CS 458 / 658
Computer Security and Privacy

Module 4
Network Security

Spring 2014

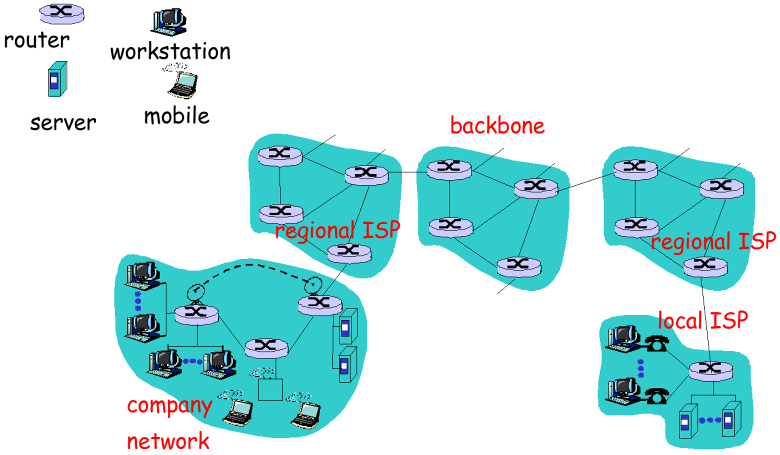
Module outline

1. Network concepts
2. Threats in networks
3. Network security controls
4. Firewalls
5. Honeypots / honeynets
6. Intrusion detection systems

Module outline

1. Network concepts
2. Threats in networks
3. Network security controls
4. Firewalls
5. Honeypots / honeynets
6. Intrusion detection systems

Architecture of the Internet

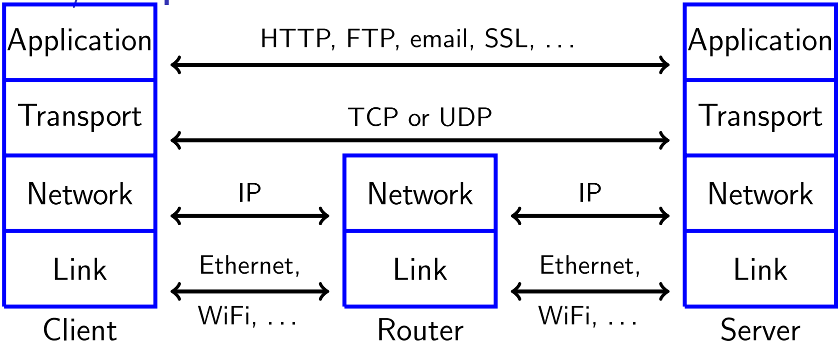


Slide adapted from “Computer Networking” by Kurose & Ross

Characteristics of the Internet

- No single entity that controls the Internet
- Traffic from a source to a destination likely flows through nodes controlled by different entities
- End nodes cannot control through which nodes traffic flows
  - Worse, all traffic is split up into individuals packets, and each packet could be routed along a different path
- Different types of nodes
  - Server, laptop, router, UNIX, Windows,...
- Different types of communication links
  - Wireless vs. wired
- TCP/IP suite of protocols
  - Packet format, routing of packets, dealing with packet loss,...

TCP/IP protocol suite



- Transport and network layer designed in the 1970s to connect local networks at different universities and research labs
- Participants knew and trusted each other
- Design addressed non-malicious errors (e.g., packet drops), but not malicious errors

Module outline

1. Network concepts
2. Threats in networks
3. Network security controls
4. Firewalls
5. Honeypots / honeynets
6. Intrusion detection systems

Port scan

- To distinguish between multiple applications running on the same server, each application runs on a "port"
  - E.g., a Web server typically runs on port 80
- Attacker sends queries to ports on target machine and tries to identify whether and what kind of application is running on a port
- Identification based on loose-lipped applications or how exactly application implements a protocol

Port scan (cont.)

- Loose-lipped systems reveal (non-confidential) information that could facilitate an attack
  - Login application can reveal information about OS or whether a userid is valid
  - Web servers typically return version information
- Nmap tool can identify many applications
  - Useful not only to attackers, but also to system administrators
- Goal of attacker is to find application with remotely exploitable flaw
  - E.g., Apache web server prior to version 1.3.26 is known to be vulnerable to buffer overflow
  - Exploits for these flaws can be found on the Internet

Intelligence

- Social Engineering
  - Attacker gathers sensitive information from person
  - Often, attacker pretends to be somebody within the person's organization who has a problem and exploits the person's willingness to help (or vice versa)
    - I forgot my password, I locked myself out, there's a problem with your Paypal account,...
- Dumpster diving
- Eavesdropping on oral communication
- Google
  - There's lots of information on the Internet that shouldn't be there
  - The right Google query will find it
- Victim's Facebook profile

Eavesdropping and wiretapping

- Owner of node can always monitor communication flowing through node
  - Eavesdropping or passive wiretapping
  - Active wiretapping involves modification or fabrication of communication
- Can also eavesdrop while communication is flowing across a link
  - Degree of vulnerability depends on type of communication medium
- Or when communication is accidentally sent to attacker's node
- It is prudent to assume that your communication is wiretapped

Communication media

- Copper cable
  - Inductance allows a physically close attacker to eavesdrop without making physical contact
  - Cutting cable and splicing in secondary cable is another option
- Optical fiber
  - No inductance, and signal loss by splicing is likely detectable
- Microwave/satellite communication
  - Signal path at receiver tends to be wide, so attacker close to receiver can eavesdrop
- All these attacks are feasible in practice, but require physical expenses/effort

Communication media (cont.)

- WiFi
  - Can be easily intercepted by anyone with a WiFi-capable (mobile) device
    - Don't need additional hardware, which would cause suspicion
  - Maybe from kilometers away using a directed antenna
  - WiFi also raises other security problems
    - Physical barriers (walls) help against random devices being connected to a wired network, but are (nearly) useless in case of wireless network
    - Need authentication mechanism to defend against free riders

Misdelivered information

- Local Area Network (LAN)
  - Connects all computers within a company or a university
  - Technical reasons might cause a packet to be sent to multiple nodes, not only to the intended receiver
  - By default, a network card ignores wrongly delivered packets
  - An attacker can change this and use a packet sniffer to capture these packets
- Email
  - Wrongly addressed emails, inadvertent Reply-To-All

Impersonation

- Impersonate a person by stealing his/her password
  - Guessing attack
  - Exploit default passwords that have not been changed
  - Sniff password (or information about it) while it is being transmitted between two nodes
  - Social engineering
- Exploit trust relationships between machines/accounts
  - Rhosts/rlogin allows user A on machine X to specify that user B on machine Y can act as A on X without having to enter password
    - ssh has a similar mechanism
    - Attacker breaking into machine Y can exploit this
    - Or attacker might be able to masquerade as machine Y

Spoofing

- Object (node, person, URL, Web page, email, WiFi access point,...) masquerades as another one
- URL spoofing
  - Exploit typos: www.uwaterlo.ca
  - Exploit ambiguities: www.foobar.com or www.foo-bar.com?
  - Exploit similarities: www.paypa1.com
- Web page spoofing and URL spoofing are used in Phishing attacks
- "Evil Twin" attack for WiFi access points
- Spoofing is also used in session hijacking and man-in-the-middle attacks

Session hijacking

- TCP protocol sets up state at sender and receiver end nodes and uses this state while exchanging packets
  - e.g., sequence numbers for detecting lost packets
  - Attacker can hijack such a session and masquerade as one of the endpoints
- Web servers sometimes have client keep a little piece of data ("cookie") to re-identify client for future visits
  - Attacker can sniff or steal cookie and masquerade as client

- Man-in-the-middle attacks are similar; attacker becomes stealth intermediate node, not end node

Traffic analysis

- Sometimes, the mere existence of communication between two parties is sensitive and should be hidden
    - Whistleblower
    - Military environments
    - Two CEOs
- TCP/IP has each packet include unique addresses for the packet's sender and receiver end nodes, which makes traffic analysis easy
- Attacker can learn these addresses by sniffing packets
- More on protecting yourself from this attack later

Integrity attacks

- Attacker can modify packets while they are being transmitted
  - Change payload of packet
  - Change address of sender or receiver end node
  - Replay previously seen packets
  - Delete or create packets
- Line noise, network congestion, or software errors could also cause these problems
  - TCP/IP will likely detect environmental problems, but fail in the presence of an active attacker
  - How in the case of TCP's checksumming mechanism?

Integrity attacks (cont.)

- DNS cache poisoning
  - Domain Name System maps host names (www.uwaterloo.ca) to numerical addresses (129.97.128.40), as stored in packets
  - Attacker can create wrong mappings

Protocol failures

- TCP/IP assumes that all nodes implement protocols faithfully
- E.g., TCP includes a mechanism that asks a sender node to slow down if the network is congested
  - An attacker could just ignore these requests
- Some implementations do not check whether a packet is well formatted
  - E.g., the value in the packet's length field could be smaller than the packet's actual length, making buffer overflow possible
  - Potentially disastrous if all implementations are from the same vendor or based on the same code base

Protocol failures (cont.)

- Protocols can be very complex, behaviour in rare cases might not be (uniquely) defined
- Some protocols include broken security mechanisms
  - WEP (see later)

Web site vulnerabilities

- Web site defacements
- Accessing a URL has a web server return HTML code
  - Tells browser how to display web page and how to interact with web server
  - Attacker can examine this code and find vulnerabilities
- Attacker sends malicious URL to web server
  - to exploit a buffer overflow
  - to invoke a shell or some other program
  - to feed malicious input to a server-side script
  - to access sensitive files
    - E.g., by including "../" in a URL or by composing URLs different from the "allowed ones" in the HTML code

Web site vulnerabilities (cont.)

- HTTP protocol is stateless, so web server asks client to keep state when returning a web page and to submit this state when accessing next web page
  - Cookie or URL (http://www.store.com?clientId=4342)
- Attacker can submit modified state information
  - Web server might fall victim to incomplete mediation

Web site vulnerabilities (cont.)

- Cross-site scripting (XSS)/request forgery (CSRF) attacks (code injection)
- Attacker adds his/her own HTML code to somebody else's web page
  - E.g., in the comments section of a blog
- Other users download and execute this code when downloading the web page
  - XSS: Code steals sensitive information (e.g., cookie) contained in the web page and sends it to attacker
    - http://www.attacker.com/aliceCookie=secretValue
  - CSRF: Code performs malicious action at some web site (e.g., user's bank) if user is currently logged in there
    - http://www.bank.com/transferMoneyToAttacker

Denial of service (DoS)

- Cutting a wire or jamming a wireless signal
- Flooding a node by overloading its Internet connection or its processing capacity
- Ping flood
  - Node receiving a ping packet is expected to generate a reply
  - Attacker could overload victim
  - Different from “ping of death”, which is a malformatted ping packet that crashes victim’s computer
- Smurf attack
  - Spoof (source) address of sender end node in ping packet by setting it to victim’s address
  - Broadcast ping packet to all nodes in a LAN

Denial of service (cont.)

- Exploit knowledge of implementation details about a node to make node perform poorly
- SYN flood
  - TCP initializes state by having the two end nodes exchange three packets (SYN, SYN-ACK, ACK)
  - Server queues SYN from client and removes it when corresponding ACK is received
  - Attacker sends many SYNs, but no ACKs
- Send packet fragments that cannot be reassembled properly
- Craft packets such that they are all hashed into the same bucket in a hash table

Denial of service (cont.)

- Black hole attack (AKA packet drop attack)
  - Routing of packets in the Internet is based on a distributed protocol
  - Each router informs other routers of its cost to reach a set of destinations
  - Malicious router announces low cost for victim destination and discards any traffic destined for victim
  - Has also happened because of router misconfiguration
- DNS attacks
  - DNS cache poisoning can lead to packets being routed to the wrong host

Distributed denial of service (DDoS)

- If there is only a single attacking machine, it might be possible to identify the machine and to have routers discard its traffic (see later)
- More difficult if there are lots of attacking machines
- Most might participate without knowledge of their owners
  - Attacker breaks into machines using Trojan, buffer overflow,. . . and installs malicious software
  - Machine becomes a zombie/bot and waits for attack command from attacker
  - A network of bots is called a botnet
  - How would you turn off a (classic) botnet (i.e., one with a central command node)?

New Generation Botnets

- Today's botnets are very sophisticated
- Virus/worm/trojan for propagation, exploit multiple vulnerabilities
- Stealthiness to hide from owner of computer
- Code morphing to make detection difficult
- Bot usable for different attacks (spam, DDoS,...)

Botnets (cont.)

- Distributed, dynamic & redundant control infrastructure
  - P2P system for distributing updates
  - "Fast Flux"
    - A single host name maps to hundreds of addresses of infected machines
    - Machines proxy to malicious websites or to "mothership"
    - Machines are constantly swapped in/out of DNS to make tracking difficult
  - Domain Generation Algorithm
    - Infected machine generates a large set (50,000 in the case of Conficker) of domain names that changes every day
    - It contacts a random subset of these names for updates
    - To control the botnet, authorities would have to take control of 50,000 different domain names each day

Botnets (cont.)

- Earlier worms (Nimda, slammer) were written by hackers for fame with the goal to spread worm as fast as possible
    - Caused disruption and helped detection
- Today's botnets are controlled by crackers looking for profit, which rent them out
    - Criminal organizations
- Spread more slowly, infected machine might lie dormant for weeks

Sample botnet: Storm

- In September 2007, Storm Worm botnet included hundreds of thousands or even millions of machines
- Bots were used to send out junk emails advertising web links that when clicked attempted to download and install worm, or to host these websites
- Botnet was also rented out for pharmacy and investment spam
- As a self-defence mechanism, it ran DDoS attacks against Internet addresses that scanned for it
- Authors were thought to reside in St. Petersburg, Russia

Active code

- To reduce load on server, server might ask client to execute code on its behalf
  - Java, JavaScript, ActiveX, Flash, Silverlight
  - Invoke another application (Word, iTunes,...)
  - Maybe inadvertently (see XSS attack)
- Obviously, this can be dangerous for client
- Java 1.1 ran in a sandbox with limited capabilities, code is checked for correctness
  - No writing to a file, no talking to random network nodes
  - Similar for JavaScript
  - But it could still use up CPU or memory resources, wreak havoc with display, or play annoying music

Active code (cont.)

- Java 1.2+ can break out of sandbox if approved by user
  - What's the problem here?
- Worse: Java 7 runs signed applets out of sandbox by default
- ActiveX
  - No sandbox or correctness check
  - Downloaded code is cryptographically signed, signature is verified to be from "trusted" entity before execution
- Third-party applications
  - Turn out to be a huge problem, for all browsers
  - Malicious input parameters, Word macros,. . .
  - Potentially disastrous if application has full access rights to a user's account

Active code (cont.)

- **Privileged:** The application will run with unrestricted access which may put your computer and personal information at risk.
- **Sandboxed:** The application will run with restricted access that is intended to protect your computer and personal information.

Script kiddies

- For all of the discussed attacks, exploit code and complete attack scripts are available on the Internet
- Script kiddies can download scripts and raise an attack with minimum effort
- There are even tools that allow easy building of individual attacks based on existing exploits
  - E.g., Metasploit Framework

Module outline

1. Network concepts
2. Threats in networks
3. Network security controls
4. Firewalls
5. Honeypots / honeynets
6. Intrusion detection systems

Design and implementation

- Use controls against security flaws in programs that we talked about earlier
- Always check inputs, don't ever trust input from a client
    - Use a white list of allowed characters, not a black list of forbidden ones

Segmentation and separation

- Don't put all a company's servers on a single machine
- Deploy them on multiple machines, depending on their functional and access requirements
- If a machine gets broken into, only some services will be affected
- E.g., the web server of a company needs to be accessible from the outside and is therefore more vulnerable
- Therefore, it shouldn't be trusted by other servers of the company, and it should be deployed outside the company firewall (see DMZ later)

Redundancy

- Avoid single points of failure
  - Even if you don't have to worry about attackers
  - Disk crash, power failure, earth quake,...
- (Important) servers should be deployed in a redundant way on multiple machines, ideally with different software to get genetic diversity and at different locations
- Redundant servers should be kept in (close) sync so that backup servers can take over easily
  - Test this!
  - Keep backup copies at a safe place in case you get hit by Murphy's law

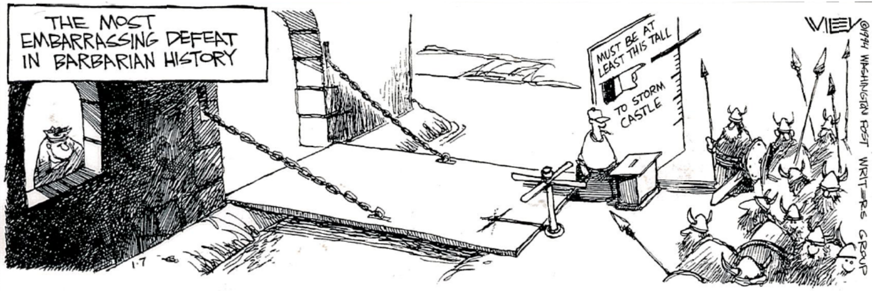
Access controls

- ACLs on routers
    - All traffic to a company typically goes through a single (or a few) routers
    - In case of flooding attack, define router ACL that drops packets with particular source and destination address
    - ACLs are expensive for high-traffic routers
    - Difficult to gather logs for forensics analysis
    - Source addresses of packets in flood are typically spoofed and dynamic
- Firewalls
    - Firewalls have been designed to filter traffic, maybe based on other criteria than just packet addresses

Module outline

1. Network concepts
2. Threats in networks
3. Network security controls
4. Firewalls
5. Honeypots / honeynets
6. Intrusion detection systems

Firewalls


THE MOST EMBARRASSING DEFEAT IN BARBARIAN HISTORY
MUST BE AT LEAST THIS TALL TO STORM CASTLE
WILEY ©1994 WASHINGTON POST WRITERS GROUP
1·7

Firewalls

- Firewalls are the castles of the Internet age
- All traffic into/out of a company has to go through a small number of gates (choke points)
  - Wireless access point should be outside of firewall
- Choke points carefully examine traffic, especially incoming, and might refuse it access
  - Two strategies: "permit everything unless explicitly forbidden" or "forbid everything unless explicitly allowed"
- Company firewalls do not protect against attacks on company hosts that originate within the company
  - Need multiple layers of defense / defense in depth

Types of firewalls

- Packet filtering gateways / screening routers
- Stateful inspection firewalls
- Application proxies
- Personal firewalls
- Firewalls are attractive targets for attackers, they (except personal ones) are typically deployed on designated computers that have been stripped of all unnecessary functionality to limit attack surface

Packet filtering gateways

- Simplest type
- Make decision based on header of a packet
  - Header contains source and destination addresses and port numbers, port numbers can be used to infer type of packet
    - 80 -> Web, 22 -> SSH
    - E.g., allow Web, but not SSH
- Ignore payload of packet
- Can drop spoofed traffic
  - uWaterloo's firewall could drop all packets originating from uWaterloo whose source address is not of the form 129.97.x.y
  - And traffic originating from outside of uWaterloo whose source address is of the form 129.97.x.y
  - Does this eliminate spoofed traffic completely?

Stateful inspection firewalls

- More expensive than packet filtering
- Keep state to identify packets that belong together
  - When a client within the company opens a TCP connection to a server outside the company, firewall must recognize response packets from server and let (only) them through
  - Some application-layer protocols (e.g., FTP) require additional (expensive) inspection of packet content to figure out what kind of traffic should be let through
- IP layer can fragment packets, so firewall might have to re-assemble packets for stateful inspection

Application proxy

- Client talks to proxy, proxy talks to server
  - Specific for an application (email, Web,...)
  - Not as transparent as packet filtering or stateful inspection
  - Intercepting proxy requires no explicit configuration by client (or knowledge of this filtering by client)
  - All other traffic is blocked
- For users within the company wanting to access a server outside the company (forward proxy) and vice versa (reverse proxy)
- Proxy has full knowledge about communication and can do sophisticated processing
  - Limit types of allowed database queries, filter URLs, log all emails, scan for viruses
- Can also do strong user authentication

Personal firewalls

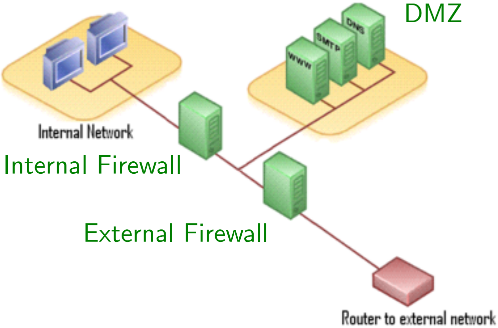
- Firewall that runs on a (home) user's computer
  - Especially important for computers that are always online
- Typically "forbid everything unless explicitly allowed"
  - Definitely for communication originating from other computers
  - Maybe also for communication originating on the user's computer
    - Why? What's the problem here?

Personal firewalls (cont.)

- Protect against attacks on servers running on computer
  - Servers that are running unnecessarily (e.g., Windows XP before SP 1 suffered from this)
  - Servers that are wrongly configured and that allow access from other computers (or that cannot be configured to disallow this)
  - Servers that have a remotely exploitable bug

Demilitarized Zone (DMZ)

- Subnetwork that contains an organization's external services, accessible to the Internet
- Deploy external and internal firewall
  - External firewall protects DMZ
  - Internal firewall protects internal network from attacks lodged in DMZ



Source: Wikipedia

Module outline

1. Network concepts
2. Threats in networks
3. Network security controls
4. Firewalls
5. Honeypots / honeynets
6. Intrusion detection systems

Honeypots / honeynets

- Set up an (unprotected) computer or an entire network as a trap for an attacker
- System has no production value, so any activity is suspicious
  - Any received email is considered spam
- Observe attacker to learn about new attacks, to identify and stop attacker, or to divert attacker from attacking real system
- Obviously, attacker should not be able to learn that attacked system is a honeypot/-net
  - Cat-and-mouse game
- Also, attacker might be able to use honeypot/-net to break into real system

Types of honeypots/-nets

- Low interaction
  - Daemon that emulates one or multiple hosts, running different services
  - Easy to install and maintain
  - Limited amount of information gathering possible
  - Easier for the attacker to detect than high interaction honeynets
- High interaction
  - Deploy real hardware and software, use stealth network switches or keyloggers for logging data
  - More complex to deploy
  - Can capture lots of information
  - Can capture unexpected behaviour by attacker

Module outline

1. Network concepts
2. Threats in networks
3. Network security controls
4. Firewalls
5. Honeypots / honeynets
6. Intrusion detection systems

Intrusion detection systems (IDSs)

- Firewalls do not protect against inside attackers or insiders making mistakes and can be subverted
- IDSs are next line of defense
- Monitor activity to identify malicious or suspicious events
  - Receive events from sensors
  - Store and analyze them
  - Take action if necessary
- Host-based and network-based IDSs
- Signature-based and heuristic/anomaly-based IDSs

Host-based and network-based IDSs

- Host-based IDSs
  - Run on a host to protect this host
  - Can exploit lots of information (packets, disk, memory,...)
  - Miss out on information available to other (attacked) hosts
  - If host gets subverted, IDS likely gets subverted, too
- Network-based IDSs
  - Run on dedicated node to protect all hosts attached to a network
  - Have to rely on information available in monitored packets
  - Typically more difficult to subvert
- Distributed IDSs combine the two of them

Signature-based IDSs

- Each (known) attack has its signature
  - E.g., many SYNs to ports that are not open could be part of a port scan
- Signature-based IDSs try to detect attack signatures
- Fail for new attacks or if attacker manages to modify attack such that its signature changes
  - Polymorphic worms
- Might exploit statistical analysis

Heuristic/anomaly-based IDSs

- Look for behaviour that is out of the ordinary
- By modelling good behaviour and raising alert when system activity no longer resembles this model
- Or by modelling bad behaviour and raising alert when system activity resembles this model
- All activity is classified as good/benign, suspicious, or unknown
- Over time, IDS learns to classify unknown events as good or suspicious
    - Maybe with machine learning

Example: Tripwire

- Anomaly-based, host-based IDS, detects file modifications
- Initially, compute digital fingerprint of each system file and store fingerprints at a safe place
- Periodically, re-compute fingerprints and compare them to stored ones
- (Malicious) file modifications will result in mismatches
- Why is it not a good idea to perform the second step directly on the production system?

IDS discussion

- Stealth mode
  - Two network interfaces, one for monitoring traffic, another one for administration and for raising alarms
  - First one has no published address, so it does not exist for routing purposes (passive wiretap)
- Responding to alarms
  - Type of response depends on impact of attack
  - From writing a log entry to calling a human
- False positives/negatives
  - Former might lead to real alarms being ignored
  - IDS might be tunable to strike balance between the two
  - In general, an IDS needs to be monitored to be useful

Recap

- Network concepts
- Threats in networks
- Network security controls
- Firewalls
- Honeypots / honeynets
- Intrusion detection systems