UNIT-V

System Security: Intruders - Intrusion Detection - Password Management. Malicious Software: Viruses and Related Threats - Virus Countermeasures. Firewalls: Firewall Design Principles - Trusted Systems - Common Criteria for Information Technology Security Evaluation.

Intruders

- significant issue for networked systems is hostile or unwanted access
- either via network or local
- can identify classes of intruders:
  - masquerader
  - misfeasor
  - clandestine user
- varying levels of competence

Intruders

- clearly a growing publicized problem
  - from “Wily Hacker” in 1986/87
  - to clearly escalating CERT stats
- may seem benign, but still cost resources
- may use compromised system to launch other attacks
- awareness of intruders has led to the development of CERTs

Intrusion Techniques

- aim to gain access and/or increase privileges on a system
- basic attack methodology
  - target acquisition and information gathering
  - initial access
  - privilege escalation
  - covering tracks
- key goal often is to acquire passwords
- so then exercise access rights of owner

Password Guessing

- one of the most common attacks
- attacker knows a login (from email/web page etc)
- then attempts to guess password for it
  - defaults, short passwords, common word searches
  - user info (variations on names, birthday, phone, common words/interests)
  - exhaustively searching all possible passwords
- check by login or against stolen password file
- success depends on password chosen by user
- surveys show many users choose poorly

Password Capture

- another attack involves **password capture**
  - watching over shoulder as password is entered
  - using a trojan horse program to collect
  - monitoring an insecure network login
    - eg. telnet, FTP, web, email
  - extracting recorded info after successful login (web history/cache, last number dialed etc)
- using valid login/password can impersonate user
- users need to be educated to use suitable precautions/countermeasures

Intrusion Detection

- inevitably will have security failures
- so need also to detect intrusions so can
  - block if detected quickly
  - act as deterrent
  - collect info to improve security
- assume intruder will behave differently to a legitimate user
  - but will have imperfect distinction between

Approaches to Intrusion Detection

- statistical anomaly detection
  - threshold
  - profile based
- rule-based detection
  - anomaly
  - penetration identification

Audit Records

- fundamental tool for intrusion detection
- native audit records
  - part of all common multi-user O/S
  - already present for use
  - may not have info wanted in desired form
- detection-specific audit records
  - created specifically to collect wanted info
  - at cost of additional overhead on system

Statistical Anomaly Detection

- threshold detection
  - count occurrences of specific event over time
  - if exceed reasonable value assume intrusion
  - alone is a crude & ineffective detector
- profile based
  - characterize past behavior of users
  - detect significant deviations from this
  - profile usually multi-parameter

Audit Record Analysis

- foundation of statistical approaches
- analyze records to get metrics over time
  - counter, gauge, interval timer, resource use
- use various tests on these to determine if current behavior is acceptable
  - mean & standard deviation, multivariate, markov process, time series, operational
- key advantage is no prior knowledge used

Rule-Based Intrusion Detection

- observe events on system & apply rules to decide if activity is suspicious or not
- rule-based anomaly detection
  - analyze historical audit records to identify usage patterns & auto-generate rules for them
  - then observe current behavior & match against rules to see if conforms
  - like statistical anomaly detection does not require prior knowledge of security flaws

Rule-Based Intrusion Detection

- rule-based penetration identification
  - uses expert systems technology
  - with rules identifying known penetration, weakness patterns, or suspicious behavior
  - compare audit records or states against rules
  - rules usually machine & O/S specific
  - rules are generated by experts who interview & codify knowledge of security admins
  - quality depends on how well this is done

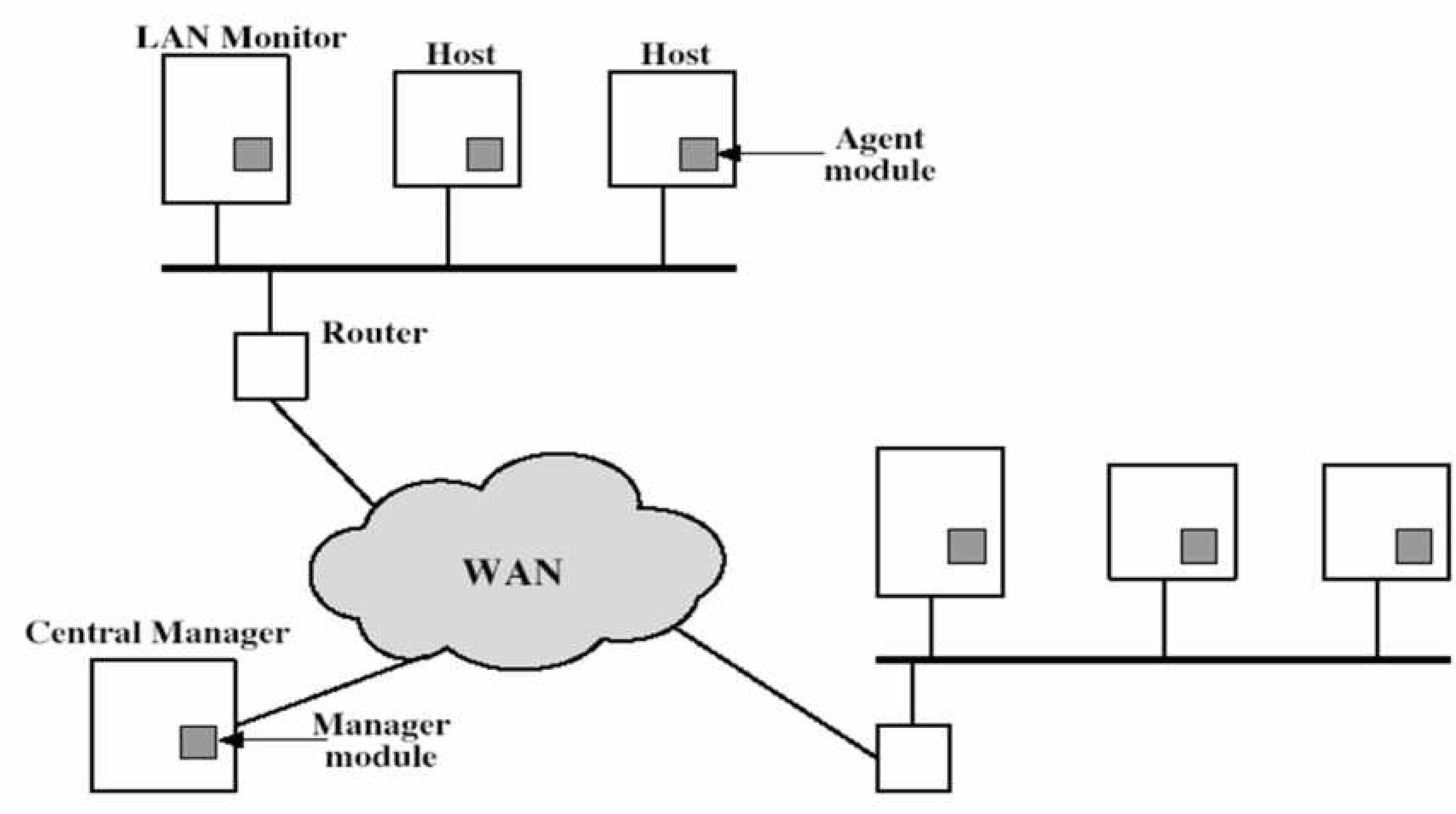
Base-Rate Fallacy

- practically an intrusion detection system needs to detect a substantial percentage of intrusions with few false alarms
  - if too few intrusions detected -> false security
  - if too many false alarms -> ignore / waste time
- this is very hard to do
- existing systems seem not to have a good record

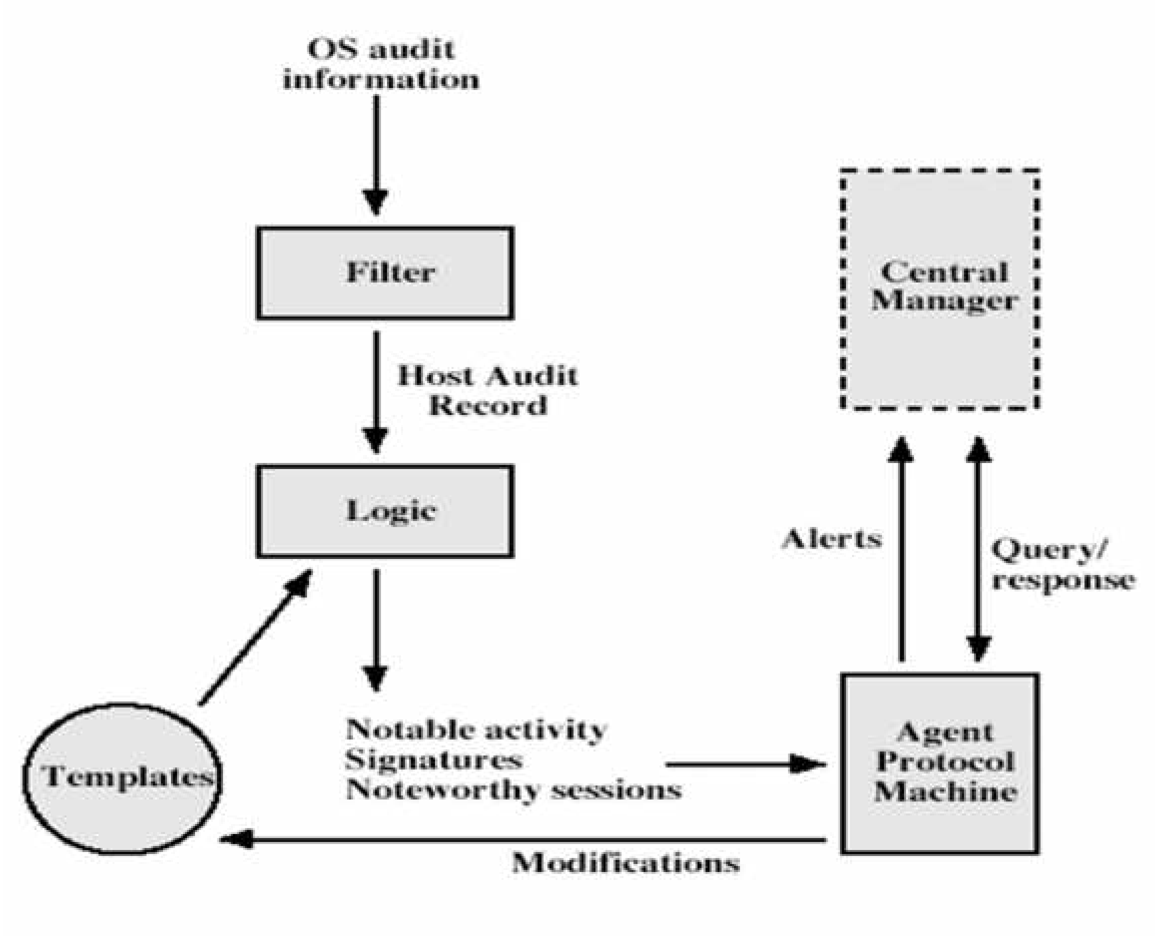
Distributed Intrusion Detection

- traditional focus is on single systems
- but typically have networked systems
- more effective defense has these working together to detect intrusions
- issues
  - dealing with varying audit record formats
  - integrity & confidentiality of networked data
  - centralized or decentralized architecture

Distributed Intrusion Detection - Architecture



Distributed Intrusion Detection – Agent Implementation



Honeypots

- decoy systems to lure attackers
  - away from accessing critical systems
  - to collect information of their activities
  - to encourage attacker to stay on system so administrator can respond
- are filled with fabricated information
- instrumented to collect detailed information on attackers activities
- single or multiple networked systems
- cf IETF Intrusion Detection WG standards

Password Management

- front-line defense against intruders
- users supply both:
  - login – determines privileges of that user
  - password – to identify them
- passwords often stored encrypted
  - Unix uses multiple DES (variant with salt)
  - more recent systems use crypto hash function
- should protect password file on system

Password Studies

- Purdue 1992 - many short passwords
- Klein 1990 - many guessable passwords
- conclusion is that users choose poor passwords too often
- need some approach to counter this

Managing Passwords - Education

- can use policies and good user education
- educate on importance of good passwords
- give guidelines for good passwords
  - minimum length (>6)
  - require a mix of upper & lower case letters, numbers, punctuation
  - not dictionary words
- but likely to be ignored by many users

Managing Passwords - Computer Generated

- let computer create passwords
- if random likely not memorisable, so will be written down (sticky label syndrome)
- even pronounceable not remembered
- have history of poor user acceptance
- FIPS PUB 181 one of best generators
  - has both description & sample code
  - generates words from concatenating random pronounceable syllables

Managing Passwords - Reactive Checking

- reactively run password guessing tools
  - note that good dictionaries exist for almost any language/interest group
- cracked passwords are disabled
- but is resource intensive
- bad passwords are vulnerable till found

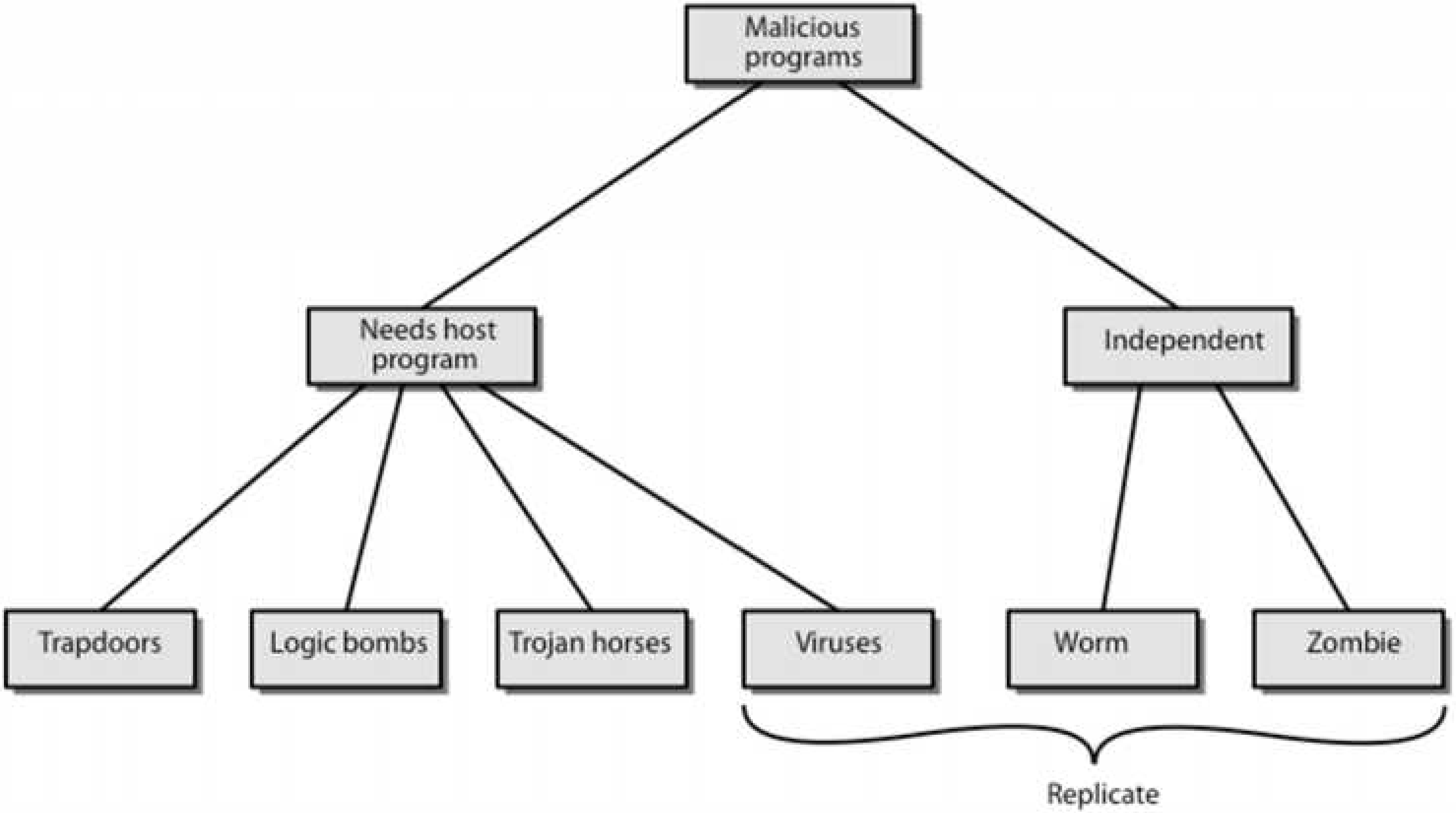
Managing Passwords - Proactive Checking

- most promising approach to improving password security
- allow users to select own password
- but have system verify it is acceptable
  - simple rule enforcement (see earlier slide)
  - compare against dictionary of bad passwords
  - use algorithmic (markov model or bloom filter) to detect poor choices

Viruses and Other Malicious Content

- computer viruses have got a lot of publicity
- one of a family of **malicious software**
- effects usually obvious
- have figured in news reports, fiction, movies (often exaggerated)
- getting more attention than deserve
- are a concern though

Malicious Software



Backdoor or Trapdoor

- secret entry point into a program
- allows those who know access bypassing usual security procedures
- have been commonly used by developers
- a threat when left in production programs allowing exploited by attackers
- very hard to block in O/S
- requires good s/w development & update

Logic Bomb

- one of oldest types of malicious software
- code embedded in legitimate program
- activated when specified conditions met
  - eg presence/absence of some file
  - particular date/time
  - particular user
- when triggered typically damage system
  - modify/delete files/disks, halt machine, etc

Trojan Horse

- program with hidden side-effects
- which is usually superficially attractive
  - eg game, s/w upgrade etc
- when run performs some additional tasks
  - allows attacker to indirectly gain access they do not have directly
- often used to propagate a virus/worm or install a backdoor
- or simply to destroy data

Zombie

- program which secretly takes over another networked computer
- then uses it to indirectly launch attacks
- often used to launch distributed denial of service (DDoS) attacks
- exploits known flaws in network systems

Viruses

- a piece of self-replicating code attached to some other code
  - cf biological virus
- both propagates itself & carries a payload
  - carries code to make copies of itself
  - as well as code to perform some covert task

Virus Operation

- virus phases:
  - dormant – waiting on trigger event
  - propagation – replicating to programs/disks
  - triggering – by event to execute payload
  - execution – of payload
- details usually machine/OS specific
  - exploiting features/weaknesses

Virus Structure

```
program V :=
    {goto main;
    1234567;
    subroutine infect-executable :=     {loop:
                      file := get-random-executable-file;
                      if (first-line-of-file = 1234567) then goto loop
                      else prepend V to file; }
    subroutine do-damage :=  {whatever damage is to be done}
    subroutine trigger-pulled := {return true if condition holds}
    main: main-program :=      {infect-executable;
                               if trigger-pulled then do-damage;
                               goto next;}
    next:
}
```

Types of Viruses

- can classify on basis of how they attack
- parasitic virus
- memory-resident virus
- boot sector virus
- stealth
- polymorphic virus
- metamorphic virus

Macro Virus

- **macro code** attached to some **data file**
- interpreted by program using file
  - eg Word/Excel macros
  - esp. using auto command & command macros
- code is now platform independent
- is a major source of new viral infections
- blur distinction between data and program files
- classic trade-off: "ease of use" vs "security”
- have improving security in Word etc
- are no longer dominant virus threat

Email Virus

- spread using email with attachment containing a macro virus
  - cf Melissa
- triggered when user opens attachment
- or worse even when mail viewed by using scripting features in mail agent
- hence propagate very quickly
- usually targeted at Microsoft Outlook mail agent & Word/Excel documents
- need better O/S & application security

Worms

- replicating but not infecting program
- typically spreads over a network
  - cf Morris Internet Worm in 1988
  - led to creation of CERTs
- using users distributed privileges or by exploiting system vulnerabilities
- widely used by hackers to create **zombie PC's**, subsequently used for further attacks, esp DoS
- major issue is lack of security of permanently connected systems, esp PC's

Worm Operation

- worm phases like those of viruses:
  - dormant
  - propagation
    - search for other systems to infect
    - establish connection to target remote system
    - replicate self onto remote system
  - triggering
  - execution

Morris Worm

- best known classic worm
- released by Robert Morris in 1988
- targeted Unix systems
- using several propagation techniques
  - simple password cracking of local pw file
  - exploit bug in finger daemon
  - exploit debug trapdoor in sendmail daemon
- if any attack succeeds then replicated self

Recent Worm Attacks

- new spate of attacks from mid-2001
- Code Red - used MS IIS bug
  - probes random IPs for systems running IIS
  - had trigger time for denial-of-service attack
  - 2nd wave infected 360000 servers in 14 hours
- Code Red 2 - installed backdoor
- Nimda - multiple infection mechanisms
- SQL Slammer - attacked MS SQL server
- Sobig.f - attacked open proxy servers
- Mydoom - mass email worm + backdoor

Worm Techology

- multiplatform
- multiexploit
- ultrafast spreading
- polymorphic
- metamorphic
- transport vehicles
- zero-day exploit

Virus Countermeasures

- best countermeasure is prevention
- but in general not possible
- hence need to do one or more of:
  - **detection** - of viruses in infected system
  - **identification** - of specific infecting virus
  - **removeal** - restoring system to clean state

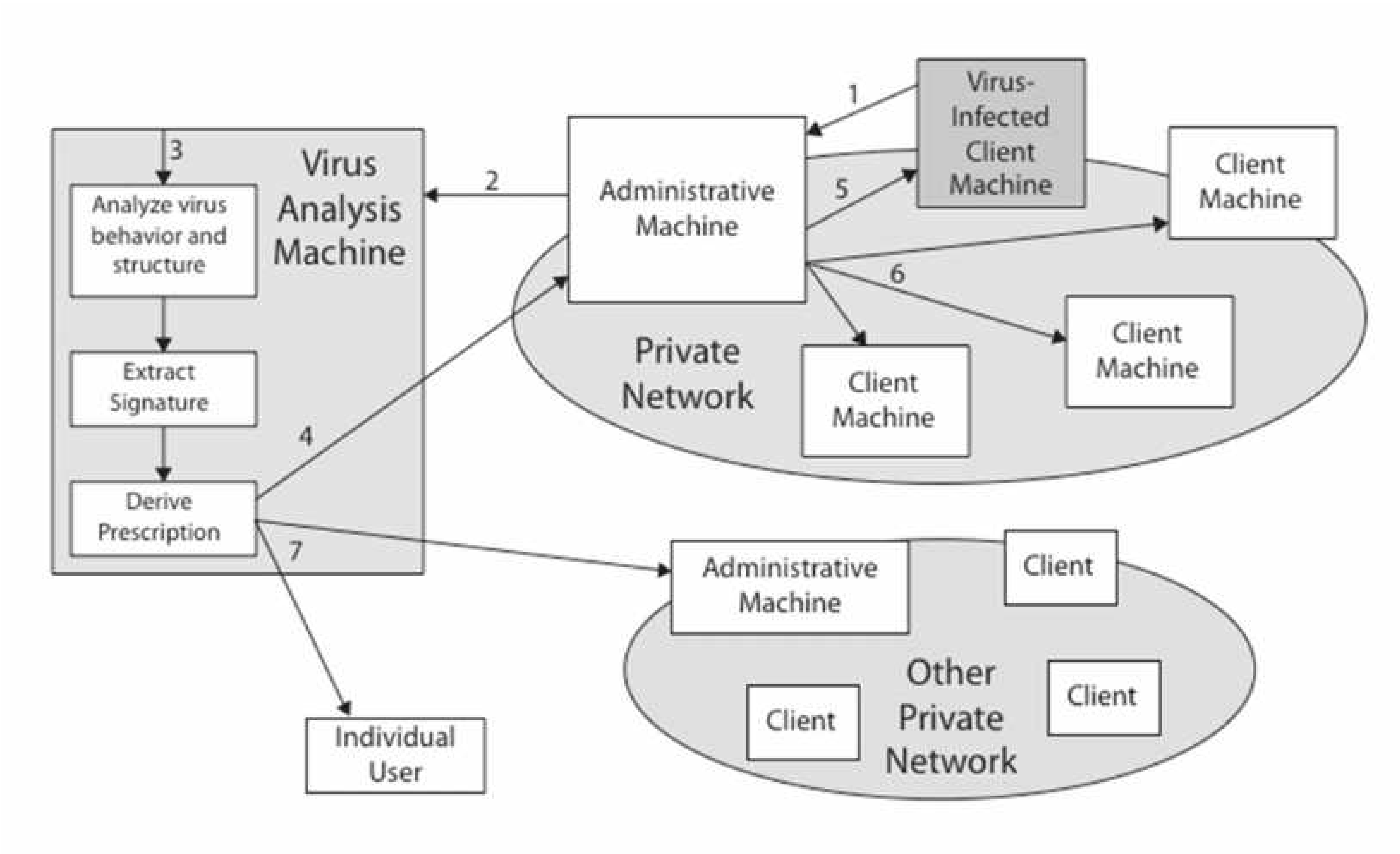
Anti-Virus Software

- **first-generation**
  - scanner uses virus signature to identify virus
  - or change in length of programs
- **second-generation**
  - uses heuristic rules to spot viral infection
  - or uses crypto hash of program to spot changes
- **third-generation**
  - memory-resident programs identify virus by actions
- **fourth-generation**
  - packages with a variety of antivirus techniques
  - eg scanning & activity traps, access-controls
- arms race continues

Advanced Anti-Virus Techniques

- generic decryption
  - use CPU simulator to check program signature & behavior before actually running it
- digital immune system (IBM)
  - general purpose emulation & virus detection
  - any virus entering org is captured, analyzed, detection/shielding created for it, removed

Digital Immune System



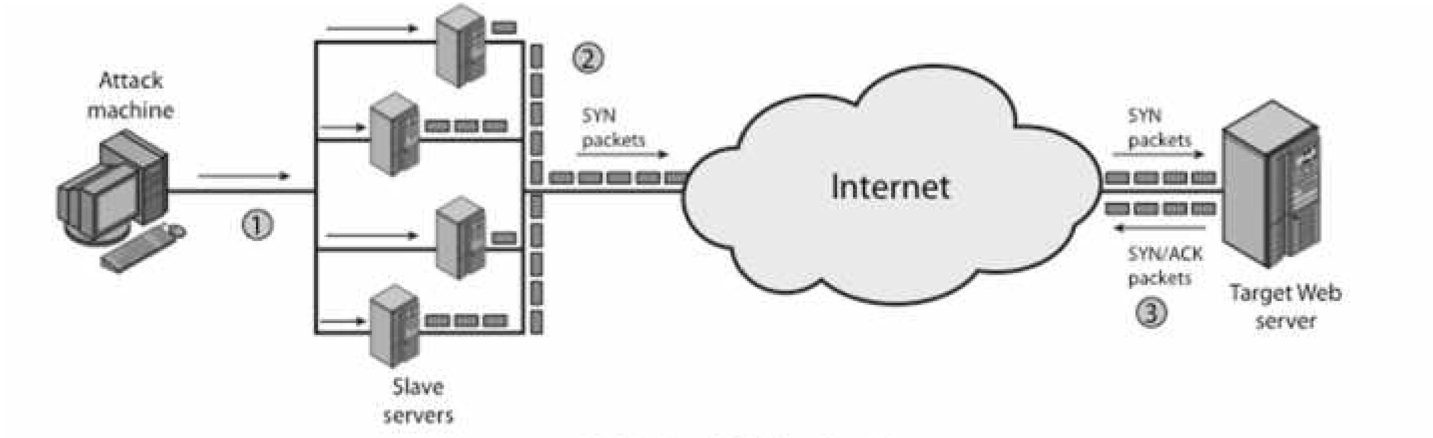
Behavior-Blocking Software

- integrated with host O/S
- monitors program behavior in real-time
  - eg file access, disk format, executable mods, system settings changes, network access
- for possibly malicious actions
  - if detected can block, terminate, or seek ok
- has advantage over scanners
- but malicious code runs before detection

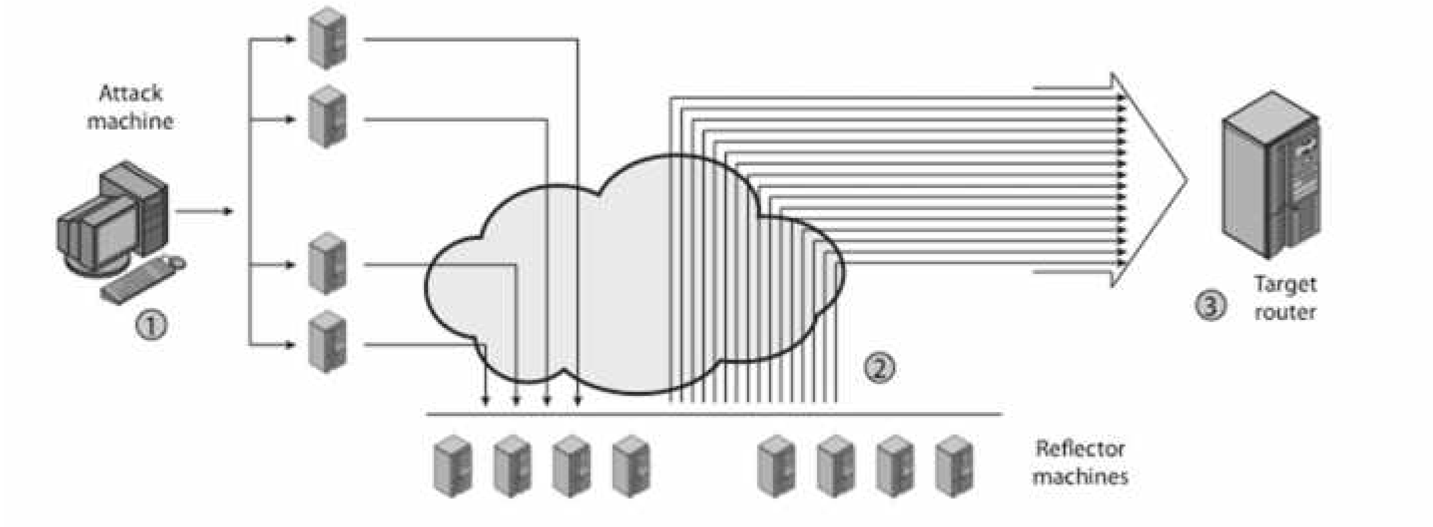
Distributed Denial of Service Attacks (DDoS)

- Distributed Denial of Service (DDoS) attacks form a significant security threat
- making networked systems unavailable
- by flooding with useless traffic
- using large numbers of “zombies”
- growing sophistication of attacks
- defense technologies struggling to cope

Distributed Denial of Service Attacks (DDoS)



(a) Distributed SYN flood attack



(a) Distributed ICMP attack

Contructing the DDoS Attack Network

- must infect large number of zombies
- needs:

1. software to implement the DDoS attack
2. an unpatched vulnerability on many systems
3. scanning strategy to find vulnerable systems
   - random, hit-list, topological, local subnet

DDoS Countermeasures

- three broad lines of defense:
    1. attack prevention & preemption (before)
    2. attack detection & filtering (during)
    3. attack source traceback & ident (after)
- huge range of attack possibilities
- hence evolving countermeasures

What is a Firewall?

- a **choke point** of control and monitoring
- interconnects networks with differing trust
- imposes restrictions on network services
  - only authorized traffic is allowed
- auditing and controlling access
  - can implement alarms for abnormal behavior
- provide NAT & usage monitoring
- implement VPNs using IPSec
- must be immune to penetration

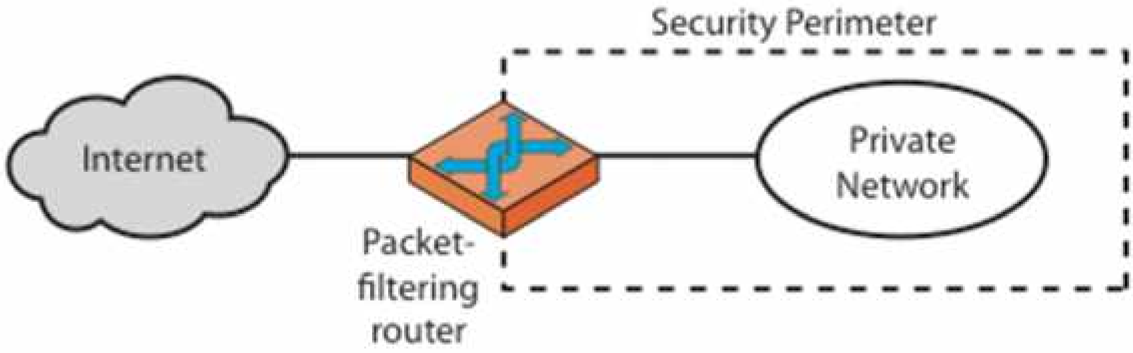
Firewall Limitations

- cannot protect from attacks bypassing it
  - eg sneaker net, utility modems, trusted organisations, trusted services (eg SSL/SSH)
- cannot protect against internal threats
  - eg disgruntled or colluding employees
- cannot protect against transfer of all virus infected programs or files
  - because of huge range of O/S & file types

Firewalls – Packet Filters

- simplest, fastest firewall component
- foundation of any firewall system
- examine each IP packet (no context) and permit or deny according to rules
- hence restrict access to services (ports)
- possible default policies
  - that not expressly permitted is prohibited
  - that not expressly prohibited is permitted

Firewalls – Packet Filters



(a) Packet-filtering router

Firewalls – Packet Filters

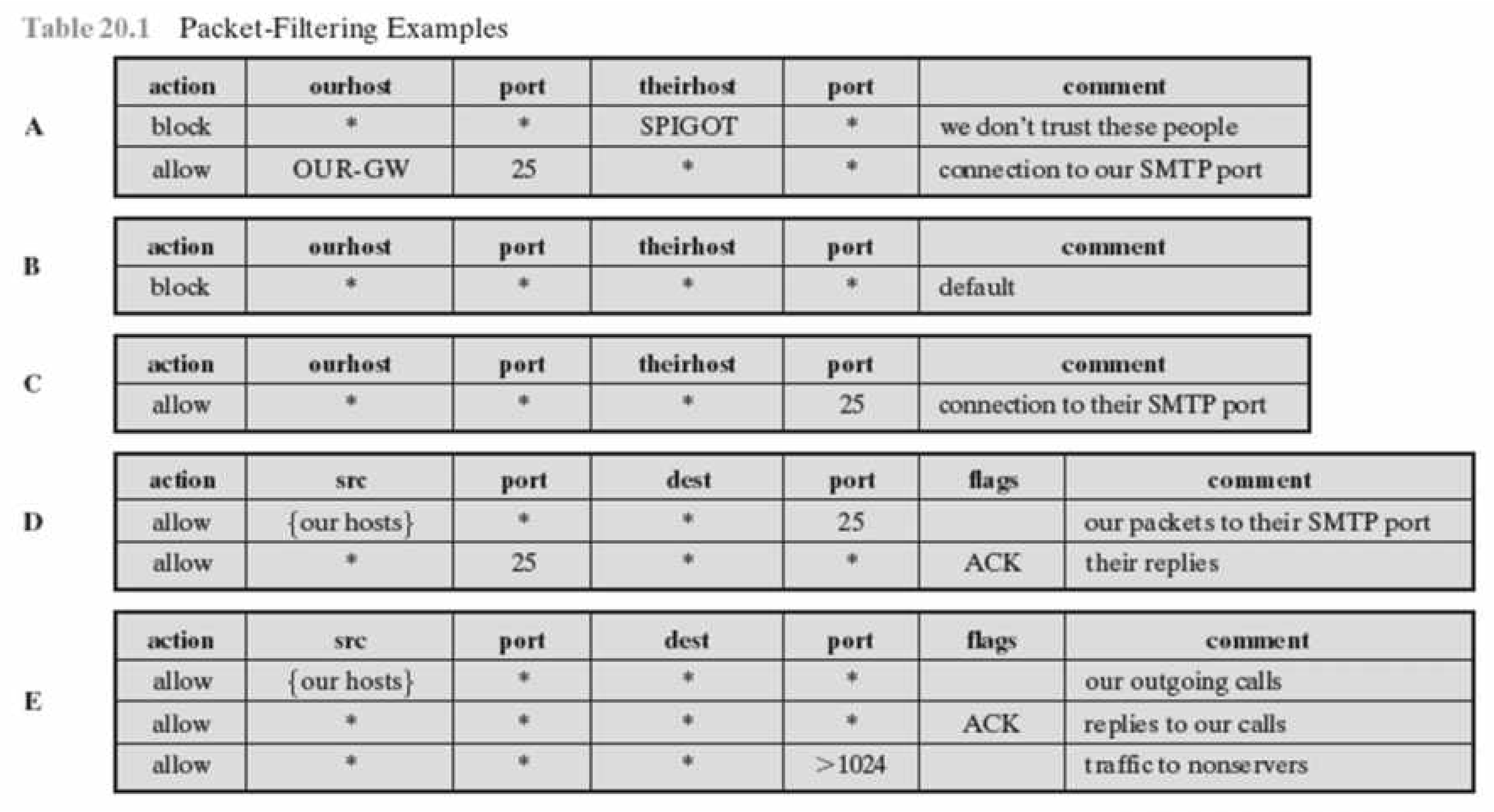


Table 20.1 Packet-Filtering Examples

A

action	ourhost	port	theirhost	port	comment
block	*	*	SPIGOT	*	we don't trust these people
allow	OUR-GW	25	*	*	connection to our SMTP port

B

action	ourhost	port	theirhost	port	comment
block	*	*	*	*	default

C

action	ourhost	port	theirhost	port	comment
allow	*	*	*	25	connection to their SMTP port

D

action	src	port	dest	port	flags	comment
allow	{our hosts}	*	*	25		our packets to their SMTP port
allow	*	25	*	*	ACK	their replies

E

action	src	port	dest	port	flags	comment
allow	{our hosts}	*	*	*		our outgoing calls
allow	*	*	*	*	ACK	replies to our calls
allow	*	*	*	>1024		traffic to nonservers

Attacks on Packet Filters

- IP address spoofing
  - fake source address to be trusted
  - add filters on router to block
- source routing attacks
  - attacker sets a route other than default
  - block source routed packets
- tiny fragment attacks
  - split header info over several tiny packets
  - either discard or reassemble before check

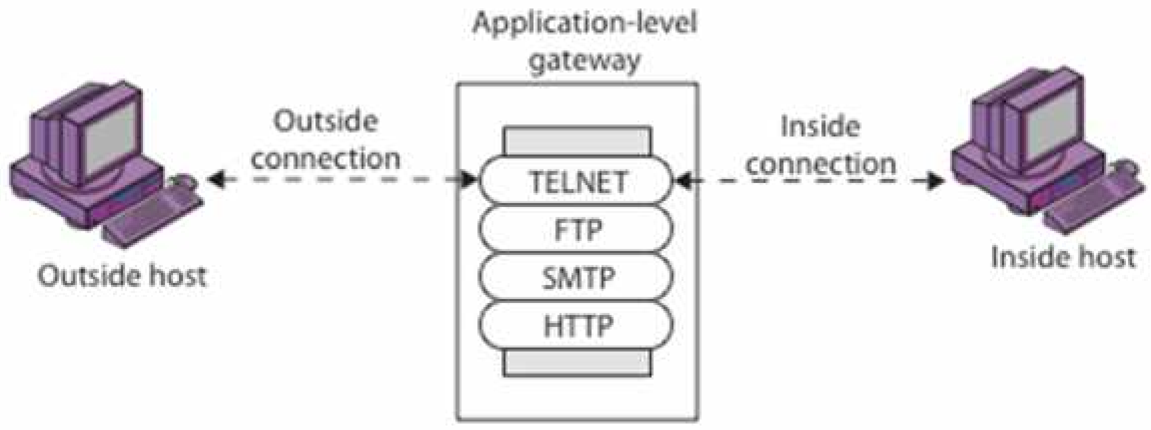
Firewalls – Stateful Packet Filters

- traditional packet filters do not examine higher layer context
  - ie matching return packets with outgoing flow
- stateful packet filters address this need
- they examine each IP packet in context
  - keep track of client-server sessions
  - check each packet validly belongs to one
- hence are better able to detect bogus packets out of context

Firewalls - Application Level Gateway (or Proxy)

- have application specific gateway / proxy
- has full access to protocol
  - user requests service from proxy
  - proxy validates request as legal
  - then actions request and returns result to user
  - can log / audit traffic at application level
- need separate proxies for each service
  - some services naturally support proxying
  - others are more problematic

Firewalls - Application Level Gateway (or Proxy)

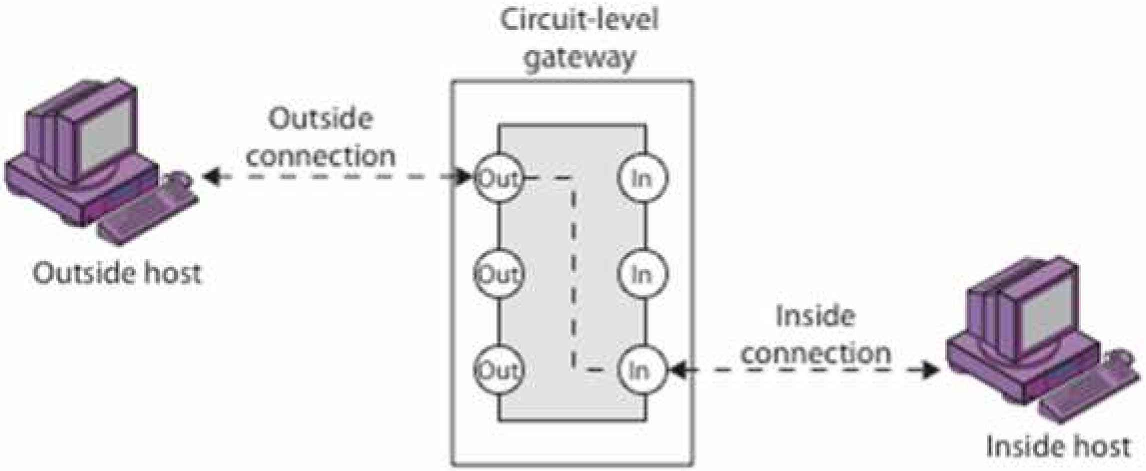


(b) Application-level gateway

Firewalls - Circuit Level Gateway

- relays two TCP connections
- imposes security by limiting which such connections are allowed
- once created usually relays traffic without examining contents
- typically used when trust internal users by allowing general outbound connections
- SOCKS is commonly used

Firewalls - Circuit Level Gateway

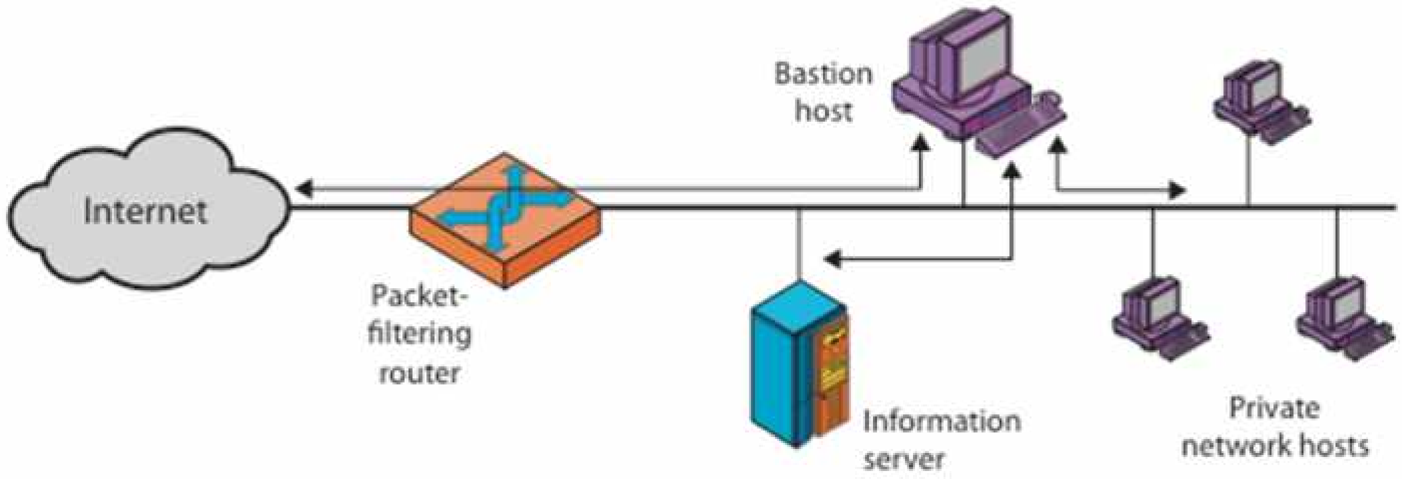


(c) Circuit-level gateway

Bastion Host

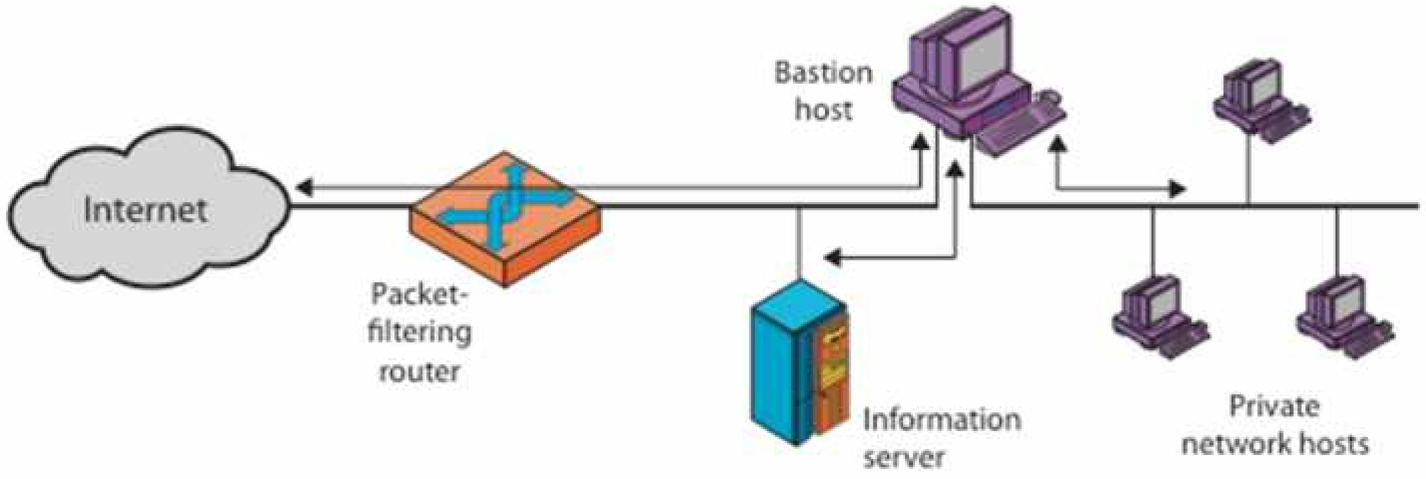
- highly secure host system
- runs circuit / application level gateways
- or provides externally accessible services
- potentially exposed to "hostile" elements
- hence is secured to withstand this
  - hardened O/S, essential services, extra auth
  - proxies small, secure, independent, non-privileged
- may support 2 or more net connections
- may be trusted to enforce policy of trusted separation between these net connections

Firewall Configurations



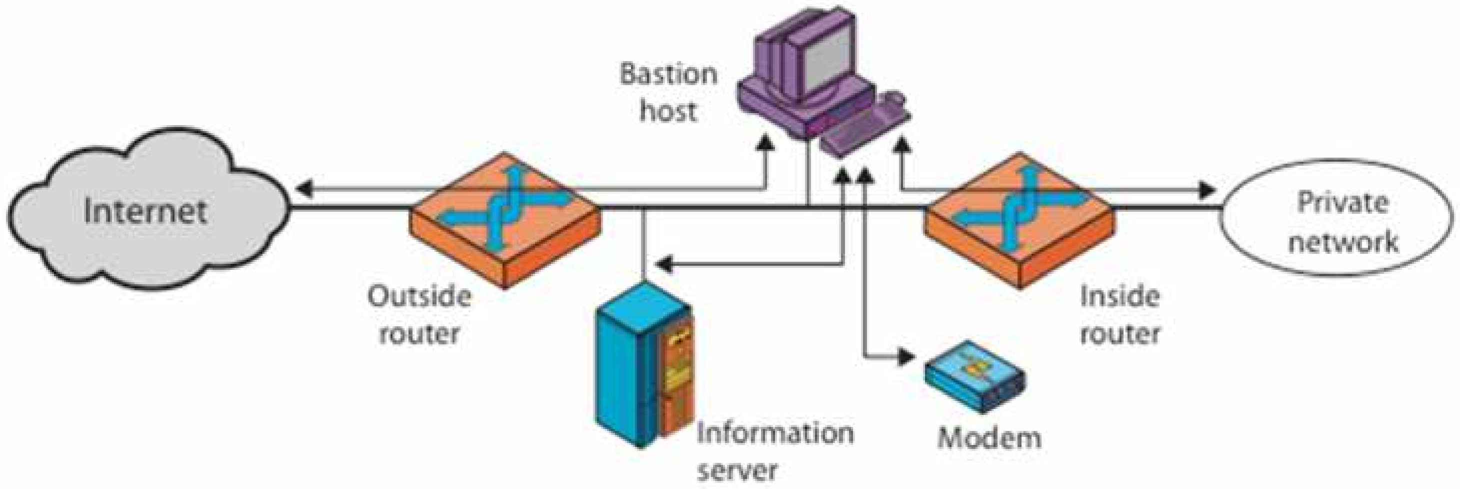
(a) Screened host firewall system (single-homed bastion host)

Firewall Configurations



(b) Screened host firewall system (dual-homed bastion host)

Firewall Configurations

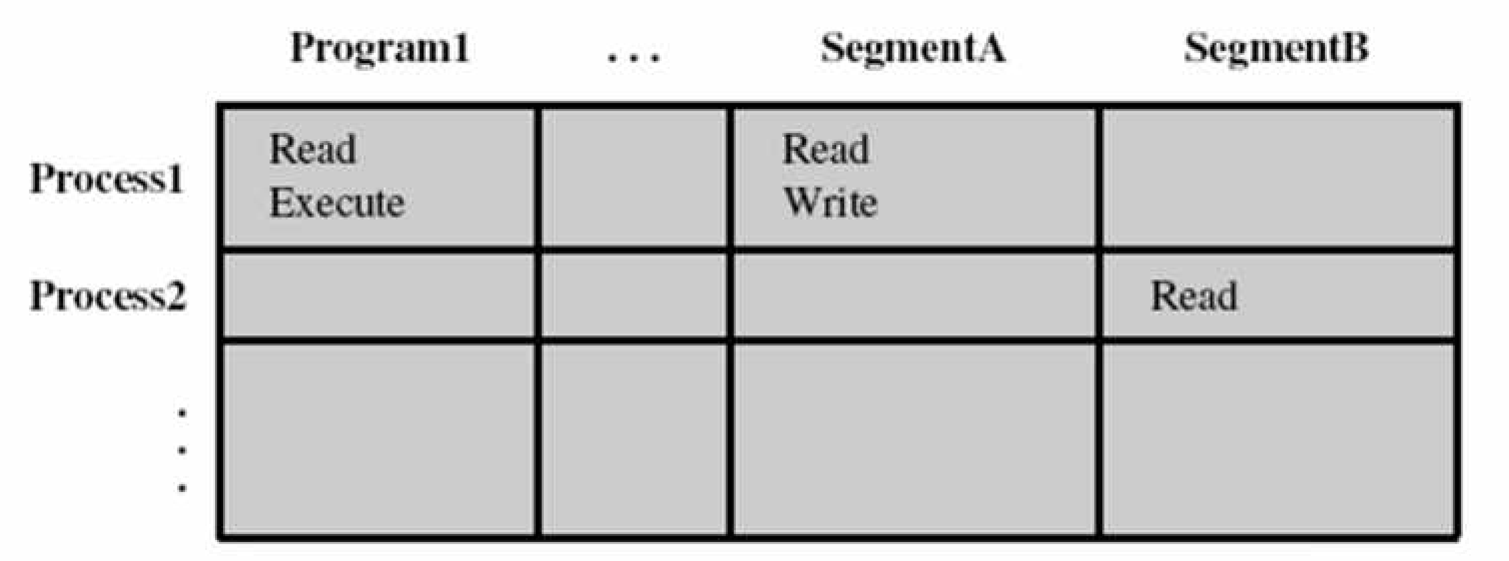


(c) Screened-subnet firewall system

Access Control

- given system has identified a user
- determine what resources they can access
- general model is that of access matrix with
  - **subject** - active entity (user, process)
  - **object** - passive entity (file or resource)
  - **access right** – way object can be accessed
- can decompose by
  - columns as access control lists
  - rows as capability tickets

Access Control Matrix



	Program1	. . .	SegmentA	SegmentB
Process1	Read Execute		Read Write	
Process2				Read
⋮				

(a) Access matrix

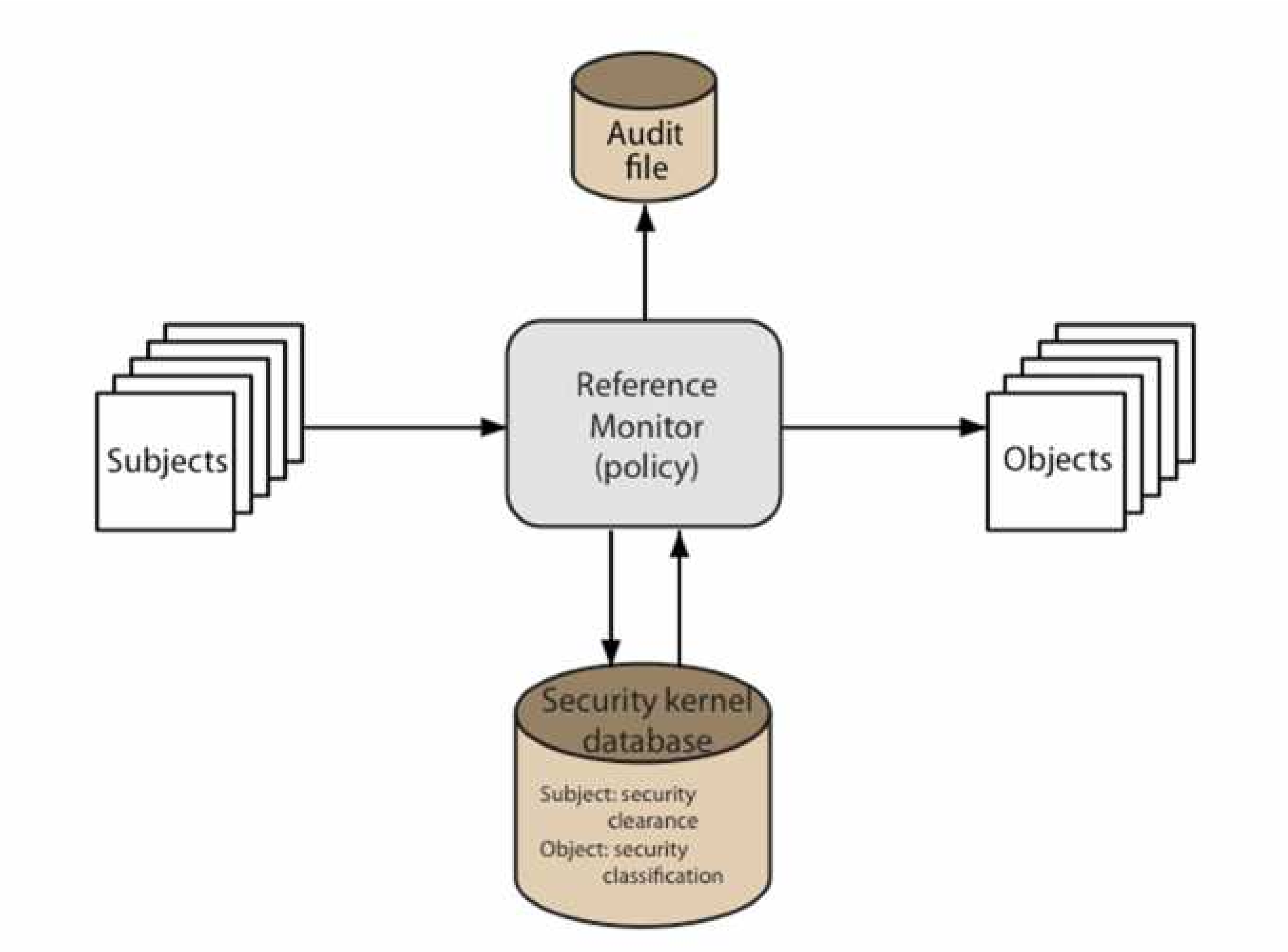
Trusted Computer Systems

- information security is increasingly important
- have varying degrees of sensitivity of information
  - cf military info classifications: confidential, secret etc
- subjects (people or programs) have varying rights of access to objects (information)
- known as multilevel security
  - subjects have **maximum** & **current** security level
  - objects have a fixed security level **classification**
- want to consider ways of increasing confidence in systems to enforce these rights

Bell LaPadula (BLP) Model

- one of the most famous security models
- implemented as mandatory policies on system
- has two key policies:
- **no read up** (simple security property)
  - a subject can only read/write an object if the current security level of the subject dominates (>=) the classification of the object
- **no write down** (*-property)
  - a subject can only append/write to an object if the current security level of the subject is dominated by (<=) the classification of the object

Reference Monitor


Audit file
Subjects
Reference Monitor (policy)
Objects
Security kernel database
Subject: security clearance
Object: security classification

Evaluated Computer Systems

- governments can evaluate IT systems
- against a range of standards:
  - TCSEC, IPSEC and now Common Criteria
- define a number of “levels” of evaluation with increasingly stringent checking
- have published lists of evaluated products
  - though aimed at government/defense use
  - can be useful in industry also

Common Criteria

- international initiative specifying security requirements & defining evaluation criteria
- incorporates earlier standards
  - eg CSEC, ITSEC, CTCPEC (Canadian), Federal (US)
- specifies standards for
  - evaluation criteria
  - methodology for application of criteria
  - administrative procedures for evaluation, certification and accreditation schemes

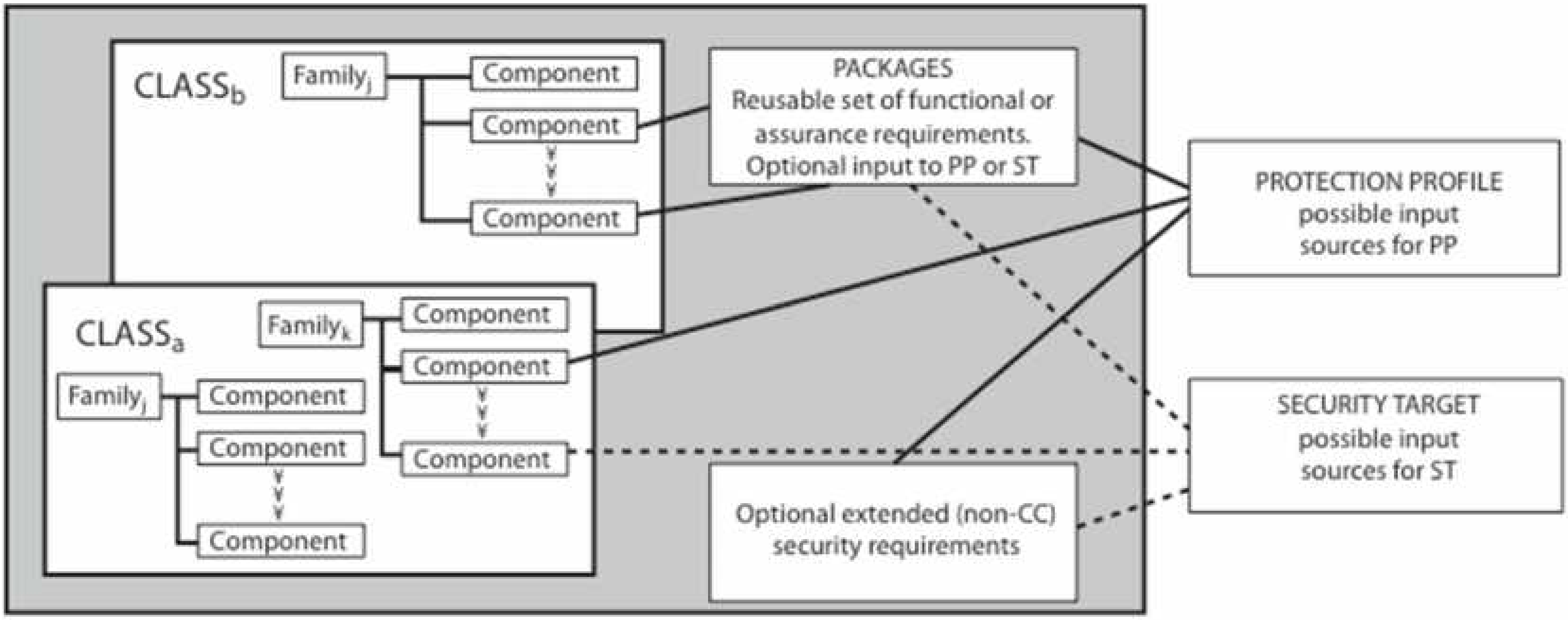
Common Criteria

- defines set of security requirements
- have a Target Of Evaluation (TOE)
- requirements fall in two categories
  - functional
  - assurance
- both organised in classes of families & components

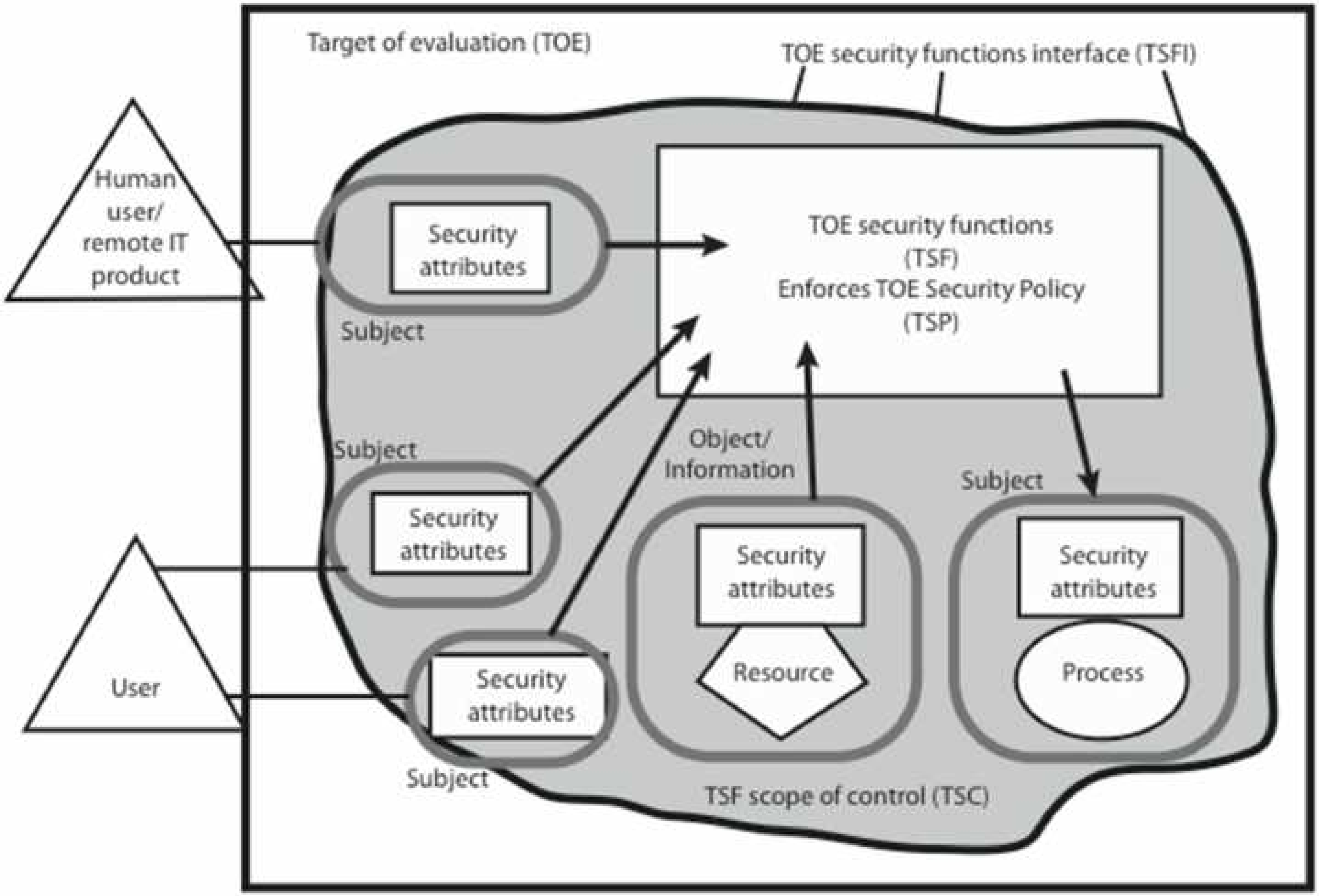
Common Criteria Requirements

- Functional Requirements
  - security audit, crypto support, communications, user data protection, identification & authentication, security management, privacy, protection of trusted security functions, resource utilization, TOE access, trusted path
- Assurance Requirements
  - configuration management, delivery & operation, development, guidance documents, life cycle support, tests, vulnerability assessment, assurance maintenance

Common Criteria



Common Criteria


Target of evaluation (TOE)
TOE security functions interface (TSFI)
Human user/ remote IT product
Security attributes
Subject
TOE security functions (TSF)
Enforces TOE Security Policy (TSP)
Subject
Security attributes
Object/ Information
Subject
User
Security attributes
Security attributes
Resource
Security attributes
Process
Subject
TSF scope of control (TSC)