. certifikati)

Zahtjevi Hash funkcija

* Proizvode jedinstven „fingerprint“ datoteke poruke ili drugog bloka podataka
* H(.) prihvaća unos poruke proizvoljne veličine
* H(.) proizvodi fiksnu duljinu jedinstvenih hash vrijednosti
* Jednostavno generiranje hash vrijednosti za svaku ulaznu poruku
* One-way property
  + S obzirom na hash vrijednost, računski je nemoguće pronaći m takav da H(m) = h
  + Gotovo je nemoguće pronaći poruku s obzirom na hash kod/vrijednost
* Weak.collision resistance
  + S obzirom na m, praktički je nemoguće pronaći takav m da H(m) = H(m)
* Strong-collision resistance
  + Praktički je nemoguće pronaći par (m, m) tako da H(m) = H(m)
  + Štiti protiv „birthday attack“

S obzirom na svojstva , hash vrijednost (sažeta poruka) može dokazati izvor i integritet poruke

Dva napada pristupa

* Kriptnoanaliza – iskoristiti logičke slabosti u hash funkciji (napad na SHA-1)
* Brute-force attack – pokušaj više ulaza (m,m) za otkriti koliziju, tj. H(m) = H(m)   
   – snaga proporcionalna veličini hash koda (2n/2, n-bit hash kod)

SHA najčešće korišteni hash algoritam

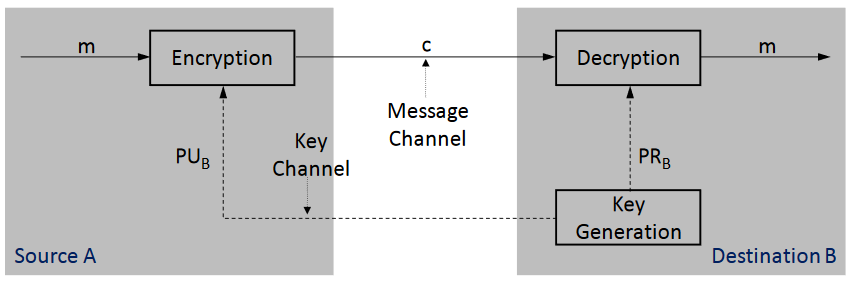
* SHA-1 daje 160-bit hash vrijednost (snage 280, nedavno 269!)
* Noviji SHA.384, SHA-512 pružaju poboljšanu veličinu i sigurnost
* Druge hash funkcije MD2, MD4 (nisu u upotrebi) i MD5 (nezaštićene, ali i dalje se koriste)

Neke primjene Hash Funkcija

* Password hashing
  + Pohranjivanje lozinke hash vrijednošću umjesto same lozinske
  + Napadač ne može obrnuti hash funkciju (jednosmjernost)
* Osigurava integritet dijela softvera
* Učinkoviti digitalni potpis (npr. digitalni certifikati=
* Commitment schemes – sometimes one needs to commit to a message without actually revealing it (npr. a bid in an auction)

Public-Key (Asimetričnost) šifriranje

* Enkripcijski i dekripcijski ključ nisu isti
* Javni ključ – na njega svi imaju pristup
* Privatni ključ – zna ga samo vlasnik
* Ako je jedan ključ upotrebljen za enkripciju, drugi je upotrebljen za dekripciju (oba mogu bit korištena za oboje)
* Duboke posljedice u području povjerljivosti, distribucije ključa i autentikacije



Public-Key Crypto: Confidentiality

Strana B generira par ključeva (PUB, PRB)

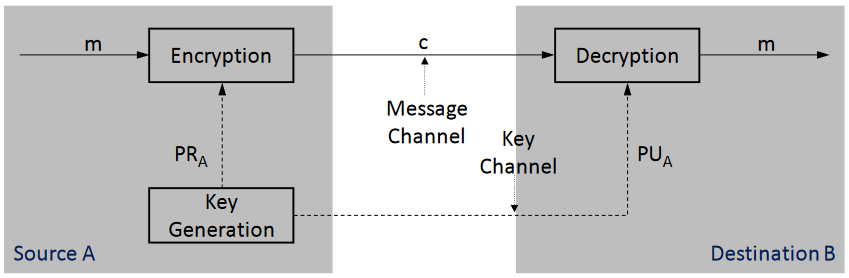
- PUB – B's javni ključ, PRB – B's privatni ključ

Zaštita povjerljivosti: A šalje tajnu poruku m strani B

* A kriptira poruku m javnim ključem PUB: c =[PUB, m]
* B dekriptira c koristeći njegov privatni ključ PRB: m=D[PRB, c] = D[PRB, E[PUB, m]]
* Nitko drugi ne može dekriptirat c (samo B ima PRB)

Public-Key Crypto: Integrity

Autentikacija i integritet: Objekt A želi poslati autentičnu poruku m objektu B

* A kriptira m koristeći njegov privatni ključ PRA : c = E[PRA, m]
* B dekriptira šifrirani tekst c koristeći javni ključ od A: PUA : m = D[PUA, c]
* Samo A zna PRA, tako da samo A može proizvesti valjani c – autentikacija izvora
* Cijela šifrirana poruka c služi kao digitalni potpis
* Nije moguće promijeniti m bez poznavanja privatnog ključa PRA, te je onda m također autentičan u smislu integriteta podataka

Public-Key Crypto: Confidentiality & Integrity

Povjerljivost i provjera autentičnosti: A želi poslati ovjerenu i tajnu poruku m strani B

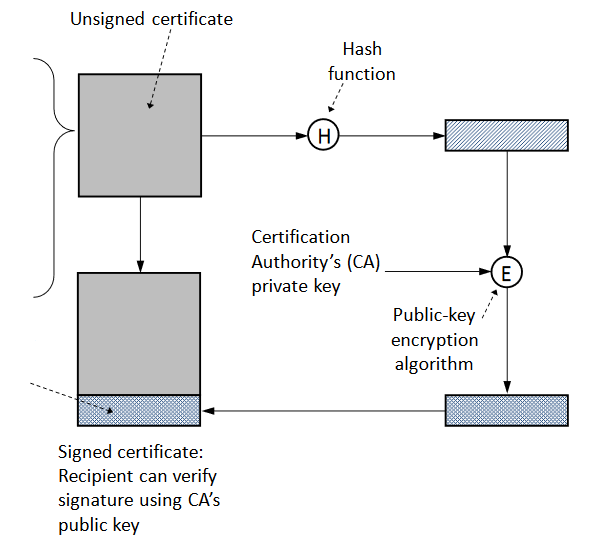
* Ovo možemo realizirati koristeći dva para (PU, PR) ključeva
* A kriptira m na sljedeći način: c = E[PRA, m], c' = E[PUB, E[PRA, m]]
* B dekriptira šifrirani tekst c na sljedeći način: c = D[PRB, c'], m = D[PUA, c]

Algoritmi tajnog ključa

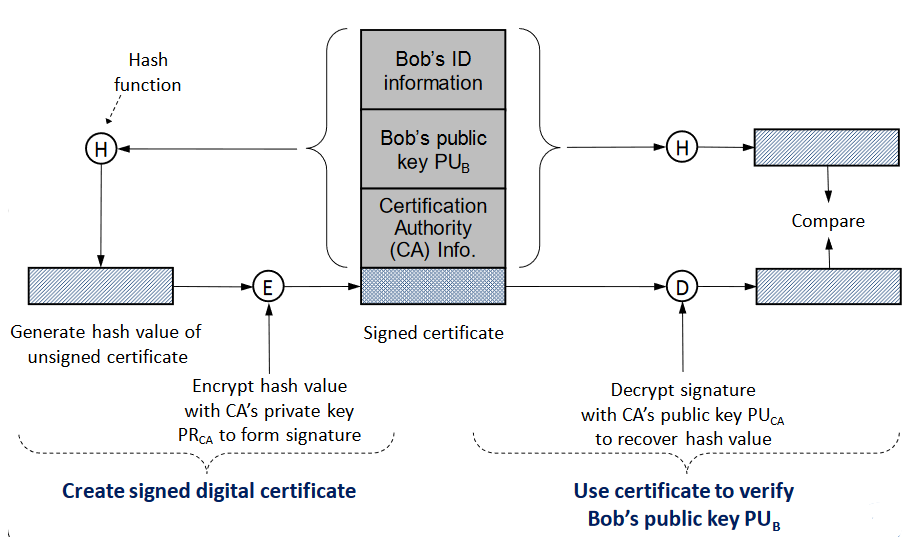
* RSA (Rivest, Shamir, Adleman)
  + Razvijen u 1977
  + Jedini široko prihvaćen algoritam za kriptiranje javnog ključa
  + Radi sigurnosti ključ ima veličinu > 1024-bit (300 decimalnih znamenki)
* Diffie-Hellman algoritam razmjene ključa
  + Omogućava sigurnu razmjenu tajnog ključa (bez enkripcije)
* Digital Signature Standard (DSS)
  + Pruža funkciju digitalnog potpisa sa hash funkcijom SHA-1
* Elliptic curve cryptography (ECC)
  + Novi, sigurnost kao RSA samo sa puno manjim ključevima

Certificiranje Javnog ključa

* Jedna od ključnih uloga šifriranja javnog ključa je adresirati problem distribucije ključa
  + Distribuvija javnih ključeva
  + Korištenje enkripcije javnog ključa za distribuciju tajnih ključeva
* Javni ključ je javan, ali „kako znamo da javni ključ pripada određenom korisniku“
  + To je riješeno pomoću certifikata autoriteta (CA) – vlade ili financijske ustanove
  + CA djeluje kao pouzdana treća strana i izdaje certifikate javnog ključa, koji se sastoje od javnog ključa i identifikatora korisnika (User ID), svi potpisani od strane CA (koristeći CA-ov javni ključ)
* Korisnik može objaviti njegov certifikat (npr. na internetu)
  + Svatko može provjeriti njegov javni ključ služeći se pouzdanim potpisom (trusted signature)

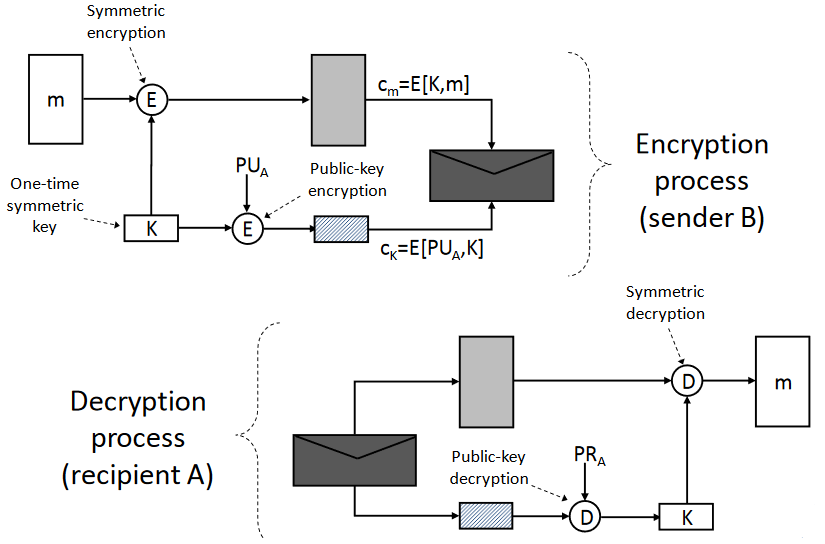


* Struktura certifikata
  + Javni ključ
  + Javni vlasnikov ključ (User ID)
  + Certifikat izdavatelja
  + Datum izdavanja
  + Valjanost certifikata
  + Druge info. (vrsta, standardi, …)
  + Digitalni potpis izdavatelja certifikata
* X.509 standard
  + IPSec (mreža)
  + SSL (web)
  + S/MIME (email)…



Digitalne omotnice (Envelopes)

* Korištenje javnog ključa za šifriranje za distribuciju tajnih ključeva
* Tajni ključevi su učinkovitiji i brži
* B želi slati povjerljive poruke m prema A
  + A i B ne dijele nikakav simetrični ključ
  + B drži autentični javni ključ PUA od A
  + B priprema poruku m i proizvodi jednokratni (sesijski) simetrični ključ K
  + B enkriptira m: Cm = E[K,m]
  + B enkriptira dio ključa K koristeći A's PUA : cK = E[PUA, K]
  + B formira digitalnu omotnicu Env = (cm, cK) i šalje prema A
  + Samo A je u mogućnosti dekriptirati dio ključa od cK i stoga obnaclja izvornu poruku m od cm
* Primjer: Windows XP Encrypting File System (EFS)

Slučajni brojevi

Igraju važnu ulogu u korištenju enkripcija.

Koriste se u generaciji.

* Ključevi se koriste u simetričnoj i public-key   
  enkripciji
* Stream ključevi u stream cipheru (?)
* Provjera autentičnosti protokola
* Osigurava tokene…

Zahtjevi

* Slučajnost – na temelju statističkih testova za ravnomjernost i neovisnost
* Neprevidivost – suksesivne vrijednosti koje nisu povezane sa prethodnim  
  (zahtjeva neki hardver generator, nije uvijek u upotrebi)

Pseudoslučajni i Slučajni brojevi

* Često korištena algoritamska tehnika za stvaranje pseudoslučajnih brojeva
  + Koji zadovoljavaju statističke testove slučajnosti
  + No, vjerojatno će biti predvidljivi
  + Npr. šifrirati dani slučajni broj n puta s tajnim kljulem
* Pravi slučajni generatori brojeva koriste nedeterministički izvor
  + Npr. zračenje, plinsko pražnjenje, curenje kondenzatora (prirodni procesi i fenomeni)
  + Sve više se pruža modernim procesorima

Autentičnost korisnika

* Zaštitne mjere (npr. kontrola pristupa, odgovornost) imaju smisla samo ako možemo identificirati i autentificirati korisnika
* Autentifikacija potvrđuje identitet korisnika
  + Češće kao preduvjet da omogući pristup resursima sustava
* Proces autentifikacije se sastoji od dva koraka:
  + Identifikacijski korak – predstavljanje identifikatora sustava (npr. userID, username)
  + Verifikacijski (autentifikacijski) korak – predstavljanje ili generiranje autentičnosti informacija koje povezuju subjekt koji predstavlja identifikator i identifikator.
* Različičt od autentičnosti poruke

Načini autentifikacije

* Nekakvo korisnikovo znanje – lozinska, osobni identifikacijski broj (PIN)
* Nešto što korisnik posjeduje – smart kartice, fizički ključevi, tokeni
* Nešto što korisnik jest (statička biometrija) – prepoznavanje otiskom prsta, lica, mrežnice, iris-om
* Nešto što korisnik čini (dinamička biometrija) – prepoznavanje glasa, rukopisa, ritam tipkanja

Može se koristiti u kombinaciji.

Svi oni imaju i prednosti i mane.

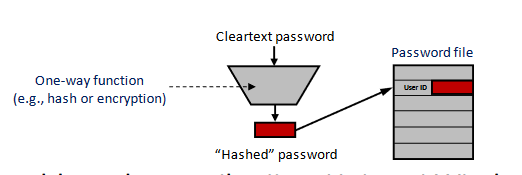
Autentičnost lozinke

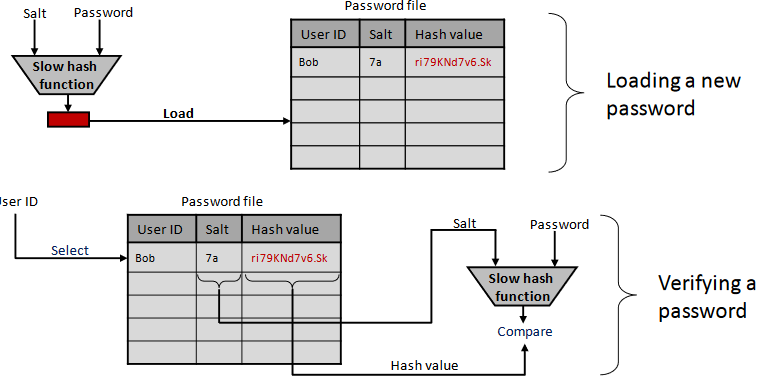
* Najkorištenija metoda autentikacije korisnika
  + Korisnik daje ime/login (korisničko ime) i lozinku
  + Sustav uspoređuje lozinku sa spremljenom za određenu prijavu
* Autentificira ID korisnika prijavom i daje sigurnost:
  + Utvrđivanjem da je korisnički ID ovlašten za pristup sustavu
  + Određuje korisnikove povlastice (npr. admin ili ne)
  + Koristi se u diskrecijskoj kontroli pristupa (npr. korisnik koji posjeduje datoteku može omogućiti drugom subjektu za pristup ovoj datoteci)

Ranjivost lozinke

* OFFLINE DICTIONARY ATTACK
  + Napadač dobavlja datoteku lozinke (sa hash lozinkom) i uspoređuje hash lozinku sa hash lozinkom iz 'dictionary'
* SPECIFIC ACCOUNT ATTACK
  + Unošenje kandidata lozinke sve dok točna lozinka nije otkrivena ili dok korisnički račun nije zaključan (npr. nakon 3 neuspjela pokušaja)
* POPULAR PASSWORD ATTACK
  + Pokušaj 'popularne' lozinke protiv raspona korisničkih ID-a
* POGAĐANJE LOZINKI PROTIV JEDNOG KORISNIKA
  + Napraviti 'educated guess-ove' temeljene na znanju o korisniku (dob, spol, bračni status …)
* WORKSTATION OTMICE (hijacking)
  + Ukrasti ključ 'radne stanice' i koristiti npr. Cain&Abel za otkrivanje lozinke
* ISKORIŠTAVANJE VIŠESTRUKO KORIŠTENJE ISTE LOZINKE
  + Problem višestrukog korištenja lozinki (zbog teškoće pamćenja)
* ELEKTRONSKI NADZOR (monitoring)
  + Presresti lozinke priopćene preko mreže (jednostavna i naivna enkripcija ovdje ne pomaže)

Spremanje lozinki

* Lozinke nikad nisu spremljene u 'clear text' – veliki rizik od krađe
* Hash lozinka se pohranjuje
  + Hashing je jednosmjerna funkcija koja daje jedinstveni neinverzni rezultat (hash vrijednost, sažeta poruka)
  + Ako korisnik daje točnu lozinku, njegov hash mora biti identičan spremljenome hash-u (prethodnom) u datoteci lozinka

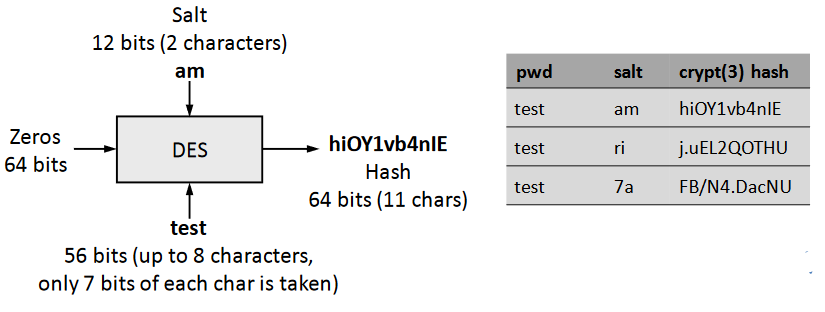
Shema Unix Lozinki

* za učitavanje (kreiranje) nove lozinke u sustav:
  + korisnik odabere ili dodijeli lozinku
  + ova lozinka je u kombinaciji   
    sa fiksnom duljinom 'salt'

'Salt' vrijednosti

* offline dictionary attacl
  + pretpostavka: cilj je pogoditi jedan pwd (bez salta)
  + napadač dobije kopiju datoteke lozinki
  + napadač hash-ira vjerojatnije lozinke i uspoređuje dobivene hash vrijednosti sa onima u datoteci lozinki
  + ako bilo koji od pretpostavljenih odgovara hash-u u datoteci, napadač je pronašao lozinku koja je u datoteci
* 'salt' vrijednost ima 3 svrhe:
  + Sprječava duplikat lozinka da bude vidljiv u datoteci lozinki
  + Povećava poteškoće offline dictionary napada (k bitova 'soli' povećava opterećenje nagađanja za faktor ̴2k)
  + Nije moguće saznati da li korisnik ima iste lozinke na dva ili više sustav

Unix Hashed Pwd implementacija

* Izvorna shema (crypt(3) routine)
  + 8 znakova lozinke u obliku 56-bitnog tajnog ključa
  + 12-bitna 'sol' remeti DES algoritam ekripcije u jednom od 4096 različitih načina
  + 0 vrijednost u više navrata šifrirana 25 puta (usporava nagađanja)
  + Output preveden u 11 znakova u nizu
* Crypt(3)- based implementation is inadequate today
  + 8 znakova (56-bitova) je jednostavno premalo
  + Dictionary attack istražen koristeći Blue Horizon superračunalo
    - Unaprijed izračunato i pohranjeno 207 milijardi hash-eva (otp 1.5 TB) za preko 50 miliona lozinki (sa saltom) za oko 80 min
* Time-memory tradeoffs
  + Učinkovito kad se 'sol' ne koristi
* Bolji hash-evi za Unix
  + Moderni Unix sustavi temeljeni na MD5 hash umjesto DES hash
  + Prednosti:
    - Lozinke mogu imati više od 8 znakova
    - Izrađuje 128 bitne hash vrijednosti
    - Dulje salt vrijednosti (48 bits)
    - „vrlo“ spor (1000 unutarnjih petlji)

Kontrola pristupa datoteci (Unix)

* Stari način: imena i hash-evi se pohranjuju u /etc/passwd
  + Čitanje slobodno za svakoga
  + Otvara se za jednostavne offline distionary napade
* Sigurniji način: hash-evi pohranjeni u zasebnoj datoteci /etc/shadow
  + Samo root može pristupiti na ovu datoteku
* Krađa Unix hash-a
  + Cilj: pristup /etc/shadow
  + Boot uređaj na CD-u
  + Dobiti root povlastice (npr. koristeći propust)

Prijava i Autentičnost u Windows-u

* Postoje četiri vrste prijavnih procesa u sustavu Windows 2000, Windows Server 2003/08 i Windows XP Professional
  + Interaktivne prijave (logon)
    - Prijave na lokalna računala na koje imaju izravan fizički pristup (uključuje terminal usluga i 'Remote Desktop' prijavni proces)
  + Mrežne prijave
    - Kontrole pristupa sustavu s različitim operativnim Windows sustavima preko mreže s računala na kojem se prijavljujemo
  + Usluge (sevice)
    - Autentikacija i autorizacija različitih Windows servisa
  + Batch
    - Rezervirano za 'seriju' radnih mjesta (npr. big print spools, bank account reconciliation); rijetko se koriste

Local Security Authority (LSA)

* Windows sigursnosni podsustav koji autentificira i prijavljuje korisnike na lokalna računala
  + Upravlja lokalnom sigurnosnom politikom (koje domene su pouzdane, koje mogu imati pristup sustavu, koja su prava dodijeljena)
  + Pruža interaktivnu uslugu autetifikacije korisnika
  + Generira tokene za pristup
  + Upravlja politikom revizije

Kerberos protocol je temeljni:

* Od uvođenja Windows 2000
* Omogućuje uzajamnu autentifikaciju klijenta (korisnika, računala, usluge) i poslužitelja
* Siguran i skalabilan
* LSA djeluje kao proxy između klijenta i Kerberos Key Distribution Service (KDS) koji izdaje tickete za servis klijentu

NTLM: LM Hash

* LM autentifikacija utemeljena na slabim kriptografskim postupcima
  + Korisnička ASCII lozinka konvertirana u velika slova
  + Lozinka null-padded ili skraćena do veličine od 14 bajtova
  + Lozinka „fikse duljine“ je podijeljena u dvije 7-bitne polovice
  + Ove dvije vrijednosti se koriste za stvaranje dva DES ključa
  + Jedna od svake 7-bitne polovice, konvertiranjem (pretvaranjem) sedam bajtova u bit stream (tok bitova), i umetanjem nul bita nakon svakih sedam bita ; ovo generira 64 bita potrebnih za DES ključ  
    - Svaki od ovih ključeva se koristi za šifriranje (enkripciju) DES-konstanti ASCII string KGS !@#$%, što je rezultiralo dvijema 8 bytnim šifriranim vrijednostima
    - Ove dvije šifrirane vrijednosti su ulančane u obliku 16-byte vrijednosti, koja je LM hash
* Rezultirajući LM hash pohranjen u SAM (Security Accounts Manager) bazu podataka (lokalno ili na kontroler domene)

Nesigurnosti

* Znakovi lozinke su ograničeni na ANSI skup znakova za ispis (95 znakova)
* Lozinke dulje od 7 znakova su podijeljene u dva dijela i svaki dio je hashan posebno (skraćeni do 14 bajtova)
* Brute-force complexity (složenost)
  + Postoje 292 lozinki sa 14 ispisivih znakova
  + Ali samo 246 lozinki sa 7 ispisivih znakova (jedna polovica)
  + Samo velika slova koriste 243 (moguće probiti u nekoliko sati!)
* LM hash ne uključuje sol, dakle time-memory-tradeoff cryptoanalasys attack, poput rainbow tablice, su također izvedive (u nekoliko sekundi probijaju LM hasheve)

Challenge-response authentication (NTLMv1)

* Kontroler domene (server) šalje random challenge (slučajan izazov) na radnu stanicu korisnika
* Radna stanica šifrira (envrypts) izazov sa NTLM hash-om lozinke i šalje serveru
* Server obavlja istu operaciju koristeći NTLM hash pohranjen na lokalnoj razini i uspoređuje rezultate
* NTLMv1 Response = DES(Key1, Challenge) + DES(Key2, Challenge) + DES(Key3, Challenge)
* Izloženost je katastrofalna
  + Koristi se slaba DES šifra (duže lozinke je lakše bruteforceat
  + Aktivan man-in-the-middle lakše pokreće dictionary napad

Security Tokens

* Predmeti koje korisnik posjeduje u svrhu provjere identiteta korisnika
* Sustav može potvrditi da korisnik posjeduje jedinstven token
* Dva-faktorska provjera autentičnosti
  + Npr. kombinacija lozinke i tokena znatno povećava sigurnost
  + Primjeri: Scratch liste, SecureID tokeni (novi 6-znam. Broj svake minute), kalkulatori, magnetske i smart kartice

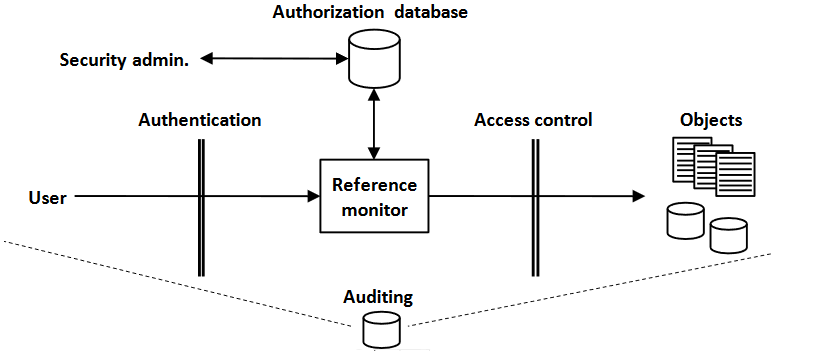
Biometrija

* Znanost mjerenja fizikalnih svojstava ljudskog bića
* Biometrički sistem za provjeru autentičnosti pokušava identificirati pojedinca na temelju jedinstvene fizičke karakteristike
* Ljudske fizičke veličine: otisak prsta, oblik ruke, iris, retina, DNA, lice
* Ljudsko ponašanje: dinamika potpisa (brzina, smjer, pritisak), glas, korištenje tipkovnice
* Biometrijski sistem
  + Fizička svojstva korisnika mapirana u digitalni prikaz
  + Za svakog korisnika jedan digitalni prikaz (predložak) pohranjen u računalu
  + Kada se korisnik identificira, sistem uspoređuje pohranjeni predložak sa predstavljenim predloškom
  + S obzirom na kompleksnost fizičkih karakteristika, nemožemo očekivati točno posudaranje između dva predloška
  + Sistem koristi algoritme za procjenu odgovarajućeg rezultata podudaranja (nppr. Izražava kao jedan broj)
* Teško je stvoriti savršeni biometrijski sustav
  + Preosjetljiv – previše lažnih negativa
  + Neosjetljiv – previše lažnih pozitiva
* Mane
  + Informacija nikad nije identična – nije moguće hash-iranje
  + Rizik od krađe
  + Nije moguće promijeniti ukraden prst
  + Neki skeneri mogu biti prevareni ili zamijenjeni
* Idealne aplikacije
  + Kontrolirani fizički pristup
  + Autentifikacija za plaćanja na blagajni

Kontrola pristupa

* ograničenja koja kažu što korisnik može učiniti direktno, kao i što programi koje pokreće korisnik mogu napraviti s ciljem:
  + spriječiti aktivnosti koje bi mogle povrijediti sigurnost
  + zaštititi od malicioznih prijetnji reguliranjem čitanja, pisanja i izvođenja podataka i programa
* temelji se i koegzistira s drugim sigurnosnim aspektima:
  + identifikaciju korisnika i autentifikaciju
  + zaštitu pohranjenih prava pristupa – autorizacijske informacije koje reguliraju tko može što raditi
* kontrola pristupa je centralni element računalne sigurnosti

Načela kontrole pristupa (Access Control - AC)

* kontrola pristupa je ojačana reference monitorom koji je posrednik u svakom pokušaju pristupa korisnika/proces objektima sustava
* reference monitor se konzultira sa bazom podataka koja sadrži prava pristupa/autorizaciju kako bi provjerio je li dozvoljeno izvođenje neke operacije od strane korisnika
* Sustav prvo mora autentificirati korisnika koji traži pristup
* Zatim, funkcija kontrole pristupa određuje je li traženi pristup dozvoljen tom korisniku
* Sigurnosni administrator održava autorizacijsku bazu podataka
* Funkcija kontrole pristupa se konzultira s tom bazom kako bi odlučila smije li dati pristup određenom korisniku
* Funkcija nadgledanja prati i održava zapis svih korisničkih pristupa resursima sustava (odgovornost, nedostaci)
* Svi operativni sustavi imaju kontrolu pristupa
  + Sustav upravljanja bazama podataka također uključuje funkciju kontrole pristupa

Politike kontrole pristupa

* Diskrecijska kontrola pristupa (DAC)
  + Sigurnosna politika orijentirana prema korisniku (bazira se na identitetu korisnika koji traži pristup)
  + Entitet može omogućiti drugom entitetu pristup resursu
* Obavezna kontrola pristupa (MAC)
  + Sustav definira kontrolu pristupa
  + MAC je baziran na usporedbi sigurnosnih oznaka resursa sustava (top security, low security) sa sigurnosnim odobrenjima entitea koji pristupaju resursu
  + Jedan enitet ne može dati prava pristupa drugom entitetu
* Kontrola pristupa temeljena na ulogama (RBAC)
  + Bazirana na ulogama koje korisnici imaju unutar sustava i na pravilima koja kažu koja prava pristupa su dozvoljena korisnicima s danim ulogama

Elementi kontrole pristupa

Subjekt – entitet koji pristupa objektima

* + To je proces koji predstavlja korisnika/aplikaciju
  + Najčešće sadrži 3 klase: vlasnik, grupa, svijet (ostali)

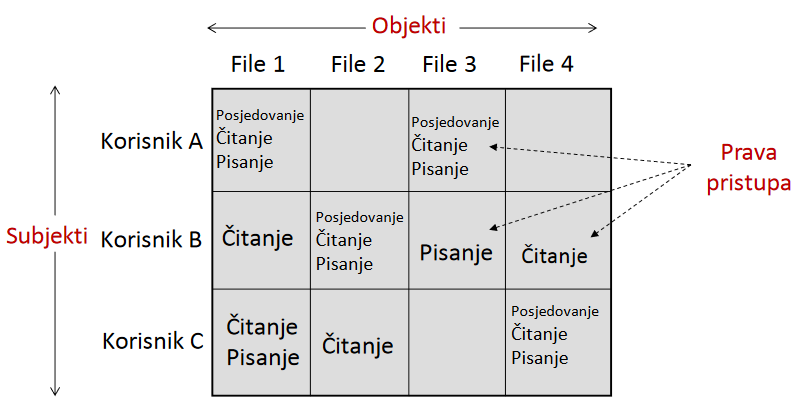
Objekt – resurs koji podliježe kontroli pristupa

* + Npr. datotekem direktorij, zapisi, programi, segmenti memorije, stranice, stabla direktorija, pretinci e-pošte itd.

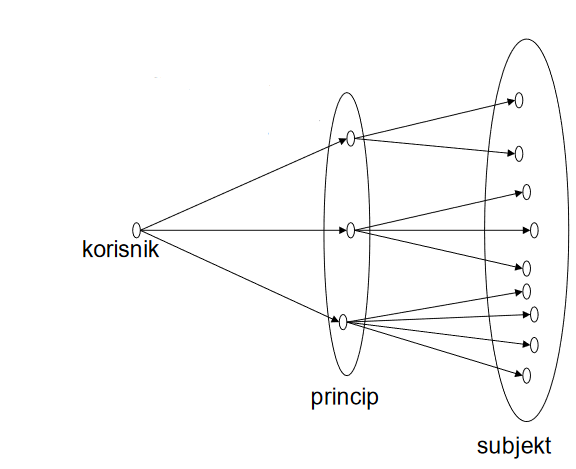
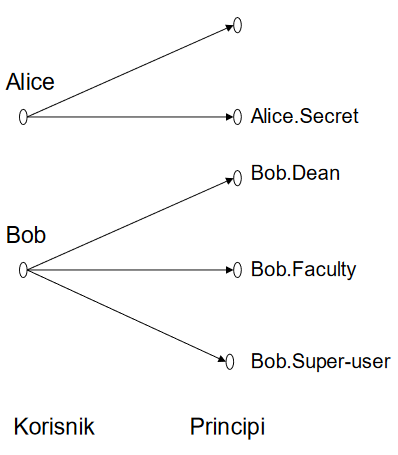
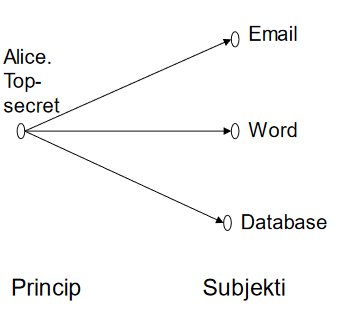
Pravo pristupa – način na koji subjekt pristupa objektu

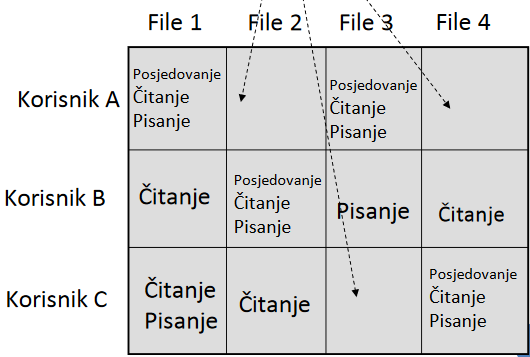
* + Npr. čitanje, pisanje, izvođenje, brisanje, kreiranje, traženje

**Diskrecijska kontrola pristupa (DAC)**

* Sigurnosna politika orijentirana prema korisniku (bazirana na identitetu(ID) korisnika)
* **Diskrecijska** jer entitet ima prava omogućiti drugom entitetu pristup resursima
* Opći pristup koji se koristi u operacijskim sustavima i sustavima za upravljanje bazama podataka.  
  ovaj pristup se prikazuje **pristupnim matricama (access matrix)**:
  + Lista subjekata – prva dimenzija (retci)
  + Lista objekata – druga dimenzija (stupci)
  + Svako polje matrice određuje prava pristupa   
    određenom objektu od stane određenog subjekta

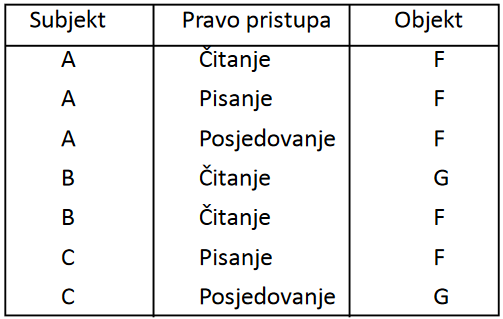
Elementi pristupne matrice:

* Korisnik – stvarna, fizička osoba
* Načelo (princip) – jedinica kontrole pristupa i autorizacije  
  + Mapiranje „1 prema mnogo“ između korisnika i načela
  + Sustav autentificira korisnika kao načelo
  + Djeljenje načela nije dobro zbog odgovornosti
* Subjekt
  + „1 prema mnogo“ između načela i subjekta
  + Program pokrenut u ime načela
  + Često se smatraju principom ako svi subjekti tog principa imaju ista prava
* Objekti
  + Bilo što na čemu subjekt može vršiti operaciju (posredovan prema pravima pristupa)
  + Općenito su pasivni (npr. datoteka, direktorij ili mapa, memorijski segmenti)
  + Međutim, subjekti mogu biti i objekti sa operacijama (Uništi [Kill], Zaustavi [Suspend], Nastavi [Resume])
* Prava
  + Pravo specificira kakav pristup subjekt može vršiti nad objektom
    - Own, read, write, execite, create, delete, transfer, …



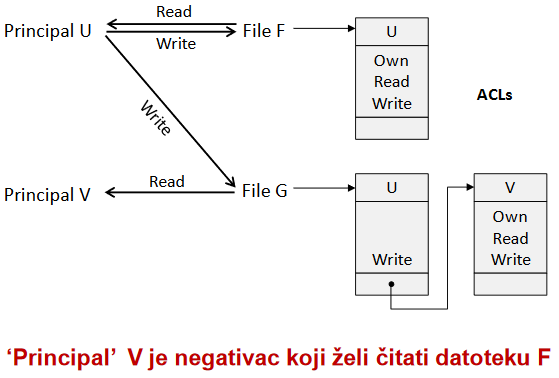
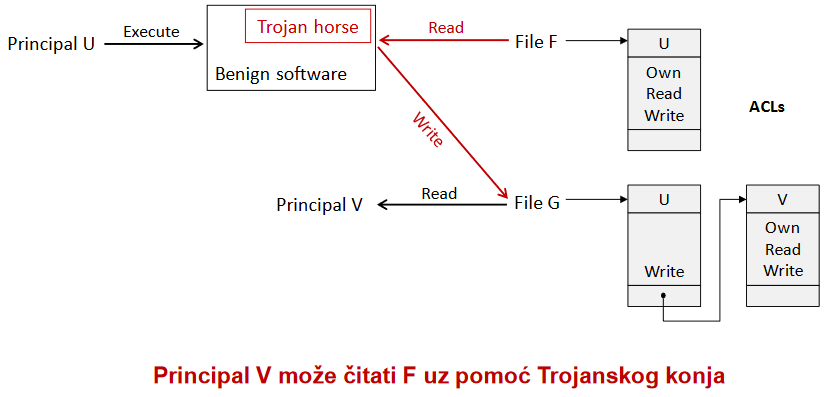
Implementacija pristupne matrice

* U praksi je obično rijetka
* Implementira se razgradnjom po:
  + Stupcima – access control lists
  + Retcima – capability tickets
* Access control lists
  + Prava pristupa su spremljena s objektima
  + Mogu sadržavati i unaprijed zadana (default) prava (elementi u ACL mogu biti pojedinačni korisnici ili grupe)
  + Pogodni su kada je potrebno odrediti koji subjekti imaju koja pristupna prava za neki objekt  
    (nisu pogodna kada trebamo naći sva prava za sve objekte za neki određeni subjekt)
  + Windows i Unix ga koriste za zaštitu podataka/procesa
  + Zahtjeva autentifikaciju subjekta prije pristupa pojedinačnom objektu!
* Capabilities
  + Prava pristupa su spremljena uz subjekte
  + 'Ticketi' mogućnosti specificiraju ovlaštene objekte i operacije za određene subjekte
  + Lako je definirati skup pravila za određenog korisnika   
    (teško je utvrditi popis korisnika koji imaju određeno pravo pristupa određenom resursu)
  + Svaki korisnik može posjedovati više tiketa
    - Korisnik može biti ovlašten za izdavanje tiketa drugim korisnicima
    - Ticketi se mogu rasprđiti diljem sustava što predstavlja veliki sigurnosni problem
    - Nemoguće krivotvorenje – uključuju 'unforgable' kripto tokene, odnosno autentifikacijske kodove (koriste se u distribuiranim sustavima, npr. Kerberos)
  + Zahtjevaju 'unforgability' i kontrolu širenja prava pristupa (kontroliranje ticketa)!

Autorizacijska tablica

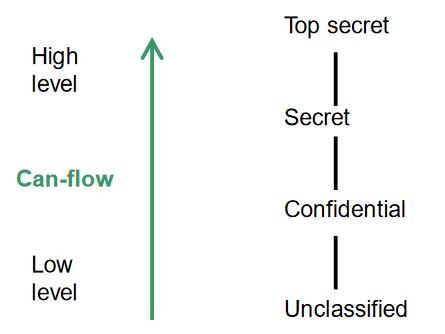
* Struktura podataka koja nije rijetka
* Puno je prikladnija od ACL i Capability lists
  + Sortiranje po subjektu i objektu
* Obično se koristi u relacijski sustavima  
  za upravljanje bazama podataka

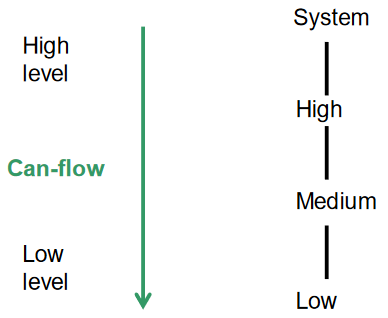
Sigurnosni problemi DAC

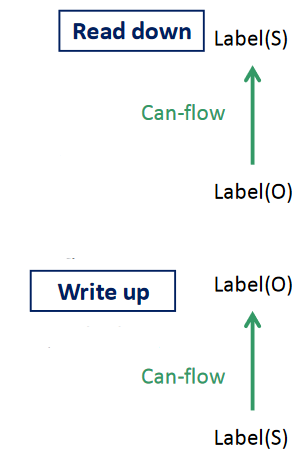
* Ne osigurava stvarnu sigurnost – ograničenja pristupa se mogu lako zaobići (npr. Trojanski konj)
* Principal V šalje U ' dobroćudni' softver sa Trojnskim konjem
* U izvršava softver -> Trojanski konj uzima privilegije od U i predaje ih principalu V (diskrecija)
* Rješenje problema DAC sigurnosti – obavezna kontrola pristupa (MAC)

**Obvezna Kontrola pristupa (MAC)**

* Dodaje sigurnosne oznake subjektima (security clearance) i objektima (security classification)
* Sustavne kontrole daju pristup resursima uspoređujući oznake resursa (npr. sustav, visoka, niska sigurnost) s oznakama subjekta koji pristupa
* Korisnici ne mogu kontrolirati oznake (kao kod DAC)
* Ograničava tijek informacija na određene can-flow puteve (ovisno o politici protoka informacija)

Kontroliranje toka informacija

* Sigurnost
  + 'vojne sigurnosne klase' (sa desne strane)
  + Ako je razina subjekta jednaka ili veća od   
    razine objekta, subjektu je dopušteno   
    čitati objekt (read down)
  + Imati na umu da subjekt može vršiti write up
* Integritet
  + Windows Vista Mandatory Integrity Control   
    (MIC) definira 4 razine integriteta: low,   
    medium, high i system
  + Ako je razina subjekta jednaka ili veća od   
    razine objekta, subjektu je dopušteno pisati   
    ili brisati objekt (write down)
  + Inače može samo čitati ako to dopusti ACL (read up)

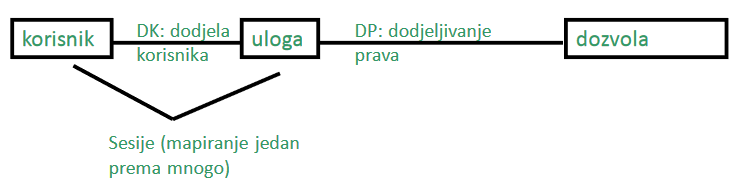
Bell and LaPadula model (BLP)

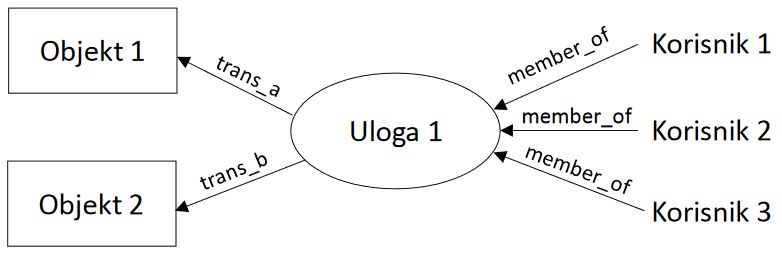
* Formalni MAC model za zaštitu tajnosti
* Općenito, baziran na principu read down i write up
* Štiti protiv Trojanskog napada u kontekstu DAC-a
* Jednostavno sigurnosno svojstvo
  + Subjekt S može čitati objekt O samo:
    - Ako Label(S) dominira (>=) Label(O)
    - Informacije se kreću od Label(O) prema Label(S)
  + Star-property (sprječava curenje inf. Od trojanskog konja)
    - Subjekt S može pisati po objektu O samo ako:
      * Label(O) dominira (>=) Label(S)
      * Informacije se kreću od Label(S) prema Label(O)
* BLP model se primjenjuje na subjekte, ne na korisnike
  + Korisnici su pouzdani
  + Subjekti nisu zbog Trojanskih konja

**Role-Based Access Control (RBAC)**

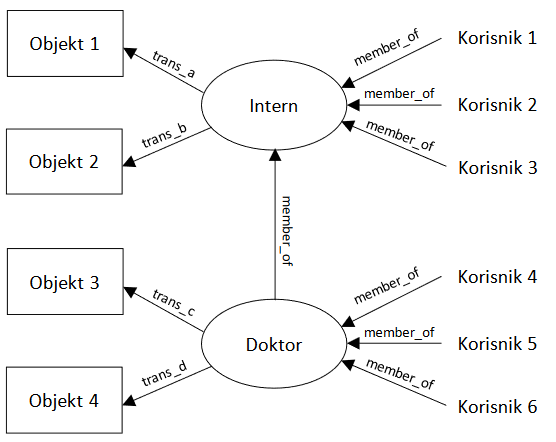
* Tradicionalni DAC sustavi definiraju prava pristupa pojedinih korisnika i grupa korisnika
* U mnogim organizacijama (u industriji), korisnik ne posjeduje informacije kojima može pristupiti.  
  Vlasnik je korporacija
  + Kontrola pristupa često se temelji na poslovnim funkcijama zaposlenika (uloge), a ne o vlasništvu podataka (npr. u bolnici: doktor, medicinska sestra, farmaceuti…)
* RBAC je baziran na ulogama koje korisnici vrše u organizaciji/tvrtci (umjesto identiteta korisnika)
  + RBAC sustavi pridodaju prava pristupa ulogama, a korisnicima dodijeljuju različite uloge

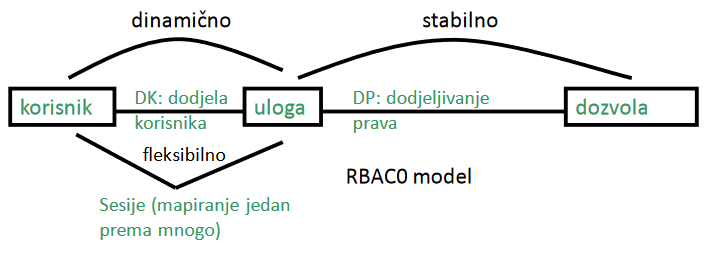
Uloge

* Predstavljaju korisnike
  + Sposobnost obavljanja zadaća
  + Odgovornost posla
  + Dodjela specifične zadaće
* Definiraju dozvole
  + Uloga operatora
  + Uloga 'Sigurnosnog službenika'
  + Uloga rezivora (auditor)

Korisnici, uloge i sredstva

* Odnos između korisnika i uloge jest   
  'mnogo prema mnogo'
* Odnos između uloge i sredstava ili sustava   
  objekta je također 'mnogo prema mnogo'

Hijerarhijske uloge – uloge mogu biti sastavljene od uloga



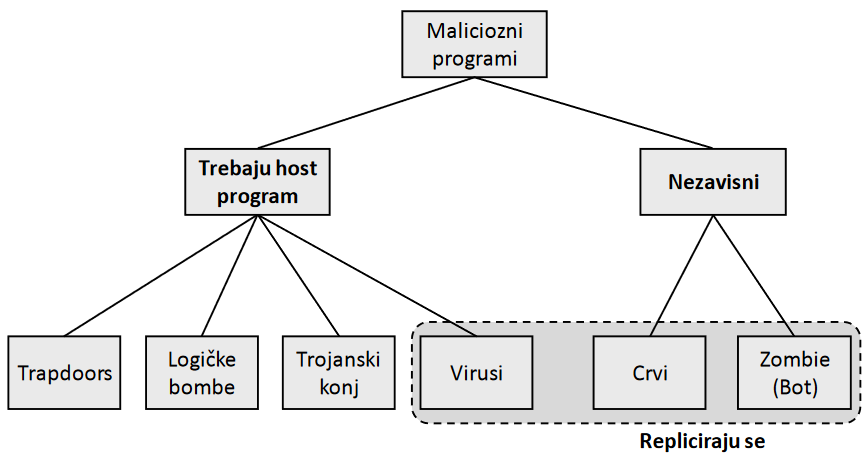
'Security Management' s RBAC

* Security management je ednostavniji s ulogama
  + Veza korisnik-uloga mijenja se tijekom vremena – skup korisnika se često mijenja (dinamička dodjela korisnika prema ulogama)
  + Skup uloga u sustavu je najčešće statičan
  + 'uloga-dozvola' veza je relativno stabilna (skup resursa i specifičnih prava pristupa sa određenom ulogom se također vrlo rijetko mijenja

Prednosti RBAC-a

* Upravljanje autorizacijom (Authorization managment)
  + RBAC razdvaja autorizaciju u 2 nezavisna dijela: jedan koji dodjeljuje uloge korisnicima i jedan koji dodjeljuje prava objektima za uloge
  + Koriscnici se mijenjaju češće nego uloge, jednostavno oduzimanje prava
* Hijerarhijske uloge
* **Najmanja privilegija**
  + Uloge omogućuju korisniku da se prijavi s najmanjim privilegijama potrebnim za odrađivanje trenutnih zadataka
  + Korisnici sa jačim/važnijim ulogama ne trebaju tolike ovlasti sve dok te privilegije isu zapravo potrebne
* **Razdvajanje dužnosti**
  + Niti jednom principu ne smijemo dodijeliti dovoljno prava za zlonamjerno/neispravno korištenje sustava
  + Npr. operacije koje zahtijevaju 2 osobe:  
    1. osoba je bilo koja autorizirana osoba, 2. osoba je bilo koja autorizirana osoba različita od prve (npr. banke)

Maliciozni softver

* Programi iskorištavaju ranjivost računalnog sustava
* Poznati kao malware
* Može se podijeliti u dvije kategorije
  + Parazitski malweri – programski fragmenti koji trebaju 'host' program (virusi, logičke bombe, 'backdoors')
  + Nezavisni 'self-contained' programi (npr. worms, 'bots' – mogu se izvoditi izravno na operacijskom sustavu)
* Razlikujemo softverske prijetnje koje se
  + Ne repliciraju – aktivirane okidačem (trigger), npr. logičke bombe, 'bot'
  + Repliciraju/šire (npr. virusi i crvi)

Virus - dio koda koji se unosi u host program (zarazi ga)  
- Ne može se izvoditi samostalno   
- zahtijeva pokretanje host programa kako bi se   
 mogao aktivirati (izvršava se potajno i kopira se)

Crv – program koji se može pokrenuti samostalni i koji se može širiti (razmnožavati) kopirajući se na druge hostove u mreži.

Logička bomba - program umetnut u softver od strane uljeza  
 - Izvršava se u specifičnim uvjetima (trigger)  
 - triggeri za logičke bombe mogu biti promjene u datoteci, određeni niz tipkanja ili specifično vrijeme ili datum

Trojanski konj - programi koji pokazuju jednu (korisnu) funkciju, ali zapravo izvode drugu (malicioznu) funkciju bez korisnikovog znanja

Backdoor (trapdoor) - bilo koji mehanizam koji zaobilazi normalnu sigurnosnu provjeru  
 - to je kod koji prepoznaje, npr, neki poseban niz znakova na ulazu  
 - programeri koriste backdoors na legitiman način za ispravljanje i testiranje programa

Exploit – zlonamjerni kod koji iskorištava određenu ranjivost

Keylogger – bilježi tipkanje na kompromitiranom sustavu

Rootkit – skup hakerskih alata instaliranih na računalu nakon što je napadač provalio u sustav i stekao administratorski (root-level) pristup

Zombie, bot – program na zaraženom računalo aktivira pokretanje napada na druga računala

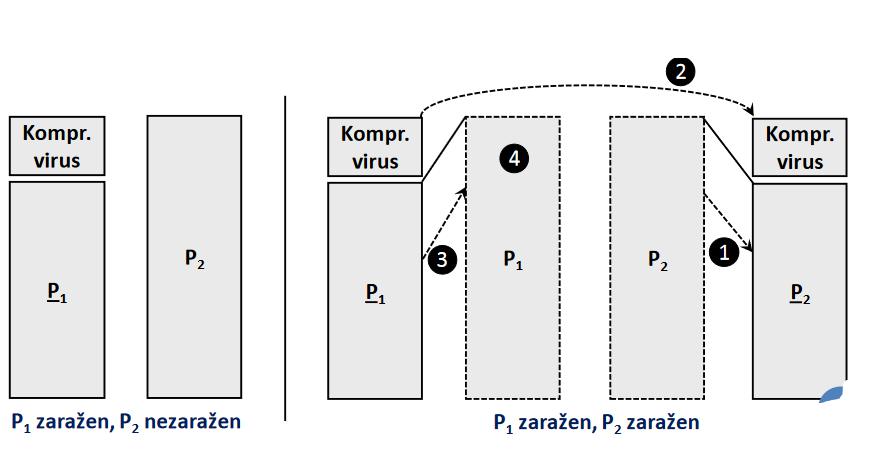
Spyware – prikuplja informacije sa računala i prenosi ih na drugi sustav

Virusne operacije

* **Faza mirovanja** (Dormant) – virus miruje, čeka okidač
* **Faza širenja** (Propagation) – stavlja kopiju sebe u druge programe ili područja na disku  
  (kopija ne mora biti identična, mijenja se kako bi se izbjeglo otkrivanje)
* **Faza okidanja** (Triggering) – aktivira se na neki događaj kako bi izveo namijenjenu funkciju
* **Faza izvršenja** (Execution) – namijenjena funkcija se izvodi (npr. prikazivanje poruke na ekranu, uništavanje programa ili datoteka

Detalji virusa ovise o hardveru/operacijskom sustavu

Virusna Struktura

* Glavne komponente
  + Mehanizam infekcije – kod koji omogućuje replikaciju
  + Trigger – događaj koji aktivira 'payload'
  + Payload – ono što čini, bilo to zlonamjerno ili dobroćudno
* Prepended / Postpended / Embedded
* Ključ virusne operacije jest: kada se zaraženi program poziva, prvo se izvršava kod virusa, a tek onda izvorni kod programa
* Prevencija: blokirati samu infekciju (teško!) ili širenje (pomoću kontrole pristupa kao u UNIX sustavima)

Kompresija virusnih operacija

* Virus koji je opisan na prethodnom slajdu je lako otkriti:
  + zaražena verzija programa duža je od nezaražene
* kako bi se izbjegla detekcija virus vrši kompresiju izvršne datoteke
  + cilj je postići istu veličinu zaražene i nezaražene datoteke

**Klasifikacija virusa – po cilju (by target)**

* **Boot sector virus** – inficira 'master boot record' ili 'boot record' i širi se kada je sustav podignut sa diska koji sadrži virus
* **File infector** – inficira datoteke operativnog sustava ili izvršni 'shell' kod
* **Macro virus** – inficira datoteke s makro kodom koji je interpretiran od strane aplikacije (npr. VBasic u MS Office dokumentima)

Boot Sector Virus

* Normalna boot procedura
  + POST (Power On Self Test) > BIOS otkriva 'bootable' uređaje > BIOS čita boot sektor tih   
    uređaja > BIOS prebacuje kontrolu na taj sektor
  + Bootable hard disk sadrži Master Boot Record (MBR)
    - 512-bajtni boot sektor, najčešće prvi sektor na hard disku (ujedno sadrži tablicu particija)
  + MBR kod traži 'bootable' particiju i prebacuje kontrolu na nju
* Ubacuju se u boot sektor područje
* Kada se sustav pokrene, virusi naprave štetu i zatim prebacuju kontrolu na premješteni MBR kod

Makro Virus

* Koristi makro-programski jezik same aplikacije
  + Makro je izvršni program ugrađen u dokument za obradu teksta ili druge vrste datoteka
  + Korisnici koriste makronaredbe kako bi automatizirali zadatke koji se ponavljaju i time uštedjeli na tipkanju
* Izrazite prijetnje
  + Ne zaražuju programe, nego dokumente
  + Neovisni o platformi
  + Jednostavno se šire (npr. email)
  + Uobičajena kontrola pristupa datotekama u svrhu suzbijanja njihovog širenja (zaražuju korisničke dokumente)

**Klasifikacija virusa – prema strategiji skrivanja**

* **Enkriptirani virus**
  + Virus stvara slučajni ključ za šifriranje, pohranjuje ga uz virus te šifrira ostatak virusa tim ključem
  + Kada se zaraženi program pozove, virus koristi pohranjeni slučajni ključ za dekriptiranje virusa
  + Kada se virus replicira, slučajni ključ se nasumično generira
* **Polimorfni virus**
  + Mutira sa svakom infekcijom, razotrikavnje prema potpisu (signature) virusa čini nemogućim
  + Ima posebno dizajnirani 'mutation engine' (ovaj engine omogućava mutiranje postupka dešifriranja)
* **Metamorfni virus**
  + Mutira se svakom infekcijom na način da se prepište pri svakoj iteraciji i time mijenja ponašanje i/ili izgled, povećavajući težinu otkrivanja
* **Skriveni (Stealth) virus**
  + Oblik virusa dizajnirad da se skriva od antivirusnog softvera
  + Cijeli virus je skriven nije samo 'payload'
  + Primjer: kompresirani virus
  + Stealth se odnosi na tehniku virusa koja mu omogućava izbjegavanje otkrivanja od strane antivirusnog softvera

Virusne Protumjere

* Najbolja protumjera je **prevencija**
  + Ne dopustiti da virus dođe u sustav
  + U načelu, ovo je nemoguće postići
* Potrebno je napraviti jedan ili više postupaka:
  + **Otkrivanje:** utvrditi da se infekcija dogodila i pronaći virus
  + **Identifikacija:** nakon otkrivanja, identificirati specifični virus
  + **Uklanjanje:** nakon identifikacije, ukloniti sve tragove virusa
* Ako je detektiran, ali se ne može identificirati ili otkloniti, zaraženi program se mora izbrisati ili zamijeniti
* Virus-antivirus koevolucija (coevolution) – vječna bitka

Detekcija: Negativni rezultat

* Kako bi utvrdili da je program P virus, mora biti utvrđeno da P zaražuje druge programe
* To nije moguće utvrditi budući da P može izbjeći proceduru donošenja odluke D i zaraziti druge programe ako D utvrdi da P nije virus
* Zaključujemo da je program koji točno razlučuje viruse od ostalih programa na temelju izgleda (appearance) neisplativ (infeasible)

**Razvoj antivirua (AV)**

* Tehnologije virusa i antivirusa su evoluirale
* Prvi virusi su bili jednostavni i laki za uklanjanje
* Usporedno sa rastom složenosti virusa rasle su i mjere zaštite
* Generacije antivirusne zaštite:
  + Prva: čitači potpisa (Signature scanners) – što je virus?
  + Druga: Heuristični (Heuristics) – što virus radi (na temelju strukture, provjera integriteta)?
  + Treća: Identificiranje radnji (Identify actions) – što virus zapravo radi?
  + Četvrta: kombinirani paketi (Combination packages)

AV Softver na bazi potpisa

* Za identifikaciju virusa potreban mu je potpis virusa
  + Rani virusi su imali isti uzorak bitova (niz bitova) u svim kopijama
  + Mali dio koda virusa kao sredstvo identifikacije
  + Dobar potpis je onaj koji će se naći u svim objektima koje je virus zarazio, ali postoji i mala vjerojatnost da će se naći u nezaraženim objektima
  + Ne prekratak (lažne potvrde), ne predug (lažna odbacivanja)
* Pronalaženje dobrog potpisa je teško i dugotrajno
  + Uključuje rastavljanje i proučavanje zaraženog objekta kako bi se identificirali ključni dijelovi virusa
  + Kada je potpis izlučen treba biti testiran na velikoj bazi nezaraženih programa kako bi smanjili vjerojatnost lažnih potvrda
* Pronalazi samo viruse čiji se potpis nalazi u bazi podataka antivirusnog softvera
  + Također, može detektirati blago modificirane verzije virusa
* Potpisi dodani u bazu podataka anti-virusa za detekciju ranijih virusa su nemoćni kod detekcije novih virusa (polimorfni virusi)

Heuristični AV Softver

* Detektira infekcije provjerom strukture cijelog programa, njegovih naredbi i ostalih podataka iz datoteke  
  (što virus radi – zaključak na temelji strukture)
* Može detektirati nepoznate zaraze
  + Općenito, traži sumnjivu logiku umjesto određenih potpisa
* Heuristični softver djeluje u dvije faze
  + Katalogiziranje ponašanja koje program može izvesti
  + Analiza katalogiziranih i zapaženih ponašanja te procjena je li to ponašanje slično virusima
* Scanner može tražiti različite vrste sumnjivih fragmenata koda
  + Npr. može pokušati pronaći dekripcijsku petlju kao kod enkripcijski virusa i otkriti ključ dešifriranja

Generička Dekripcija (GD)

* Pokreće izvršne datoteke kroz GD skener
  + CPU emulator za tumačenje instrukcija (ne koristi pravi CPU)
  + Virusni skener za provjeru virusnih potpisa
  + Emulacijski upravljački modul za upravljanje procesima
* Dopušta virusu samo-dešifriranje u interpreteru
* Periodično skeniranje potpisa virusa
* Generičko dešifriranje pretpostavlja:
  + Tijelo polimorfnih virusa je kodirano
  + Polimorfni virus se mora dekriptirati prije nego što se može izvršiti
  + Jednom kada se zaraženi program počinje izvršavati, polimorfni virus mora odmah prisvojiti kontrolu nad računalom kako bi mogao dekriptirati tijelo virusa, a zatim predati kontrolu nad sustavom dekriptiranom virusu
* GD skener učita test datoteku u samostalno virtualno računalo, kreirano nad virtualnim RAM.om
  + Unutar virtualnog računala, program se izvršava kao da se radi o stvarnom računali
  + Virus pokrenut unutar virtualnog računala ne može napraviti štetu jer je izoliran od stvarnog računala
* Svaki dio memorije u virtualnom uređaju ima odgovarajuću izmjenjenu memorijsku ćeliju
* Generičko dešifriranje koristi memorijske ćelije za predstavljanje područja memorije koji su modificirani tijekom procesa dešifriranja
* Jednom kad je virus dovoljno razotkriven, GD kreće u sljedeću fazu
* GD skener traži potpis virusa na onim područjima virtualne memorije koja su bila dešifrirana/modificirana virusom
* Ne rješava sve probleme (previše načina da zamaskiramo maliciozne kodove
* Napredna antivirusna tehnologija
  + Često, jedini način da saznamo da je kod maliciozan jest nadgledanje njegovog izvršavanja u realnom vremenu
  + Ako kod pokušava izvršiti funkcije koje narušavaju predefinirana sigurnosna pravila treba zaustaviti program
  + **Behaviour-Blocking AV Software**
  + Dobri protiv 'zero-day' napada

Internet crvi

* **Samoreplicirajući** program koji se širi preko interneta  
  + **Preko e-maila** – crv šalje kopiju sebe drugom sistemu
  + **Sposobnost udaljenog izvršavanja** – crv izvršava kopiju sebe na udaljenom sustavu, koristeći eksplicitno daljinsko izvršenje ili iskorištavajući mane (npr. preljevanje buffer) u nekom dijelu net servisa
  + **Daljinska prijava** – crv se prijavi na udaljeni sistem kao korisnik i zatim iskoristi komande da se kopira na udaljeni sistem
* Uses/Applications
  + Pokrenuti DDoS
  + Pristup osjetljivim informacijama
  + Širenje dezinformacija
  + Nepoznati razlozi (uglavnom potreba da bude prepoznat i postane poznat [nikad nije bilo slučajno])

**Operacije crva**

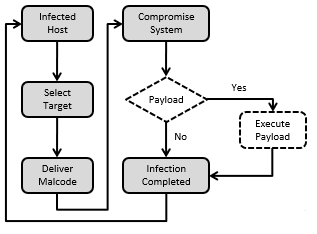
* **Latentna faza** – crv je u stanju mirovanja, čeka određeni događaj
* **Faza širenja** – crv traži druge sisteme, spoji se na njih, kopira se i pokrene (kopije ne moraju bit identične)
* **Faza događaja** – crve je aktiviran određenim događajem te izvodi namjenjenu funkciju
* **Faza pokretanja** – namijenjena funkcija je izvršena (npr. DoS napad na određenu metu)

Faza razmnožavanja (širenja)

* Da vi se proširio crv uglacnom radi sljedeće funkcije
  + **Traži druge sisteme** da bi ih zarazi koristeći ispitivanje različitih repozitorija udaljenih adresa
    - IP adresa-skeniranje prostora da bi se otkrile ranjive mete
    - Aktivna faza koja nije prisutna u virusima
  + **Ostvaruje konekciju** sa udaljenim sistemom
  + **Kopira se** u udaljeni sistem i pokrene kopiju

Generalizirani propagacijski model crva

* U prvoj fazi zaraženi domaćin traži ranjive mete
* Kad je meta pronađena zaraženi domaćin pokušava   
  dostaviti kod crva pronađenoj meti
* Pokretanjem tog koda odabrana meta je kompromitirana
* Jednom kad je sistem kompromitiran malware moze obaviti dodatne zadatke
  + Paket se odnosi na dodatne zadatake crva(Dos, instaliranje backdoora, samorepliciranje)



Akcije u svakoj fazi

* Faza odabiranja mete
  + Skeniranje random ip adresa
  + Nabavljanje e-mail adresa(npr. Iz adresara)
  + Preko sistema za razmjenu datoteka
* Faza dostavljanja koda (samo dio se može poslati u ovoj fazi)
  + Paket povezan s prelijevanjem buffera
  + Koristeći mail servisa za poruke
  + Posebno stvorene HTML stranice podignute na web serveru
* Kompromitiranje sistema
  + Izvršavanje koda crva: slabosti e-maila, korisnička intervencija, automatsko izvršavanje
  + Npr. Prelijevanje buffera, backdoor

**Razmnožavanje crva u stvarno životu**

Crv Morris (Robert Morris 1988)

* Da bi se širio, prvi zadatak crva je bio da otkrije druge domaćine poznate prvom zaraženom domaćinu koji bi dozvolili ulaz ovom domaćinu
  + Promatrane tablice sistema koje deklariraju koji drugi uređaji imaju povjerenje ovog domaćina, korisničke mail forwarding datoteke, tablice s daljinskim pristupom, izvještaji servisa koji su izvještavaju o statusu net konekcije
* Za svakog otkrivenog domaćina, razni napadi na UNIX sisteme
  + Probijanje password datoteke kako bi se prijavili na druge sisteme
  + Iskorištavanje buga u finger protokolu
  + Iskorištavanje buga u slanju maila
* Ako je barem jedna od gore navedenih metoda uspjela imamo udaljeni pristup shellu
  + Slanje bootstrap programa operacijskom sustavu kompromitiranog uređaja
  + Bootstrap program poziva parent program i downloada ostatak crva da se kopira na uređaj
* Oko 4000 od 60,000 tadašnjih internet domaćna su bili inficirani u roku od 16 sati nakon što je crv poslan

Code red (July 2011)

* Code red crv se širi preljevom buffera preko indeksnog servisa Microsoft Internet informacijskog servisa (IIS)
  + Infekcija je počela izdavanjem HTTP GET komande ranjivim IIS sistemima
* Crv skenira slučajne IP adrese da bi se proširio
* Tijekom određenog perioda crv se samo širi
* Nakon toga inicira denial-of-service napad protiv vladine web stranice tako što zatrpa stranicu sa paketima sa brojnih domaćina
* Code Red I v2 je inficirao skoro 360,00 servera unutar 14 sati
  + Prouzročio je probleme inficiranim serverima
  + Važnije- zauzeo je značajan dio kapaciteta interneta
* Code Red II je varijanta koja isto usmjerena na Micfosoft IIS
  + Ona također instalira backdoor, dozvoljavajući hackeru da pokrene naredbe na zaraženom računalu

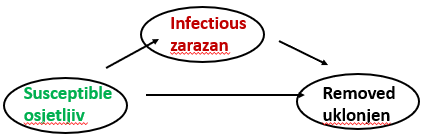
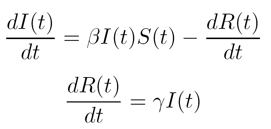
SQL Slammer (Siječanj 2003)

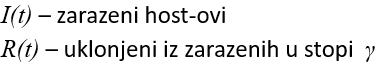
* Iskorištava pretek buffera u Microsoftovim SQL serverima
  + Jedan kratak paket (400 byta) preko UDP porta 1434 je bio dovoljan
* Crv je zarazio preko 90% ranjivih meta unutar 10 minuta
  + Izazvao je značajnu smetnju financijskim, transportacijskim i vladinim institucijama te isključuje bilo kakve akcije čovjeka
  + Nema malicioznog sadržaja, preopterećuje mreže
* Strategija širenja koristi nasumično skeniranje
  + Nasumično izabire IP adrese, te naposljetku pronađe i inficira osjetljive domaćine
* Slammer se proširio skoro dva reda veličine brže nego Code Red, a ipak je inficirao manje računala
  + **Najbrži računalni crv u povijesti ( Stopa skeniranja je 55 milijuna skenova u sekundi nakon samo 3 minute)**
* Brži od Code Red-a
* Slammer je ograničen širinom pojasa (bandwith) (njegov skener ima samo 400 bytoava, jedan UDP paket može iskoristiti ranjivost SQL servera)
* CR je limitiran kašnjenjem(njegov skener napravi TCP handshake i stoga mora čekati da primi SYN/AFK paket od mete)
* Autor Slameera je napravio nekoliko pogrešaka u generatoru nasumičnih brojeva (mnoge aktivne IP adrese su preskočene-manje inficiranih računala)

Crvi se šire eksponencijalnom brzinom

* Npr., 10M domaćina u < 5 minutes
* Teško se nositi s ručnom intervencijom
* Kako zaštititi naše sisteme? Koje su moguće posljedice?

Da bi bili u mogućnosti se zaštititi od budućih crva moramo ih razumjeti

* Obrazac širenja crva
* Utjecaj ljudskih protumjera (kao što su patchevi računalnih sistema, firewall, isključivanje uređaja sa mreže) na širenje crva
* Utjecaj na mrežni promet (prisjetimo se crva Slammer)
* **Jednostavan model epidemije**
  + Koristi vremenski model infektivnih bolesti da bi modelirao širenje crva
  + Tri moguća stanja-Ranjivo, Inficirano, U karanteni/uklonjeni
  + Pretpostavke:
    - Veličina populacije (#hosts) je velika
    - Svaki host ima jednaku vjerojatnost da kontaktira bilo koje druge host-ove u sustavu
    - Broj kontakata je proporcionalan # zaraznim X # osjetljivim
* „Inficirajući” domaćini: konstantno inficiraju druge
* „Uklonjeni” domaćini u području epidemije
  + Obnavljanje i imunost na virus
  + Mrtvi radi bolesti
* „Uklonjeni” domaćini u području računala:
  + Patchirana računala koja su čista i imuna na crva
  + Računala koja su ugašena ili odsječena od cirkulacije crva

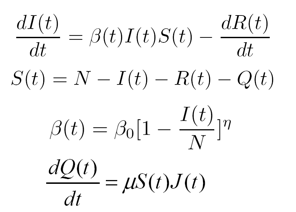
Klasični i generalni SIR modeli nisu savršeno prikladni kao **ljudska protumjera   
koja će ukloniti i osjetljive i zarazne host-ove iz optjecaja**

Ljudske protumjere uključuju:

* Clean and patch: download programa za čišćenje, zakrpe
  + Filter: stavljanje filtera na firewall i gateway
  + Isključivanje računala (kao u slučaju Code Red crv)

Također, infekcijska stopa *β* smanjena je zbog velike količine skeniranja prometa (npr. SQL Slammer crv)

Dvo Faktorski Crv Model

* **Ljudske protumjere i smanjena stopa infekcije** *β*
  + *N* – populacija host-ova
  + *S(t)* – susceptible-osjetljivi hostovi
  + *I(t)* – infectious- zarazni hostovi
  + *R(t)* – uklonjeni iz zaraznih hostova prema stopi *γ*
  + *Q(t)* – uklanjanje iz osjetljivih prema stopi *μ*

**Tehnike sondiranja (primjeri)**

* Random skeniranje
* Lokalno Subnet skeniranje
* Usmjeravanje crva
* Pre-generiran Hit List
* Topologijski

Random skeniranja

* 32 bitni broj je nasumično generiran I korišten kao IP adresa
  + Osim toga: IPv6 crvi ce biti drugaciji…
* Npr., Slammer i Code Red I
* Pogađa black-holed IP prostor često
* Samo 28.6% IP prostora je alocirano (dodjeljeno)
* Osim toga: može pratiti crve praćenjem neiskorištene adrese
* Honeypots

Subnet Skeniranje

* Generira zadnjih 1, 2, ili 3 bajta IP addrese slučajno
* Code Red II i Blaster
* Neka skeniraja moraju biti sasvim slučajna da zaraze cijeli Internet

Usmjeravanje crva – BGP informacija može reći koji IP adresni blokovi su namijenjeni (alocirani) (informacija javno dostupna)

Topologijski

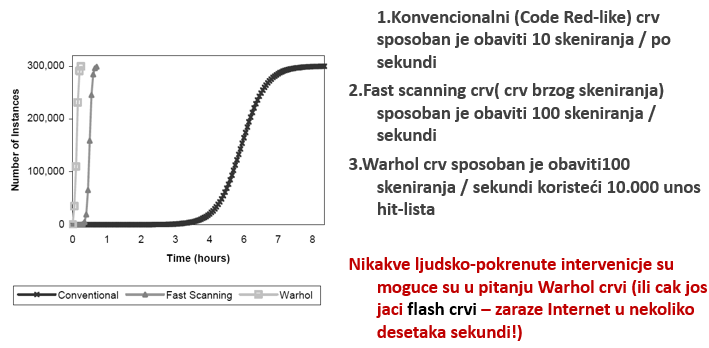
* Koristi informacije na zaraženom host-u za pronalazak sljedeće mete
* Morris Crv korišteni /itd/host-ovi , .rhost-ovi
  + Email adrese knjige
  + P2P software obično pohranjuju informacije o vršnjacima (peers) na koji se svaki host spaja

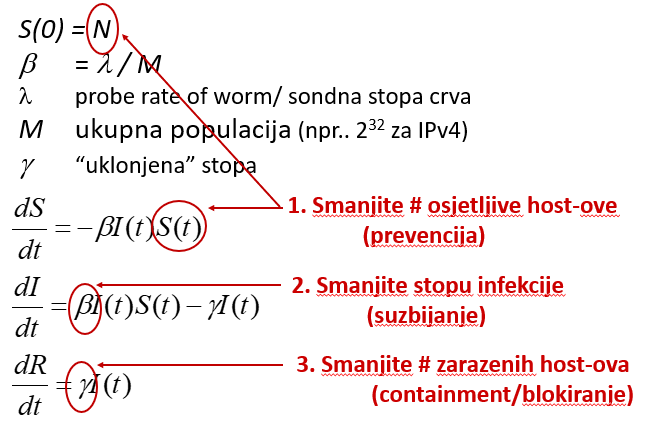
Hit lista

* Hit lista ranjivih uređaja je poslana sa (payload-om) korisnim sadržajem
  + Određena prije pokretanja crva skeniranjem
  + Daje crvu poticaj u fazi sporog starta
  + Preskaće fazu koja prati eksponencijalni model
  + Stopa infekcije izgleda linearno u fazi brzog napretka
* Moze izbjeći otkrivanje (detekciju) od ranih sistema otkrivanja

Warhol: Hit Lista + Permutacijsko skeniranje

* Vrijeme infekcije procijenjeno na oko 15 minuta
* Andy Warhol: „U budućnosti, svatko će imati 15 minuta slave“





Protumjere crva

Prevencija

* Cilj je smanjiti veličinu ranjive populacije
* Sigurno programiranje, primjena nadogradnje softvera, AV zaštita
* Krpanje (Patching)
  + Općenito, zakrpe (patches) trebaju par dana za objavljivanje   
    (Tek sada relativno pouzdane distribucijske mreže za zakrpe niču)

Blokiranje i suzbijanje (najlakši)

* Firewall, Filtriranje sadržaja, automatizirane Routing crne liste, odspajanje zaraženih računala

Protumjere Crva

* Preklapaju se sa anti-virusnim tehnikama
* Kada je crv jednom u sistemu A/V ga može otkriti
* Crvi također uzrokuju znajčanu net aktivnost
* Skeniranje za druge ciljeve/mete (targets) (stopa skeniranja od10-10000 scans/second)

Crvni obrambeni pristupi ukljucuju:

* Signature-based worm scan filtering
  + Generira skenirani potpis crva (worm scan signature) za prevenciju ulaska worms scana u mrezu/host (prevent worm scans from entering a network/host)
* Filter-based worm containment
  + Fokusira se na sadržaj crva rađe nego na skeniran potpis   
    (Focuses on a worm content rather than a scan signature)
* Klasifikacija-Korisnog sadržaja na temelju blokiranja crva (Payload-classification-based worm containment)
  + Paket na bazi provjere
* Prag slučajno skenirane detekcije (Threshold random walk scan detection)
  + Iskorištava slučajnost u odabiru destinacije za spajanje (za otkrivanje skeniranja)
* Ograničavanje i zaustavljanje stope (Rate limiting and rate halting)  
  + Limitiranje ili blokiranje odlaznog prometa kada je dani prag prekoračen (za brze crve)

What is DoS?

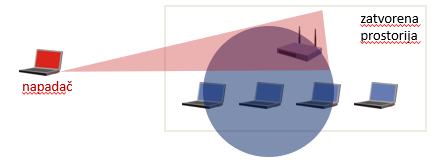
* DoS attack is an attempt (malicious or selfish) by an attacker to cause a victim to deny service to its customers
  + DoS is an action that prevents the authorized use of **networks**, **systems**, or **applications** by exhausting resources such as CPU, memory, bandwidtch, and disk space
* Categories of resources that could be attacked
  + **Network bandwidth**
    - Related to the capacity of network links connecting a server to the wider Internet
  + **System resources**
    - Overloading/crashing network handling software (e.g., SYN spoofing,   
      ping-of-death – a “huge” ping packet)
  + **Application resources**
    - Overloading the capabilities of a web server application (e.g., valid querries)
* Typical approaches to DoS attacks
  + Exploit system design weaknesses (e.g., ping-of-death)
  + Impose computationally intensive tasks on a victim (e.g., calculation of a secret Diffie-Hellman key, RSA operations)
  + Flood the victim with a huge number of packets (requests)
    - A DoS attack if it comes from (involves) a single computer
    - **Distributed DoS (DDoS)** when multiple stations coordinate to flood the victim
      * Independent of underlying network protocols
      * Exploits the resource asymmetry between the Internet and the victim
      * Perhaps one of the major threats to the stability of the Internet
* DoS attacks can happen on any of the TCP/IP layers
  + Physical layer (IEEE 802.11)
  + Data-link layer (IEEE 802.11)
  + Network layer (ARP, ICMP...)
  + Transport layer (TCP, UDP...)
  + Application layer (PGP, SSL, etc.)
* Many DoS are regularly considered straightforward
  + Radio jamming, flooding, etc.
  + As a result there are very few sound DoS solutions
* Also, DoS attacks are usually termed as “malicious attempts”
  + However, often an attacker is driven by pure selfish interest
  + Where there is interest, there is dedication and concentrated effort

IEEE 802.11b/g: fizička razina

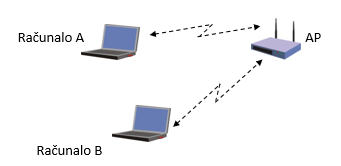
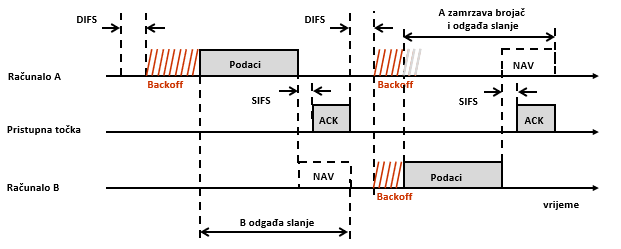
* 2.4 GHz (2.4–2.4835 GHz) 14 frekvencijskih kanala
  + Centralne frekvencije kanala su pomaknute za 5 MHz
  + 13 se koristi u EU, 11 US
* Koristi tehniku raspršenog spektra (Spred Spectrum - SS)
  + Poskakivanje frekvencije (Frequency Hopping SS)
  + Pseudoslučajni raspršeni spektar (Direct Sequence SS)
* Kodiranje i modulacijske sheme odredjuju max. komunikacijske brzine (1, 2, 5, 11, 54Mbps, ...)
  + 802.11b na 11Mbps
    - Complementary Code Keying (CCK)
    - Differential Quadrature Phase Shift Keying (DQPSK)
  + 802.11g na 54Mbps
    - Orthogonal Frequency Division Multiplexing (OFDM)

Komunikacija između AP i klijenta

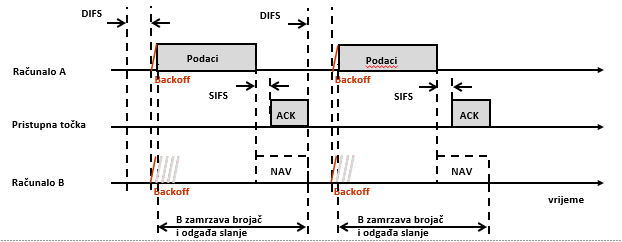
* AP komunicira sa klijentom koristeci jedan kanal (npr. channel 6)
* Samo jedan klijent komunicira sa AP u svakom trenutku
* Znacajna interferencija preostaje na kanalu
  + Od susjednih kanala
  + Od okoliša u kojem se nalazi AP
* DoS napadi ometanjem radio singala (**radio jamming**)
  + Napadač emitira radio signal u frekvencijskom podrucju aktivnog kanala (npr., kanal 6 u podrucju 2.437 GHz +/- 10MHz)
  + Da bi pojačao efekt ometanja, napadač koristi usmjeravajuće antene
  + IEEE 802.11 ne pruža zaštitu protiv radio ometanja
    - DoS putem radio ometanja često se zanemaruje (pogrešno)



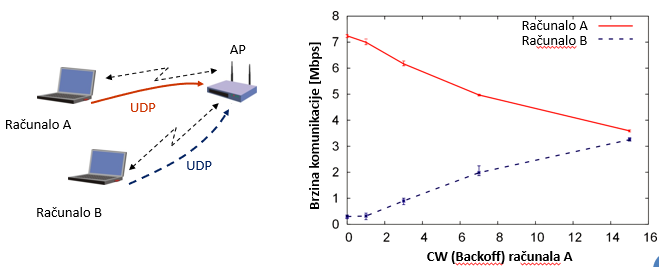
IEEE 802.11b Data Link Layer

* MAC (**Media Access Control**) arbitrira pristup zajednickom kanalu izmedju veceg broja bezicnih stanica
* **Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA)** paradigma
  + Prije transmitiranja paketa na kanalu, mobilno računalo provjerava da li je kanal već “zauzet” (npr., od strane drugog računala)
  + Izbjegavanje kolizija između paketa dva ili više računala putem randomiziranog “back-off” mehanizma
* Notacija:
  + **DIFS**: Distributed Inter-Frame Spacing
  + **SIFS**: Short Inter-Frame Spacing
  + **Backoff**: slučajan broj iz skupa {1,2,…, CW} – izražava se u vremenskim intervalima
  + **CW**: maksimalno trajanje Backoff-a
  + **NAV**: Network Allocation Vector

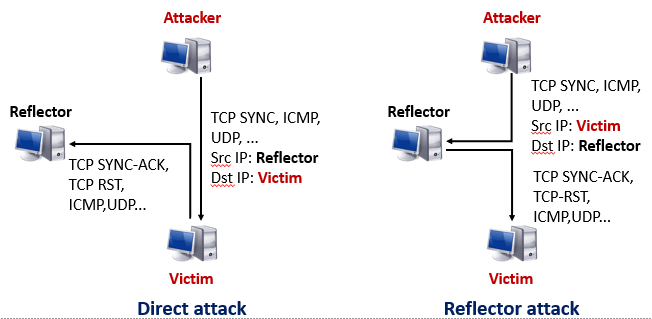
Link Layer DoS Attacks: System Weakness

* Notacija:
  + **DIFS**: Distributed Inter-Frame Spacing
  + **SIFS**: Short Inter-Frame Spacing
  + **Backoff**: slučajan broj iz skupa {1, 2, …, CW}
  + **CW**: maksimalno trajanje Backoff-a
  + **NAV**: Network Allocation Vector

Link Layer DoS Attacks: Implications

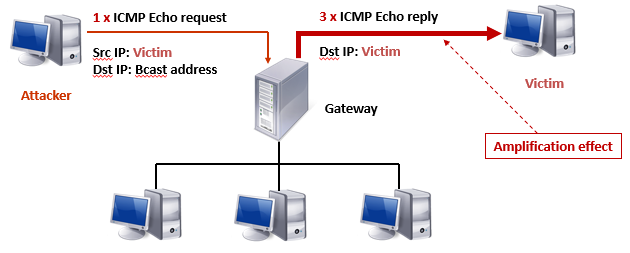
* Primjer: manipulacija Backoff vrijednostima
  + Jednostavna implementacija (jedna linija koda kod bežičnih adaptera koji koriste Atheros radio čipove, npr. Proxim Orinoco - )
  + IEEE 802.11e sa QoS (Quality of Service) podrškom omogućava manipulaciju Backoff-a, DIFS-a, SIFS-a!

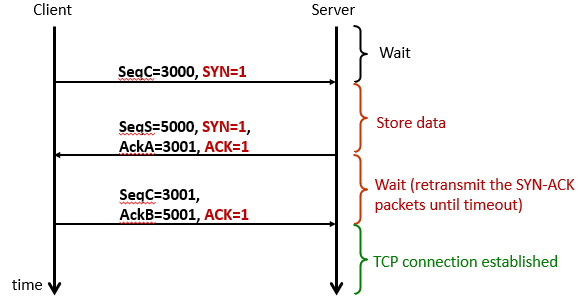
Flooding-Based DoS Attacks

* Flood the victim with a huge number of packets (requests)
  + A DoS attack if it comes from (involves) a single computer
  + **Distributed DoS (DDoS)** when multiple stations coordinate to flood the victim
    - Independent of underlying network protocols
    - **Exploits the resource asymmetry** between the Internet and the victim
    - Perhaps one of the major threats to the stability of the Internet
* Generally, there are two types of flooding attacks
  + **Direct attacks**
  + **Reflector attacks**

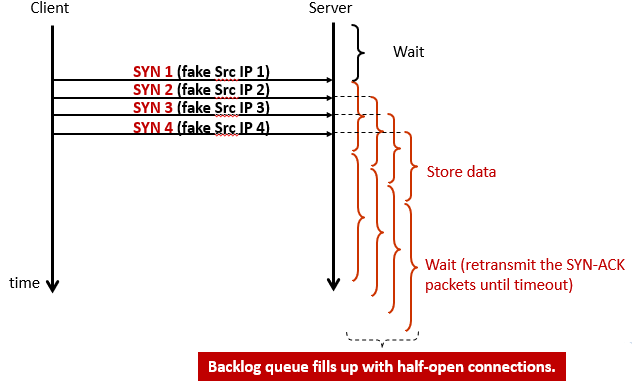
Smurf DoS Attack

* Victim flooded with ICMP “Echo-reply” packets
* Attacker sends numerous ICMP “Echo-request” packets to the broadcast address of many subnets
  + ICMP “Echo-request” packets contain the victim’s address as the source IP address
* Machines on the subnets respond by sending ICMP packets to the victim
  + Recall: ping utility implemented using the ICMP “request” and “reply”





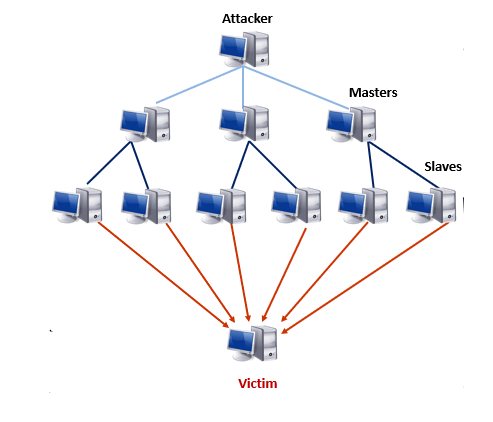
SYN Flooding Attack

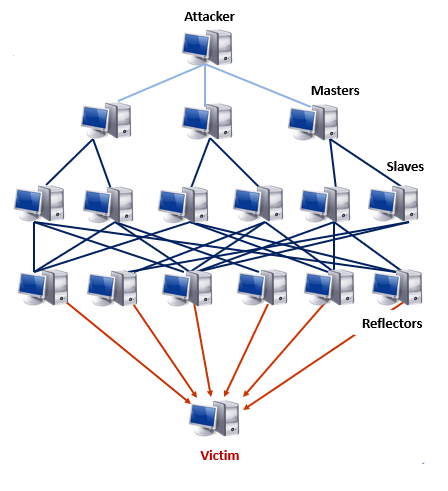
* **TCP handshake**
* Server waits Client’s ACK and retransmits the SYN-ACK several times before giving up (half-open connections can quickly consume all the memory allocated for pending connections)
* Each half-open connection is held for a certain amount of time before giving up [Chang’02]
  + Windows system ~ 9 s (2 retransmissions), BSD system ~ 75 s (3-4 retransmissions), Linux ~ 309 s (up to 7 retransmissions)
* If each SYN packet is 84 bytes long
  + 56 kb/s connection sufficient   
    to stall Linux and BSD servers  
    that admit up to 6,000   
    half-open TCP connections
  + 1 Mb/s connection sufficient   
    to stall all three servers   
    that admit up to 10,000 h-o. c.
* Jamming T1 link
  + Capacity 1,544 Mb/s
  + Direct ICMP ping requires around 5000 pckts/s

Low Rate SYN Countermeasures

* Straightforward (and inadequate)
  + Increase the backlog queue size
  + Decrease the time that a server holds half-open connection
* SYN Cookies
  + Sever does not store any state until the client passes the “random” challenge (returns a random value carried by the server’s cookie)
    - If attacker spoofs the source IP, it cannot receive the SYN Cookie
  + Better than the first solution

Direct Distributed DoS (DDoS)

* Attacker coordinates multiple “zombie” machines to simultaneously flood a target machine or the network uplink
  + High flooding rate attacks (many stations involved)
* Direct DDoS
  + The army of the attacker consists of  
    “master zombies” and “slave zombies”
  + Both classes are **compromised** machines  
    (infected by malicious code)
  + **The attacker coordinates masters**
  + **The masters trigger and coordinate  
    slave zombies**
  + **Zombies flood the victim’s system (e.g., SYN flood)**
* Spoofed source IP addresses are used
  + Attacker wants to hide IDs of zombies so that victim cannot trace attack back to them
  + Attacker wants to discourage any attempt of the victim to filter out the malicious traffic

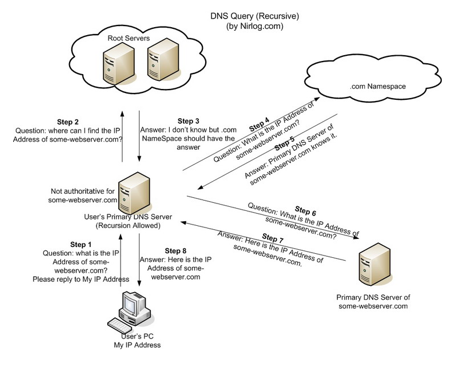
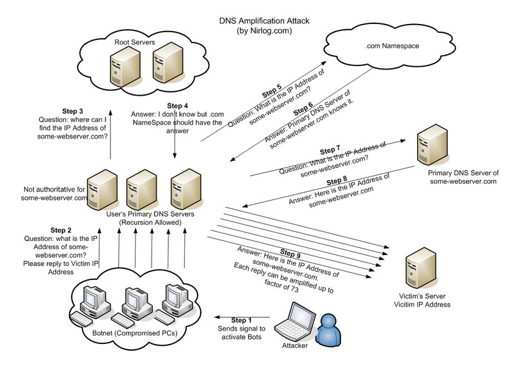


Reflector DDoS

* The army of the attacker consists of
  + Master zombies, slave zombies and  
    **reflectors**
* Slave zombies are led by master zombies to send   
  a stream of packets **with the victim’s IP address   
  as the source IP address to other uninfected   
  (non-compromised) machines (reflectors)**
  + **Reflectors generate messages for the   
    victim in response to messages from the slave   
    zombies**
  + E.g., if SYN flood is used, reflectors are victims themselves
    - But the situation is not that severe as the  
      direct attack
* Some differences wrt. direct DDoS attacks
  + A reflector attack requires a set of predetermined reflectors (DNS servers, HTTP servers, routers)
  + The magnitude of the attack is based on the size of the reflector pool (instead of the zombie pool)
  + The reflectors could also be more dispersed on the Internet
    - **The attacker does not have to compromise such machines**
  + The reflected packets are normal packets with legitimate source addresses and packet types
    - **Harder to filter such packets based on address spoofing or other route-based mechanisms**

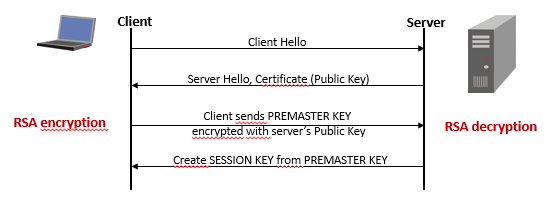
DNS Amplification Attack

* Classic DNS protocol
  + 60 byte UDP request packet
  + Maximum 512 byte UDP response
* Recently, the DNS protocol extedned (IPv6, security)
  + Allow larger responses of over 4000 bytes
* Attacker can exploit this and achieve significant gain
  + Invests 60 bytes to get over 4000 bytes for “free”
  + **Amplification effect**
* Turning this into an attack
  + Attacker creates a series of DNS “small” requests using the spoofed source IP address of the victim
  + The victim gets flooded with “large” DNS responses



Upper Layer DoS Attacks

* Transport layer (e.g., SSL/TLS)
  + RSA-encrypt speed approx. 10 x RSA-decrypt speed (**amplification effect**)



* Application layer
  + eBanking service locks an account after 3 consecutive failed login attempts
    - An attacker can easily lock down a number of such accounts
  + Send HTTP request for some large PPT file (e.g., SRP webpage :-)
    - **Amplification effect (easy work for the client, hard work for the server)**

**General Countermeasures**

* Attack prevention and preemption (**before the attack**)
  + Enable the victim to endure attack attempts without denying service to other clients
  + Backup resources on demand, enforce policies for resource consumption
  + Scan hosts for the presence of malware such as zombies or worms
* Attack detection and filtering (**during the attack**)
  + Attempt to detect the attack as it begins and respond immediately
  + Detection involves looking for suspicious patterns of behavior
  + Response involves filtering out packets likely to be part of the attack (e.g., firewalls)
* Attack source traceback and identification (**during and after the attack**)
  + Try to identify the source of the attack as a first step in preventing future attacks
  + Does not prevent an ongoing attack

Client Puzzles

* The main idea is to slow down the attacker by forcing him to commit considerable computational resources
  + In the event of an attack, each connection request will be challenged with a relatively easy problem (“puzzle”)
  + Only those connection requests that submit puzzle solution with the request will be “approved”
  + Example: SSL/TLS “puzzle” [Dean and Stubblefield’01]
* Simple puzzle
  + For a preimage resistant hash function h(x), a client puzzle is the triple (n, x’, h(x))  
    where x’ is x with n lowest bits set to 0.
  + The solution to the puzzle is the full value of x
  + Because h(x) is preimage resistant, it takes, on average, 2n-1 calculations of h(x) for the client (attacker)
* Generating and checking puzzles easy
  + The server only needs to generate a random block of data and evaluate the hash function twice
* Does not work with DoS attacks where the attacker saturates the victim's incoming link
  + En route routers do not verify the validity of puzzles

CAPTCHA

* **C**ompletely **A**utomated **P**ublic **T**uring test to tell **C**omputers and **H**umans **A**part
  + Carnegie Mellon University
  + Challenge-response test used to determine whether or not a user is human
  + Computers are not able to solve tests of the following kind
* Applies to application layer DoS
  + Attacker cannot automate an attack whereby he locks out many eBanking accounts

DDoS Attack Detection and Filtering

* **Detection** – identifiying DDoS attacks or attack packets
* **Filtering** – after identifying attack packet flows or packet, filtering responsible for classifying and dropping them
  + The effectiveness depends on **false positive** and **false negative ratios**
* Detect-and-filter approach can be performed in four places on the paths between the victim and the attacking machines

Network Ingress Filtering (RFC 2827)

* Preventing DDoS attacks which employ IP address spoofing
  + An input traffic filter on the ingress (input) link of the router restricts traffic to allow only traffic originating from source addresses within the 204.69.207.0/24 prefix, and prohibits the attacker from using source addresses outside of this prefix range. [RFC 2827]
* All ISP must do this (global trust)
  + **If only 10% do not implement ingress filtering, there is no defense**

At a Victim’s Upstream ISP Network

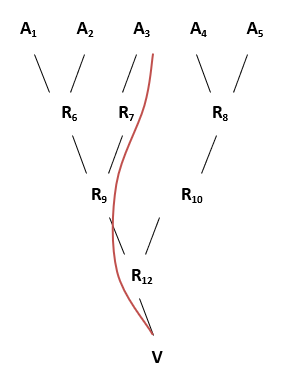
* Frequently, upstream ISP is requested (through telephone calls) by a DDoS victim to filter attack packets
  + Before doing this, attack packets have to be identified to avoid dropping legitimate packets
  + Ideally, a victim network may send to an upstream ISP router an intrusion alert message, which specifies the signature of the attack packet flows
  + These alert messages may be pushed further upstream so that the upstream ISPs are notified to filter those packets matched in the signature
  + This approach is effective only if ISP networks are willing to cooperate and to install packet filters upon receiving intrusion alerts
    - Seldom the case
    - Many issues – how to trust alert messages?

Source Traceback

* **IP traceback refers to the problem (solution) of identifying the actual source of any packet sent across the Internet without relying on the source information in the packet**
* Two general approaches
  + Routers record information about packets they have seen for later traceback requests
  + Routers send additional information about the packets they have seen to the packets’ destinations (within the packets or using other channels such as ICMP messages)
* **Infeasible to use IP traceback to stop on-going DDoS**
  + IP traceback could be very helpful in collecting evidence for post-attack law-enforcement
  + **IP traceback is ineffective in reflector attacks**

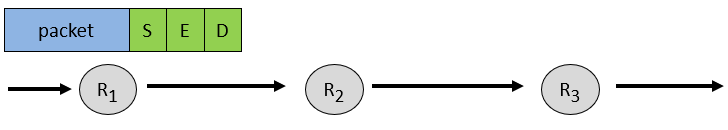
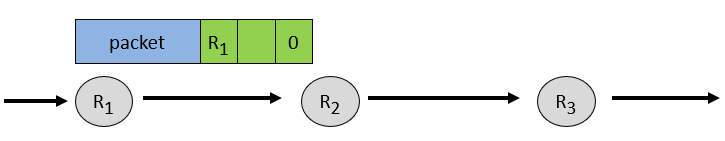
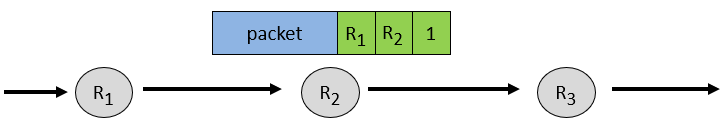
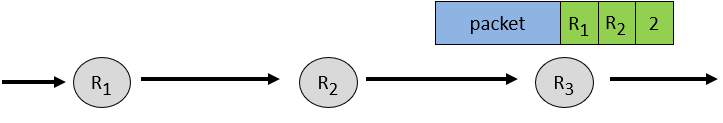
Simple Traceback Method

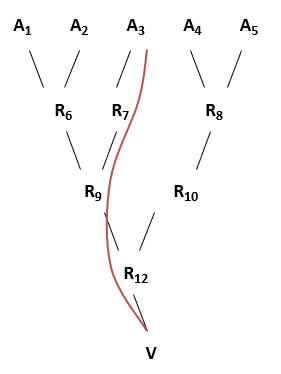
* Write path into network packet
  + Each router adds its own IP address to packet
  + Victim reads path from packet
* Problem:
  + Requires space in packet
    - Path can be long
    - No extra fields in current IP format
      * Changes to packet format too much to expect

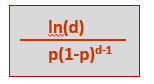
Better Traceback Method

* DDoS involves many packets on same path
* Store one link in each packet
  + Each router probabilistically   
    stores own address
  + Fixed space regardless   
    of the path length
* Packet marking algorithm
  + E.g., **edge sampling**

Edge Sampling

* Data fields written to packet:
  + Edge: **start** and **end** IP addresses
  + Distance: number of hops since edge stored
* Marking procedure for router R
* Packet received
  + R1 receives packet from source or another router
  + Packet contains space for start (S), end (E), distance (D)
* Begin writing edge
  + R1 chooses to write start of edge
  + Sets distance to 0
* Finish writing edge
  + R2 chooses not to overwrite edge
  + Distance is 0
    - Write end of edge, increment distance to 1
* Increment distance
  + R3 chooses not to overwrite edge
  + Distance > 0
    - Increment distance to 2

Path Reconstruction

* Extract information from attack packets
* Build graph rooted at victim
  + Each (start,end,distance) tuple provides an edge
* The number of packets needed to reconstruct path
  + p is marking probability
  + d is length of path

Many Traceback Proposals

* Advanced and Authenticated Marking Schemes for IP Traceback
  + Song, Perrig. IEEE Infocomm ’01
  + Reduces noisy data and time to reconstruct paths
* An algebraic approach to IP traceback
  + Stubblefield, Dean, Franklin. NDSS ’02
  + Hash-Based IP Traceback
  + Snoeren, Partridge, Sanchez, Jones, Tchakountio,  
    Kent, Strayer. SIGCOMM ‘01

Inferring Internet DoS Activity [Moore’01]

* Backscatter analysis
  + Provides an estimate of **worldwide denial-of-service**activity
  + In 2001 observed more than 12,000 attacks against more than 5,000 distinct targets

