Intrusion Detection

or

"Technologies for detecting anomalous behavior in distributed systems"

http://www.gnist.org/~lars/hovedfag/spesialpensum/

Intrusion Detection?

What?

- security as in .... safe?
- policy?- the boring!
- mechanism? – the real deal!
- overview!

Why?

- threats!
- types of attack
- possible intruders!

Intrusion Detection? (2)

How?

- detect (IDS) and protect (firewall?)

Intrusion Detection Systems (IDS)

- history
- different types
- common problems
- do we really need IDS?
- future challenges:
  - * ad-hoc networks
  - * application level

Security?

*** more than one definition of (computer) security:**

1. most common: *"Measures and controls that **ensure confidentiality, integrity, and availability** of information-system (IS) assets including hardware, software, firmware, and information being processed, stored, and communicated."* (US Department of Homeland Defense)

2. *"The application of hardware, firmware and software security features to a computer system in order to **protect against, or prevent**, the unauthorized disclosure, manipulation, deletion of information or denial of service."*

3. *"The protection resulting from all measures to **deny unauthorized access and exploitation** of friendly computer systems. Also called COMPUSEC."* - DOD

4. ***"Broadly speaking, security is keeping anyone from doing things you do not want them to do to, with, or from your computers or any peripherals"*** - William R. Cheswick

Security? (2)

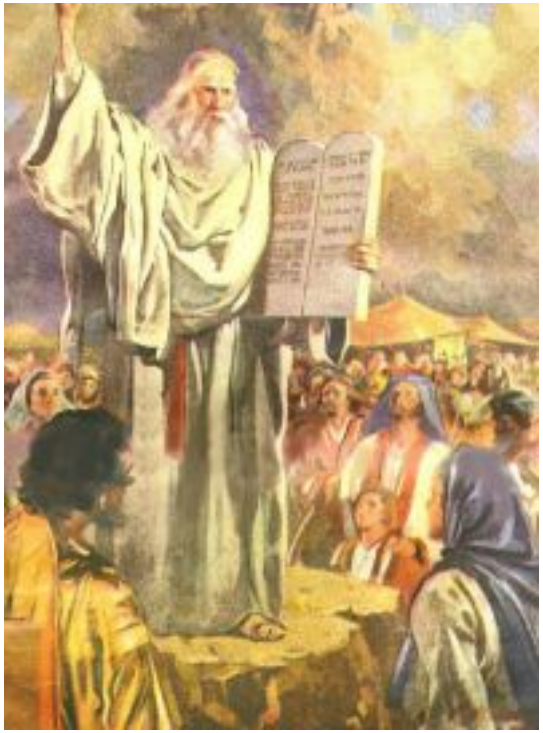
Host vs. network security (Hallingstad):

"**network security**: Commonly used term to address the security in transmission of information between computers."

"**host security**: Commonly used term to address the security of all systems running on one computer."

*"A **security policy** is a statement of what is, and what is not allowed"* -- Bishop

*"A **security mechanism** is at method, tool, or procedure for enforcing a security policy"* -- Bishop



Policy

Computer security rests on:

*** confidentiality**

- keeping data and resources hidden
- early works: military's “need-to-know” principle
- historical: “confidentiality = security”

*** integrity**

- prevent: block any unauthorized attempts to change data
- detect: report that the data is no longer trustworthy
- origin/data integrity
- important in commercial business

*** availability**

- the ability to use the information or resource as desired

Policy (2)

A policy may be presented mathematically or in a high level language.

UiO's “IT-reglement”:

Reglement for bruk av universitetets IT-tjenester

Vedtatt av Det Akademiske Kollegium den 20. februar 1996. English version

1. Omrdet for dette reglement

1.1 Dette reglement gjelder for bruk av universitetets IT-tjenester. Med IT-tjenester siktes det til felles maskiner og IT-systemer, sluttbrukerutstyr, nettverk, programmer, data m.v. som stilles til disposisjon av universitetet inklusive lokale, nasjonale og internasjonale nettverk, eller andres maskiner og systemer som man fr tilgang til gjennom slike ressurser. Reglementet gjelder for ansatte, studenter og andre som fr tilgang til IT-tjenestene, heretter kalt brukere.

1.2 Reglementet skal vre oppslttt p egnede steder som f.eks. terminalstuer, samt finnes tilgjengelig p elektronisk form. Det kan fs ved henvendelse til lokal IT-ansvarlig eller til USIT. Reglementet skal ogs deles ut ved frste gangs brukerregistrering, eller p annen mte gjres kjent for brukere.

1.3 En bruker plikter holde seg informert om det til enhver tid gjeldende reglement og eventuelle supplerende bestemmelser. Slike bestemmelser skal godkjennes av IT-faglig rd. Unntatt fra dette er supplerende bestemmelser som gjelder overvkning og sanksjoner. Disse skal godkjennes av kollegiet.

2. Lojal bruk

2.1 Nr en bruker skal identifisere seg ved bruk av IT-tjenestene er vedkommende forpliktet til alltid identifisere seg med sitt eget brukernavn, passord eller p annen regulr mte.

2.2 En bruker har plikt til flge driftspersonalets anvisninger om bruk av IT-tjenestene. En bruker har ogs plikt til sette seg inn i bruksanvisning, dokumentasjon m.v. p forsvarlig mte, slik at brukeren reduserer muligheten for av uvitenhet skape risiko for driftsforstyrrelser eller tap av data, programmer eller utstyr.

2.3 Ved opphr av ansettelses-, studie- eller brukerforhold er brukeren ansvarlig for at kopier av data, programmer m.v. som eies/disponeres av universitetet sikres for den lokale IT-ansvarlige eller USIT. Andre filer m.v. som er lagret under brukerens navn eller lignende skal brukeren selv slette. Skjer dette ikke innen rimelig tid, og senest innen 3 mneder, kan lokal IT-ansvarlig eller USIT slette slike filer. Det samme gjelder ved ddsfall, men da skal prrende varsles og f adgang til overta kopi av materialet fr sletting finner sted.

3. Datasikkerhet

3.1 En bruker m ikke bidra til at det oppstr avbrudd i noen del av systemet eller p annen mte forrsake ulempe for andre.

3.2 En bruker plikter selv treffe de tiltak som er ndvendig for at tap av egne data, programmer eller lignende skal f minst mulig flger gjennom sikkerhetskopiering, forsvarlig oppbevaring av media, bruk av anbefalte rutiner vedrrende nettet og utnytting av de beskyttelsesmekanismer som finnes (f.eks. filbeskyttelseskoder).

3.3 En bruker plikter ikke gjre passord eller andre sikkerhetselementer kjent for andre.

3.4 Dersom en bruker blir gjort kjent med at hans/hennes passord kan identifiseres ved hjelp av maskinelle kontrollrutiner, plikter brukeren endre passordet.

3.5 En bruker plikter forhindre at uautoriserte personer fr tilgang til bruk av nettet og IT-tjenestene, adgang til maskiner eller tilgang til rom hvor utstyr er tilgjengelig, og ogs ellers bidra til hindre at uautoriserte personer fr tilgang til utstyret. Brukernavn og passord m ikke lnes ut.

3.6 En bruker plikter vre oppmerksom p at programmer eller data kan inneholde unskede elementer (virus), og selv treffe hensiktsmessige tiltak for kvalitetskontroll.

3.7 En bruker plikter rapportere forhold som kan ha betydning for IT-tjenestenes sikkerhet eller integritet til nrmeste foresatte, lokal IT-ansvarlig og USIT.

4. Respekt for andre brukere og personvern

4.1 En bruker m ikke ske oppn uautorisert tilgang til andres data, programmer m.m, eller ske gjre seg kjent med andres passord eller andre sikkerhetselementer.

4.2 En bruker plikter gjre seg kjent med de regler som srlig gjelder for personlige opplysninger. Datamaskinbaserte filer som inneholder opplysninger om enkeltpersoner eller om selskaper eller organisasjoner (fysiske og juridiske personer) vil normalt bare kunne opprettes etter samtykke fra Datatilsynet. Hvis en bruker nsker registrere personlige opplysninger, plikter vedkommende forsikre seg om at det er adgang til dette etter personregisterloven eller forskrifter gitt med hjemmel i loven, eller i medhold av konsesjon gitt til universitetet. Dersom registeret ikke vil vre tillatt etter de nevnte regler, plikter brukeren selv ske om ndvendig tillatelse. Den som har ansvar for driften vil kunne gi veiledning om hvordan brukere skal forholde seg.

4.3 En bruker har taushetsplikt om noens personlige forhold som man fr kjennskap til gjennom bruk av IT-tjenestene.

5. Smmelig bruk og ressursbevissthet

5.1 Brukeren har et medansvar for at ressursene utnyttes best mulig. Som ressurser siktes det her til tid og kapasitet bde for maskiner, nettverk og personale knyttet til virksomheten rundt IT-tjenestene.

5.2 En bruker skal vre meget tilbakeholden med bruk av IT-tjenestene til virksomhet som ikke har direkte tilknytning til faglig virksomhet, administrasjon, egen forskning, studier eller organisasjonsarbeid i forening ved universitetet.

5.3 En bruker m ikke benytte IT-tjenestene til fremsette rekrenkelser eller diskriminerende uttalelser, formidle pornografi eller taushetsbelagte opplysninger, krenke privatlivets fred eller oppfordre eller medvirke til ulovlige eller reglementsstridige handlinger.

6. Rettigheter

6.1 Bde til datamaskinprogrammer og til data (tekst s vel som samlinger av opplysninger som f.eks. databaser) er det normalt knyttet rettigheter som gjr bruk avhengig av avtale med rettighetshaver. Bruker forplikter seg til respektere andres rettigheter. Dette gjelder blant annet ved bruk av musikkinnslag og annen rettighetsbelagt informasjon i World Wide Web og andre elektroniske informasjonssystemer.

6.2 Nr universitetet gjr programmer, data eller annet tilgjengelig, plikter bruker respektere de begrensninger for bruk som flger av avtalen. Avtalen vil vre tilgjengelig hos lokal IT-ansvarlig eller hos USIT.

6.3 En bruker har ikke adgang til kopiere programmer ved hjelp av universitetets utstyr utover det som flger av avtaler om disposisjonsrett (lisensavtaler). Det er ikke i noe fall tillatt kopiere lisensbelagte programmer til privat bruk.

7. Tjenestekvalitet og erstatningsansvar

7.1 Brukerne har selv ansvaret for bruk av opplysninger, programmer m.v. som gjres tilgjengelig gjennom IT-tjenestene. Universitetet fraskriver seg ansvar for konomisk tap som flge av feil eller mangler i programmer, data, bruk av opplysninger fra tilgjengelige databaser eller andre opplysninger innhentet gjennom nettet m.v.

8. Lokal IT-ansvarlig og USITs rett til innsyn

8.1 USIT har rett til ske tilgang til den enkelte brukers reserverte omrder i anlegget med sikte p: (1) sikre anleggets funksjonalitet, eller (2) kontrollere at brukeren ikke krenker eller har krenket dette reglements bestemmelser. Det forutsettes at slik tilgang bare skes nr det er av stor betydning for driften eller universitetets ansvar, og kun ved srlig grunn til mistanke. Tillatelse til innsyn i elektronisk post skal skes srskilt. Lokal IT-ansvarlig har ikke adgang til en brukers reserverte omrder uten etter spesiell tillatelse fra Universitetsdirektren.

8.2 Hvis USIT sker slik tilgang, skal det p forhnd innhentes tillatelse fra Universitetsdirektren, med mindre det foreligger srlig tungtveiende grunner til yeblikkelig inngrep. Slike tungtveiende grunner skal dokumenteres overfor Universitetsdirektren i etterkant av inngrepet.

8.3 Hvis bruken av arbeidsstasjon, terminal eller annet sluttbrukerutstyr p grunn av driftssikkerhet eller av andre hensyn overvkes av lokal IT-ansvarlig eller USIT, skal dette opplyses med merke p enheten eller p annen hensiktsmessig mte.

8.4 Lokal IT-ansvarlig og USIT har taushetsplikt med hensyn til opplysninger om brukeren eller brukerens virksomhet som fs p denne mten, med det unntak at forhold som kan representere brudd p reglementet kan meddeles til overordnede instanser.

9. Sanksjoner

9.1 Overtredelse av dette reglementet kan fre til at bruker nektes tilgang til universitetets IT-tjenester, hertil kommer sanksjoner som universitetet etter andre regler kan gjre gjeldende.

9.2 Brudd p regler om personvern, taushetsplikt m.m. kan fre til erstatningsplikt og/eller straff. De vanlige regler om oppsigelse eller avskjed av arbeidstaker eller disiplinrforfyning mot studenter vil kunne ramme misbruker av IT-tjenestene.

Security mechanism:

*** Every practical move you do to ensure the security policy:**

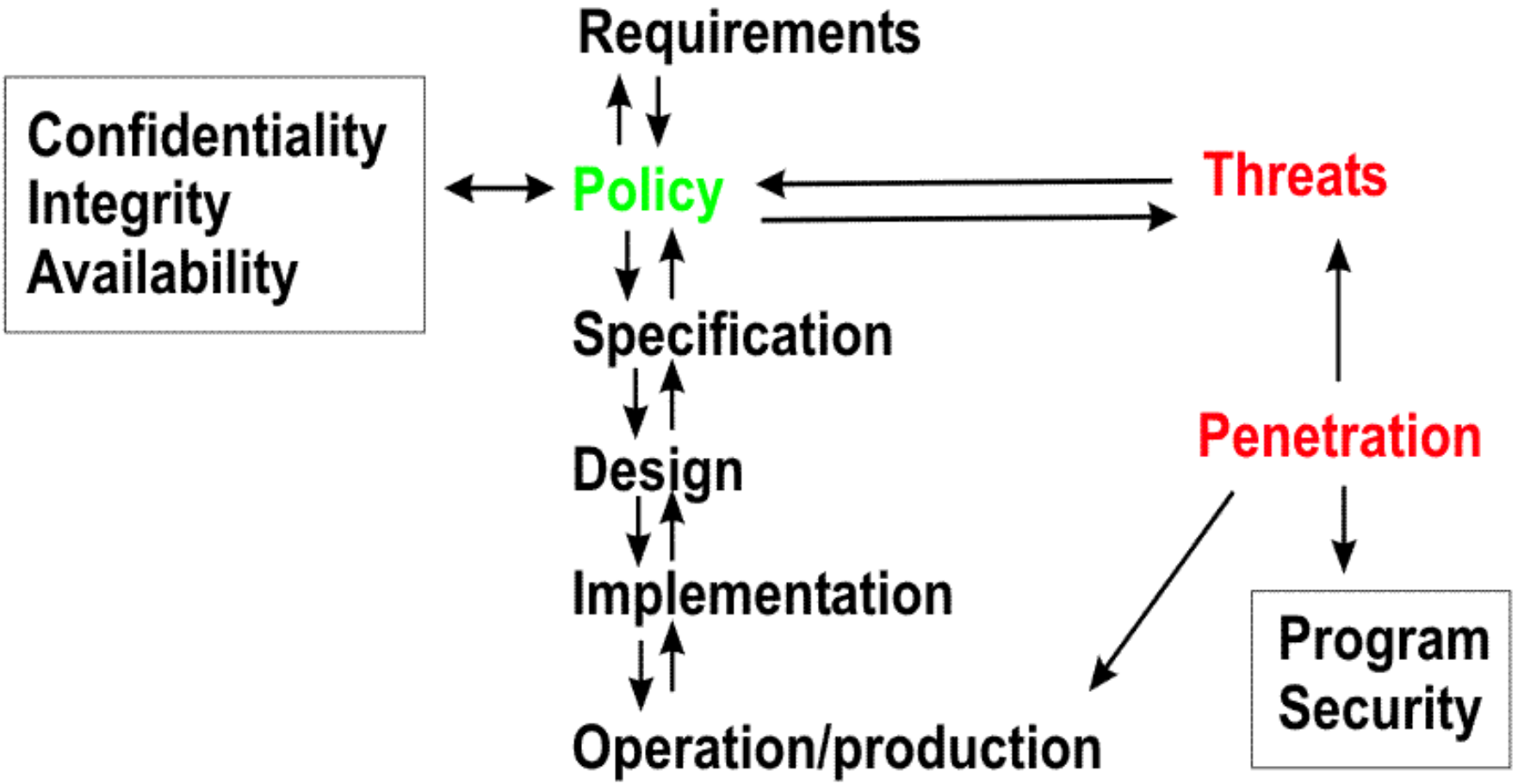
- check a persons ID before change password
- filter some packets (firewall)
- detect anomaly (IDS)
- ...

"Security mechanism: A mechanism that is designed to detect, prevent, or recover from a security attack"
- Stalling

* Security mechanisms may not always fulfill a security policy...

Goal of security mechanisms:
* detect - monitor, log and report.
* prevent - an attack will fail.
* recover - repair any damage caused by the attack

Overview:


Requirements
Confidentiality
Integrity
Availability
Policy
Threats
Specification
Design
Penetration
Implementation
Program
Security
Operation/production

Why?

"What is all the fuss about anyway?"

*"A **threat** is a potential violation of security"* - Bishop

Attack: actions that violate security (policy)

Different types of attack (RFC2828):

*"A '**passive attack**' attempts to learn or make use of information from the system but does not affect system resources."*

*"An '**active attack**' attempts to alter system resources or affect their operation."*

Passive attack

1. Relase of message content: (confidentiality)

Ole

Kari

2. Traffic analysis: (confidentiality)

- observe pattern (even when encrypted!)
- ex: burst of traffic, locations

*** difficult to detect**

*** prevent!**

Active attack

1. Masqurade (spoofing): appears to be from Ole (integrity)

2. Replay: passive capture, delay, then retransmit to Kari (integrity/availability)

Active attack (2)

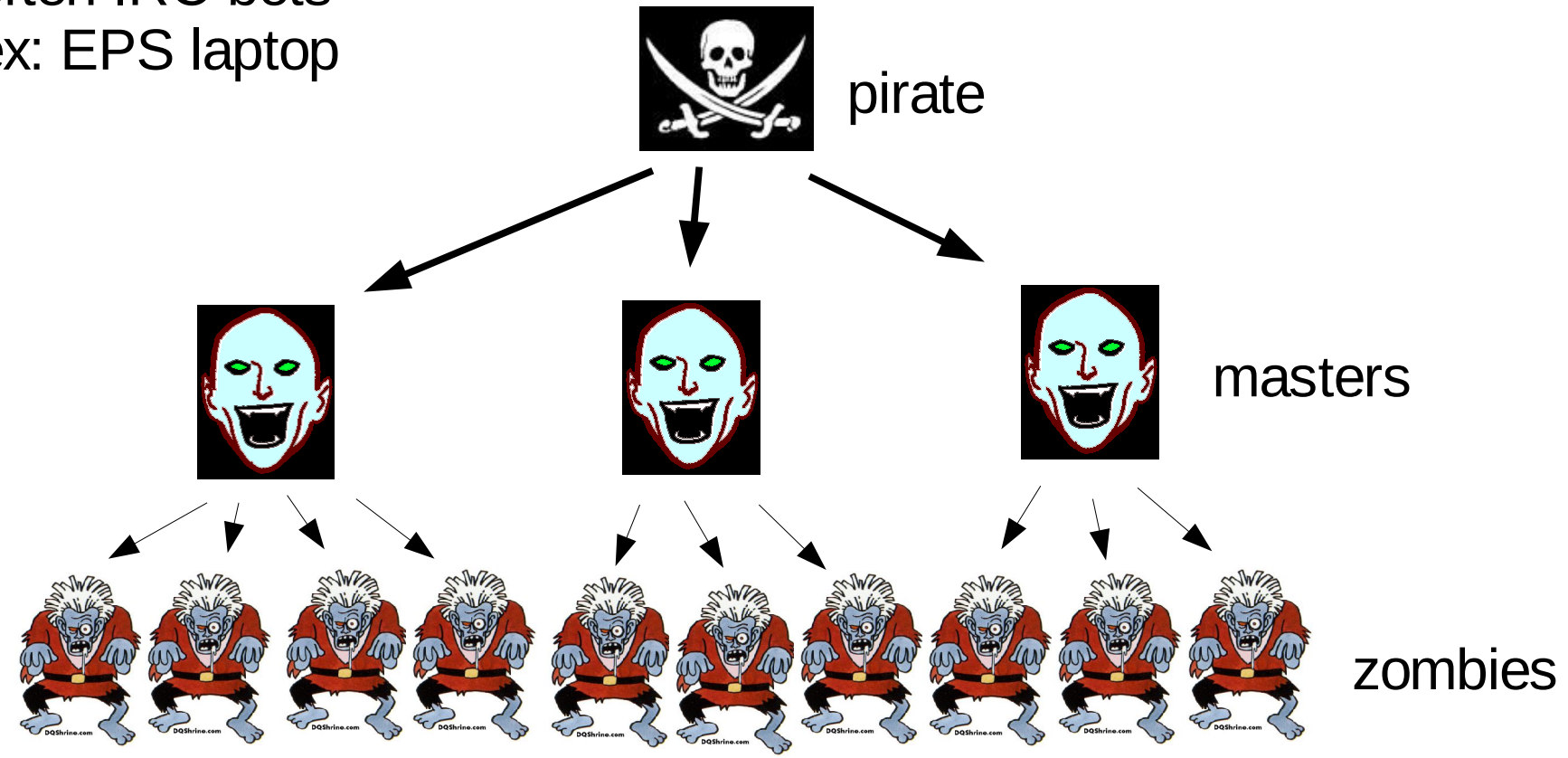
3. Modification of message: (integrity)

4. Denial of Service: (availability)

Distributed Denial of Service:

*** DDoS are common!**
- search google news for “denial of service” ~900 hit

*** How does it work?**
- often IRC bots
- ex: EPS laptop



Example:

* My father complained that his laptop seemed slow/sluggish:
* I started sniffing:

```
NOTICE AUTH :*** Looking up your hostname....
NICK HXDCC-06
USER hellxdcc 32 . :#HellXDCC
MODE HXDCC-06 +i
NOTICE AUTH :*** Checking Ident.
JOIN #HELLXDCC
PING irc.he.net
NOTICE AUTH :*** Found your hostname.
PING irc.he.net
PING irc.he.net
PING irc.he.net
JOIN #HELLXDCC
PING irc.he.net
PING irc.he.net
NOTICE AUTH :*** No Ident response.
:irc.he.net 433 * HXDCC-06 :Nickname is already in use..
:irc.he.net 451 * MODE :Register first..
:irc.he.net 451 * JOIN :Register first..
:irc.he.net 451 * PING :Register first..
ERROR :Closing Link: [hellxdcc@255.255.255.255] (Connection Timed Out).
NICK HXDCC-06
USER hellxdcc 32 . :#HellXDCC
MODE HXDCC-06 +i
NICK HXDCC-06
USER hellxdcc 32 . :#HellXDCC
MODE HXDCC-06 +i
NICK HXDCC-06
USER hellxdcc 32 . :#HellXDCC
MODE HXDCC-06 +i
```

Real-life attacks:

Spoofed packets = fake source address:

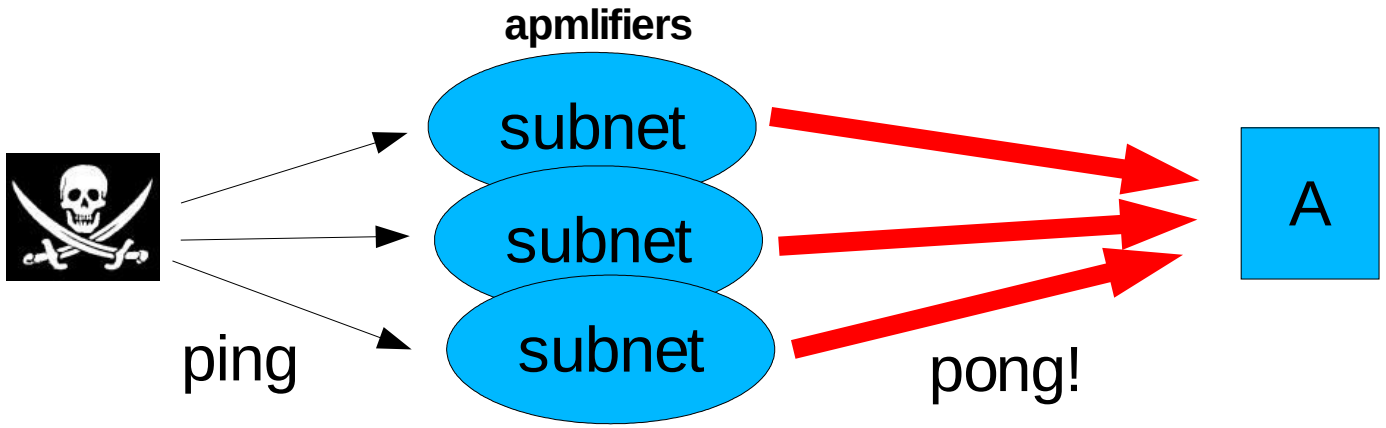
* may spoof on several layers:
  - network layer (ex: Ethernet MAC spoofing)
  - session/application layer (ex: email spoofing)

*** SYN-flood**
  - the archetypal DoS attack
  - send large number of TCP SYN packets to target
  - the target replies with TCP SYN + ACK and wait for response
  - there is no response (the adresses are spoofed!)
  - the target wait for response until timeout
  - must spoof addresses that do NOT reply!
  - SYN-cookies

Real-life attacks (2)

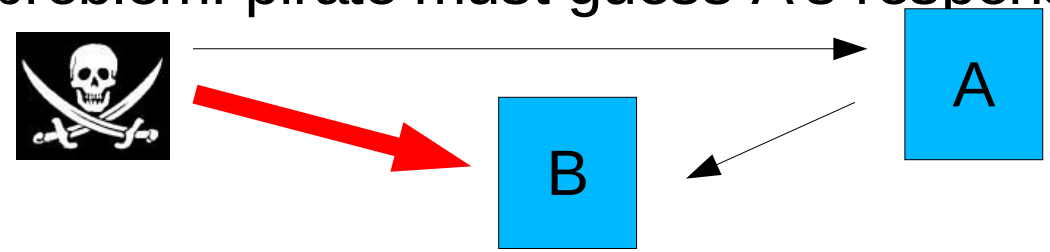
*** Smurf-attack**

- spoofed ICMP echo request (ping) are sent to a subnet's broadcast address
- each active host send reply to the source
- 'open' smurf amplifiers (http://www.powertech.no/smurf/)
- attack tool: nemesis, hping



*** TCP Connection spoofing**

- DoS out (trusted) host B --> B can't reply to A
- pirat connect to host A masqueraded as B
- problem: pirate must guess A's response

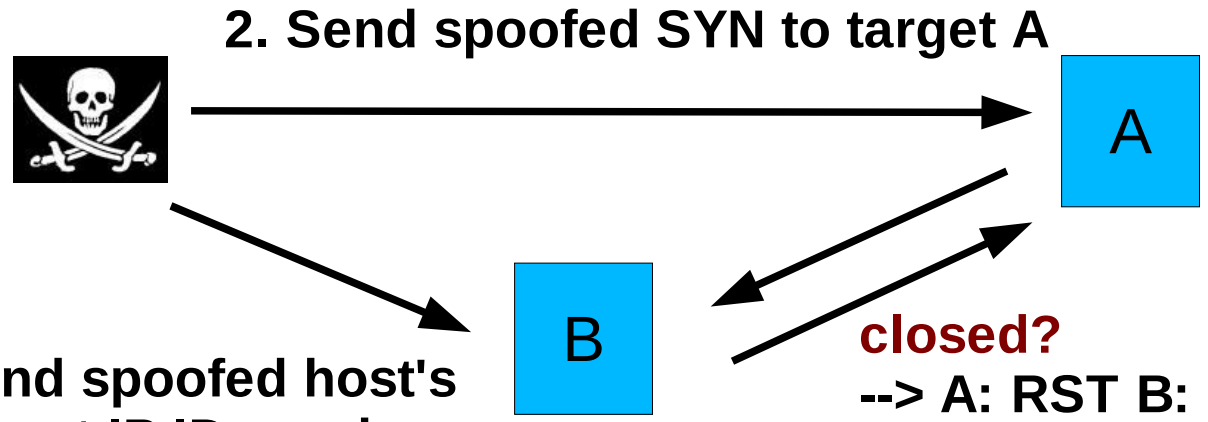


Real-life attacks (3)

*** Bounce Scan**

- scanning computer? - must see the replies!
- spoof a computer on your network segment wait for the reply
  --> difficult today: switched networks

- bounce scan: spoofed packet + indirectly observe the target's replies:
- uses the IP 'identification number' (OpenBSD, heavy traffic)
- (on most systems) increases by one for each packet



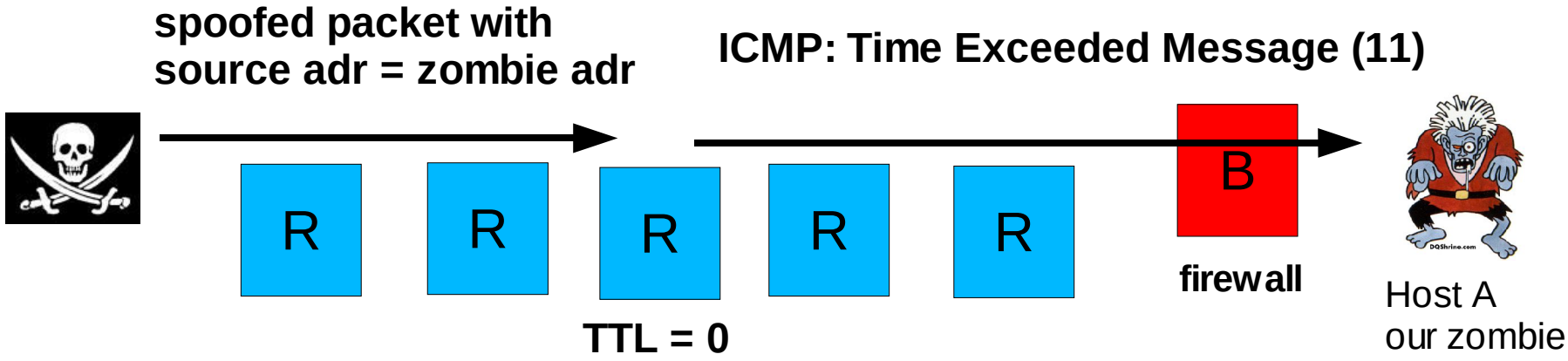
1. find spoofed host's current IP ID number
3. recheck the ID number
- up by one? --> port is closed
- up by two? --> port is open

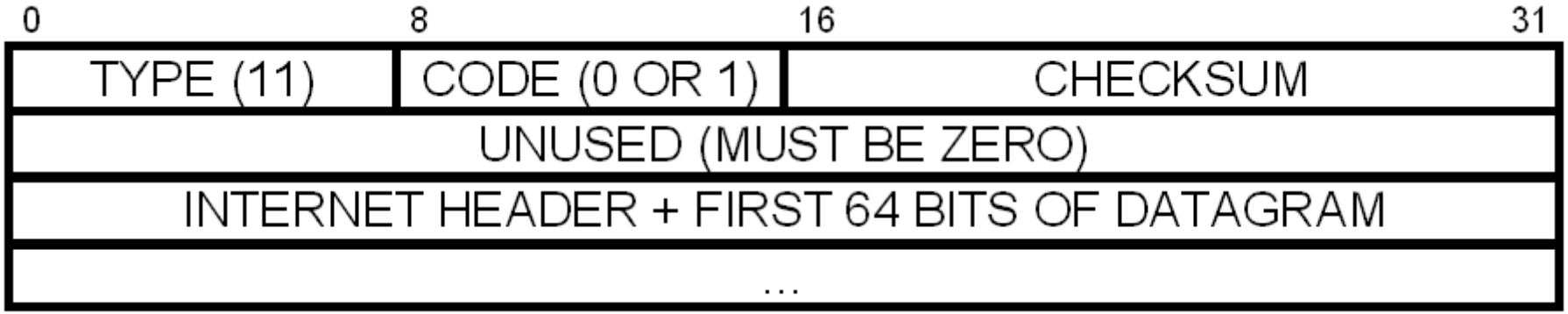
closed?
--> A: RST B: does nothing
open?
--> A: ACK B: RST + increment IP ID

Real-life attacks (4)

*** Zombie Control**

- pirate (masters) send on-way control messages to zombies
- attacker may control zombies without observing the reply and spoof the origin of the control message.
- Trin00, Tribal Flood Network (TFN), TFN2K, Stacheldraht..
- may be sent indirectly:





Protect? Firewall!

“Ok I get the picture – now what?!”

Why a firewall?

- protect against big scary internet!
- protect (the private) network from illegal traffice from the outside
- more (sensitive) data on the (intra)net
- avoid nazi control/update on all hosts

today:

- protect the private network from illegal traffic from the inside!
- what about illegal network traffic **from** the inside?!?
- ban different types of traffic: p2p (~50% of ISP/corp. traffic)
- ban different types of OS (using OpenBSD p0f)

Protect? Firewall (2)

How?
*** Analyse and filter traffic between different networks (typical: internet <--> private)**
eks: ping of death, portscans, worms

past: simple (packet) filters
today: complex functionality (IDS, learning)

Three main functions/types (Hallingstad et. al):
1. Packet Filtering
2. Network Address Translation (NAT)
3. Proxies

*** protect against “given rules” --> new rules?**

Detect: IDS!

* The best intrusion **prevention** may fail.
* second line of defense = IDS

Why? (D. Denning - 1987)

1. Systems has flaws! Fixing all of them is not always technical/economical possible.
2. System with flaws are not easly replaced with more secure systems.
3. Developing systems that are absolutely secure is extremely difficult.
4. The most secure systems are vulnerable to abuse by insiders.

Goal:

*** be accurate and learn**

- **min false positives – authorized users identified as intruders**
- **min false negatives – intruders not identified as intruders**

*** efficient - detect intrusion in a timly fashion**

*** present analysis in simple, easy-to-understand format**

"Intrusion detection is based on the assumption that the behavior of the intruder differs from that of a legetimate users in way that can be quantified" -- Stalling

IDS (2)

Threats that can be adressed by audit trail analysis: (J.P. Anderson – 1980)

* **External penetrators** – not authorized to use the computer. Detect: Observe by auditing failed login attempt (not good enough today! Bufferoverflow in a web-server)

* **Internal penetrators** – who are authorized to use the computer but not for the data, program og resource addressed, including:

i. **Masquraders** – operate under another user's ID and password. Detect: Observing depature from established pattern of use.

ii. **Clandestine users** – who evade auditing and access control, seizes supervisory control. Detect: (difficult) Monitor certain system-wide parameters (CPU, I/O) and compare with historical data.

* **Misfeasors** – authorized users of the computer and resources accessed who misuse their privileges. Detect: (difficult) Establish a priori rules for “socially unacceptable” behavior?

IDS (3)

Types of IDS: (P. Porras 1992)

1. Statistical anomaly detection – collect data of behaviour of legetimate users over a period of time. Apply statistical tests to new behavior.

a. Threshold detection – defining threshold, independent of users.

b. Profile based – a profile for each user is build and used to detect changes in behavior in time

*** tries to define normal expected behaviour**

2. Rule based Detection – define a set of rules that are used to decide that a given behavior is that of an intruder.

a. Anomaly detection – Rules are developed to detect deviation from previous usage patterns.

b. Penetration identification – An expert system that searches for suspicious behavior.

*** tries to define proper behavior**

Real life IDS classification?

“Generic:”
- a more generice IDS framework that may apply to both OS/program/network

“Program/OS:”
- a IDS that check for anomalies on the host's programs and/or OS (system call)

“Network:”
- IDS that check for anomalies using the network (observe packets)

Audit records

* native records (“/var/log/messages”, “Event viewer”)
* detection spesific records:

*** audit record** – some detailed record of (ongoing) activity by users

Example (Denning): Each audit record contains the following:

* Subject – initiator og actions
* Action – operation performed by the subject on the object
* Object – recptors of action. May be files, messages, printers...
* Exception-Condition – any exceptions (write-violations)
* Resource-Usage – the amount used of some resource
* Time-stamp – when did the action took place?

Ex:

```
$ cp sample.py /usr/local/bin/sample2.py
```

Lars	**Execute**	**<library>cp**	**0**	**CPU = 0002**	**1071049142**
Lars	**Read**	**<Lars>sample.py**	**0**	**RECORDS = 0**	**1071049143**
Lars	**Execute**	**<library>cp**	**Write-violation**	**RECORDS = 0**	**1071049144**

Metrics

* To build profiles, we must have som metrics! How to measure?

Three types of metric (Denning):

1. **Event counter** – x is incremented until reset by management action.

Ex: Number of logins during an hour, numbers of file accessed..

2. **Interval timer** – x is the length of time between two related events.

Ex: Length of time between successive login...

3. **Resource Measure** – x is the quantity of resources consumed by some action during a period as specified in the Resource-Usage field.

Ex: Pages printed, CPU time consumed by a program..

Statistical Models

* Given metric, various test can be performed to determine whether current activity fits within acceptable limits.

Models (Denning):

1. **Operational model:** everything outside this threshold = abnormal.

2. **Mean and standard deviation model** of a parameter over some time. May “learn” threshold pr users.

3. **Multivariate model:** Same as 2, except that it is based on correlation between two or more metrics. CPU time vs. I/O.

4. **Markov process model:** establish transition probabilities among various states.

5. **Time series model:** the order and interarrival times of observations.

Profiles

"A profile is a description of the normal behavior of a user with respect to a particular measure." **-- Lunt/Jagannathan**

*** An activity profile contains information that:**

- contains 10 components (variable-type, threshold, action-pattern….)
- identifies the statistical model
- identifies the metric of a variable
- set of audit events measured by the variable

- new audit record --> abnormal?

- **yes: issue alarm**
- **no: update profile**

- new user!

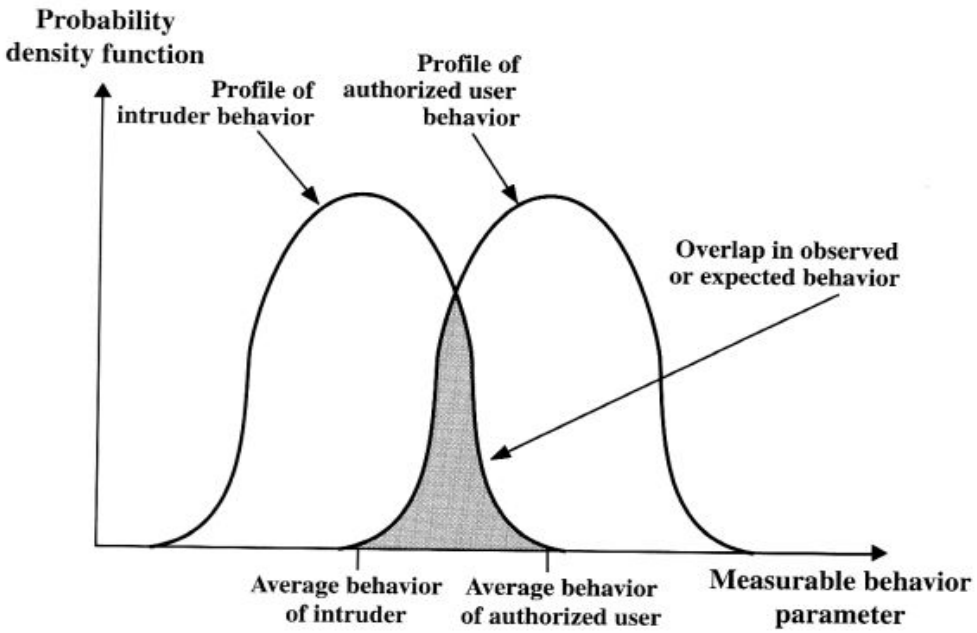
- **profile from template --> contain useless data?**
- **"blank" profile --> evil new user?**

Profiles (2)

*** main advantage:**
- prior knowledge of security flaws is not required
- the detector program “learns” what is normal behavior and then looks for deviations (but must continue to learn!)

*** disadvantage:**
- a user may “learn” the system to think an attack is normal.
- compute intensitive
- “erratic” users

* accuracy (more data) vs. less intruders detected:



Rule based

Rule based anomaly:
"the future will be like the past" -- Stallings

* historical audit records are analyzed --> usage pattern
* generate automatically rules that describe those pattern
* current behavior --> match any rule?
* large database of rules needed to be effective!
  - anywhere from 1000 to 10^6 rules

Rule based penetration identification (expert system):
* have signatures for know penetrations/exploits
* "catch up" with the intruders

--> Suspicion rating = enough rules broken = alarm!
*** lack flexibility = any number of sequences of rules**

IDES

*** Intrusion Detection Expert System (IDES)**

- rule based expert system trained to detect know malicious activity
- based on model described by Denning (1987)

* Lunt/Jagannathan: implement a prototype IDES described by Denning

- written prototype in C, Orcale database
- hardware used: Sun-3/60, Sun-3/260 w/ 560mb disk
- one of the first working IDS?

“The prototype is able to report an anomally usually within a minute from when the user caused the anomaly at the target system. These statistics were obtained by analyzing several thousand audit records” -- Lunt/Jagannathan

--> TO SLOW!!! “chown 000 /” “rm -rf /”

*** SRI: IDES --> NIDES --> EMERALD**

Sequential pattern

*** Model: Adaptive real-time anomaly detection using inductively generated sequential patterns**

*** learning: discover repeating pattern --> generalize --> hypotheses**

*** must have a attack free learning period**

*** high quality hypotheses:**
- **high accuracy in prediction**
- **high level of confidence**

*** input data = epsiodes --> sequences of events --> event is a “snapshot” of a process**

Event?

- consist of: time, type (file access, program use), object name, the object beeing accessed, status, process ID...

Sequential pattern (2)

*** try to predict a future event given a set of other events.**

Profile:
*** consist of security events rules**
*** each rule describe a sequential pattern that predicts the next possible event:**

A – B – C – S – T – S – T – A – B – C – A – B – C

R1: A – B --> (C, 100%)
R2: C --> (S, 50%; A, 50%)
R3: S --> (T, 100%)
R4: T --> (A, 50%; S, 50%)

* may be more complex involving mulitply attributes!

Data mining

* generalise from both **know attack** and **normal behavior** to detect new unknow attack:

"Data mining seeks association rules that estimate the expectation of observing a particular item in a transaction given the occurrence of a particular itemset in the transaction." -- A. Valdes

Three problems with data mining system:

*** acuracy** – they have a **higher false positive** rates (false alarms) --> Not every anomaly is a evil! Signature based better?!

*** inefficient** – **computationally expensive**. Not in real-time..

*** training** – require **large amount of training..**

Data mining (2)

Accuracy:
- assumes: normal activities may be distinguished from intrusive.

--> this “evidence” gets extraced from raw audit data using “a set of data mining algorithms” and call them features.

* “Specialized data mining programs” are used to extract patterns between features.

“more grounded up from emperical data and is thus more objective than expert knowledge” -- Lee et. al. ==> is it really?!?

Data mining (3)

Efficieny:
* Have cost assigned to each feature:

Level 1 features – computed from the first packet. **Cost 1**
Level 2 features – can be computed any point during the life of the connection. **Cost 5**
Level 3 features – can be computed at the end of the connection, using only information about the connection (bytes sent..). **Cost 10**
Level 4 features – can computed at the end of the connection and require access to the data. **Cost 100**

*** level 1 cost rules are computed first,**
- not enough info? --> level 2...

*** (Lee et. al.) reduced the computational cost by 97%**

Data mining (4)

Usability/training:

* require large set of data to train
    - difficult if many different types of data
    - once analyzed, models need to be updated
    - difficult to deploy because they need large set of clean data

* detect new attack without the need for labelling data (as either "clean" or "attack")
    - anomalies are very rare
    - anomalies are different from normal elements
    - *"attack stand out against the background of normal data"* -- Lee et. al. always?

"Because anomalous data and normal data are very different from each other, they do not cluster together." -- Lee et. al.

Program behavior

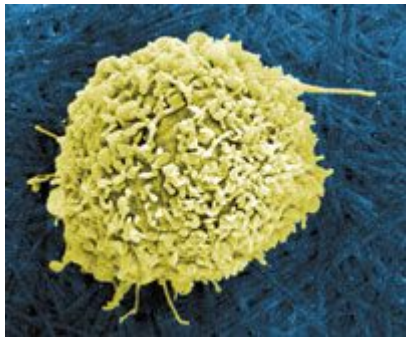
Forrest et. al.: (1998)

Today: IDS vs.

* end-to-end encryption
* high speed network (real time IDS)

*** program misused? --> differs in behavior!**

* range of behavior of privileged processes is limited (compared to behvior of users) and often has a spesific function (qmail)



* capture **the use of system call** during normal use
  - do not consider the source code – just observe
  - the program = “a black box”

*** similarity with natural immune defence: (Forrest/Hofmeyr)**
  - protect highly complex system from penetration
  - must distinguish “self” from “nonself”
  - immune cell: peptide (= short protein fragment that is universal in the body
  - computer system's “peptide” = short sequences of system call

* Related: Tripwire/Aide

Program behavior (2)

* Must have a profile:
    1. scan traces of system calls generated by a particular process
    2. build a database of all uniq sequenses of a given length (k) that occurred during the trace

* Must have a different database for each program version, configuration, architecture...
* problems with vfork() --> replace existing process with a new one. Must be able to switch database.

Eks running program:

```
open, read, mmap, mmap, open, read, mmap...
```

unique sequences with window size k = 3:

```
open, read, mmap
read, mmap, mmap
mmap, mmap, open
mmap, open, read
open, read, mmap
```

Eks: Sendmail database = 1318 unique sequences of length 10

Program behavior (3)

* Anomaly: look at overlapping sequences of length *k*
* not in database? --> mismatch

* the database can not contain ALL possible variations of normal behavor
    - ideal: one mismatch = anomaly!

* many mismatches? --> stronger anomaly.
* intrusion = often a burst of mismatches

* How to get start data in a production enviroment?
* What response is most appropiate once a possible intruision detected? --> kill process and delete program?

Static analysis

D. Wagner, D. Dean: (2001)

*** Do not consider the program as a "black box"**

1. examine the source code (extract use of system call)
2. precompute a model
3. monitor program
4. check for compliance to the model at runtime

*** What model?**

Trivial model:

* *S* is the set of system call the application can ever make

* for every system call not in *S*:

- kill program and issue alarm

* simple, easy to implement and efficient

* will fail to detect many attacks:

- no ordering - may use system calls S in attack
- many dangerous system calls! (open() - attacker may modify any file)
- scale poorly: large application = huge set of S

Static analysis (2)

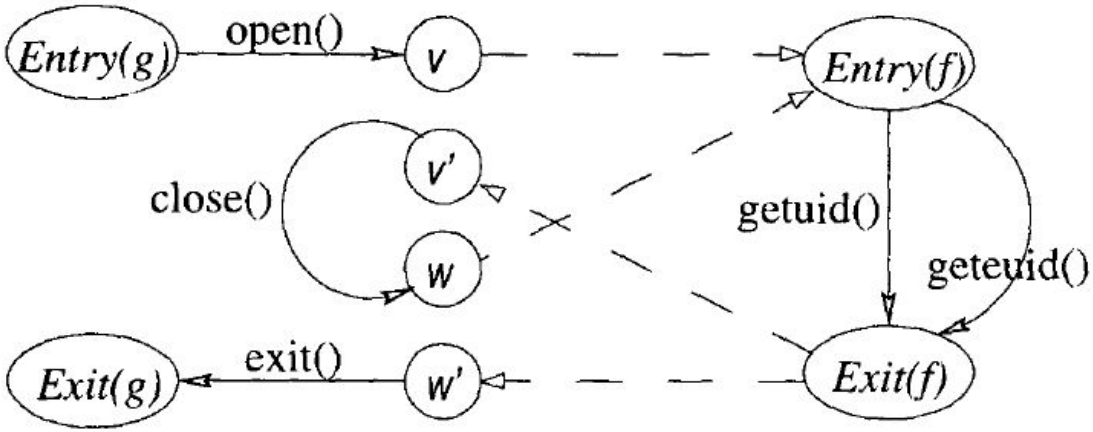
Callgraph:

*** retain (some) ordering of system calls**

*** no data about the internal state of the program**

*** non-deterministic; can not predict wich branch of an if-then expression will be taken at runtime**

```
f(int x) {
   x ? getuid() : getuid();
   x++;
}
g() {
   fd = open('foo', O_RDONLY);
   f(0); close(fd); f(1);
   exit(0);
}
```



Static analysis (3)

* must monitor, that is exploring all possible path, in advance! (caching may be possible)
* imprecision: include impossible paths!
  - cannot occur in any real execution (function calls will always return to correct site)
  - instrusion along impossible paths? = undetected

Abstract stack model:
* monitor the the state of the stack
* uses a 'context-free' language
* computative intesitive

Diagraph model:
* consider windows of consecutive system calls
* efficient, but less precise

Trouble of learning

A. Valdes (2003)

Two approches:

1. Observe a large and varied set of data know (or belived) to be attack free

2. Present work (A. Valdes): usage fall into a number of "modes", which can be represented as patterns in a pattern library

- training data need not to be attack free
- start: cluster of data --> (magic) training algorithm identifies important features in a cluster --> generating a library of class prototype patterns. (may be seeded)

Port pattern (Valdes)

* Port pattern anomaly detection: (Valdes)

* attempt to discover underlying clusters of patterns

* misues detector with a knowledge base (library) encoded as hypotheses and conditional probability

* uses competitive learning and data mining to build “pattern library” = learning on the fly
* total number of patterns should be small (relative to the possible number of possible patterns)
* new learning infrequent (web-server)

* alert generated on activity considered extremly rare = pure anomaly detector
--> what about frequent observed malicious pattern? Script-attack (port-scan)?

* Combine with other IDS component (EMERALD)

Detecting spoofed packets

Methods:

1. Routing based - rely on routers and other network devices to identify traffic or aid detection

- ingress/egress filtering

2. Non-routing – include both active and passive:

a) active – involve probes or methods that cause change in network behavior

- require response from the claimed source

- TTL methods (TTL doesn't change that much) – probe

- IP identification number – should be near in value (problems with OpenBSD)

- OS fingerprint (may be passive or active)

Detecting spoofed packets(2)

- TCP assures reliable transmission (spoofed host will not respond to any packets)

- TCP includes a window size field – change this to zero – the flow should stop, if not --> spoofed

- induce a resynchronization ACK from host being probed --> RST in reply = connection is spoofed

b) passive – observing packets
- learning TTL from host on same subnet
- TCP window size can be predictable

c) administrative – security personnel A contact security personnel B.
- extremely inefficient
- mostly used today?

*** not as a IDS 'per se', but more as a plugin to exisiting IDS**

Example attack

Recent Debian attack:

“On Wednesday 19th November (2003), at approximately 5pm GMT, a sniffed password was used to access an (unprivileged) account on klecker.debian.org”

“Somehow they got root on klecker and installed suckit.” -- James Troup

*** stupid 'super user control kit' - SucKIT (Phrack 58/7):**
- /dev/kmem is our friend
"Mem is a character device file that is an image of the main memory of the computer. It may be used, for example, to examine (and even patch) the system."

“Selecting address is done through lseek(), reading using read() and writing with help of write() ... simple.” -- Phrack

“It can work totally alone (without libs, gcc ...) using only syscalls (this applies only to server side, client is running on your machine, so we can use libc ;)”

Future IDS: Ad-Hoc

*** ad-hoc networks**

- two different manet connect ---> exchange IDS data?
- XML standard!
- secure routing?

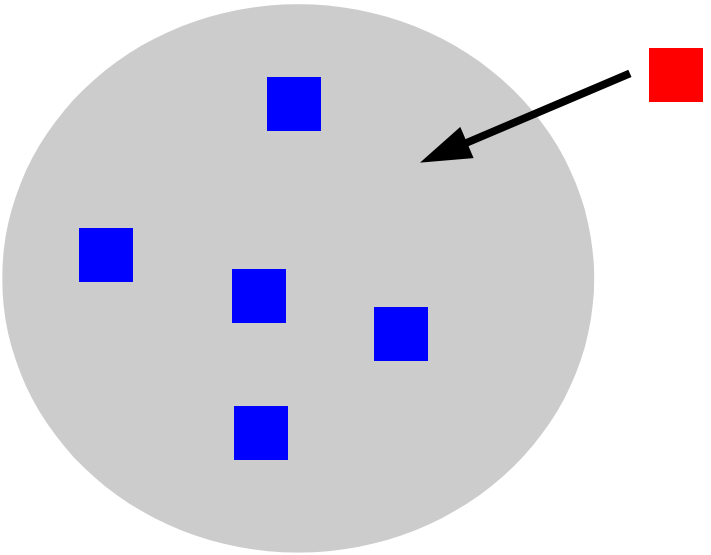
*** mobile 'gadets':**

- scarce computational resource
- limited power resource

==> impose heavy limitations on functionality of an effective IDS!

*** 'Evil hacker' may access the manet any time!**

*** no clear line of defense**
*** no centralized unit**
*** must develop cooperative algorithm**



*** How to trust each other?**

- authenticate?
- only trust data from trusted hosts?

Summary IDS

* basic security
– policy, mechanism

* different attacks – active/passive

* using firewall to protect

IDS to detect:
* misuse intrusion detection
  – what is bad is known (signature)
* anomaly instrusion detection
  - what is usual, is known
  - what is unusual, is bad

* best from both world? --> the hoily grail

* program behavior, static analysis, detecting spoofed packets