CS459/698 Privacy, Cryptography, Network and Data Security

Network Security Primer

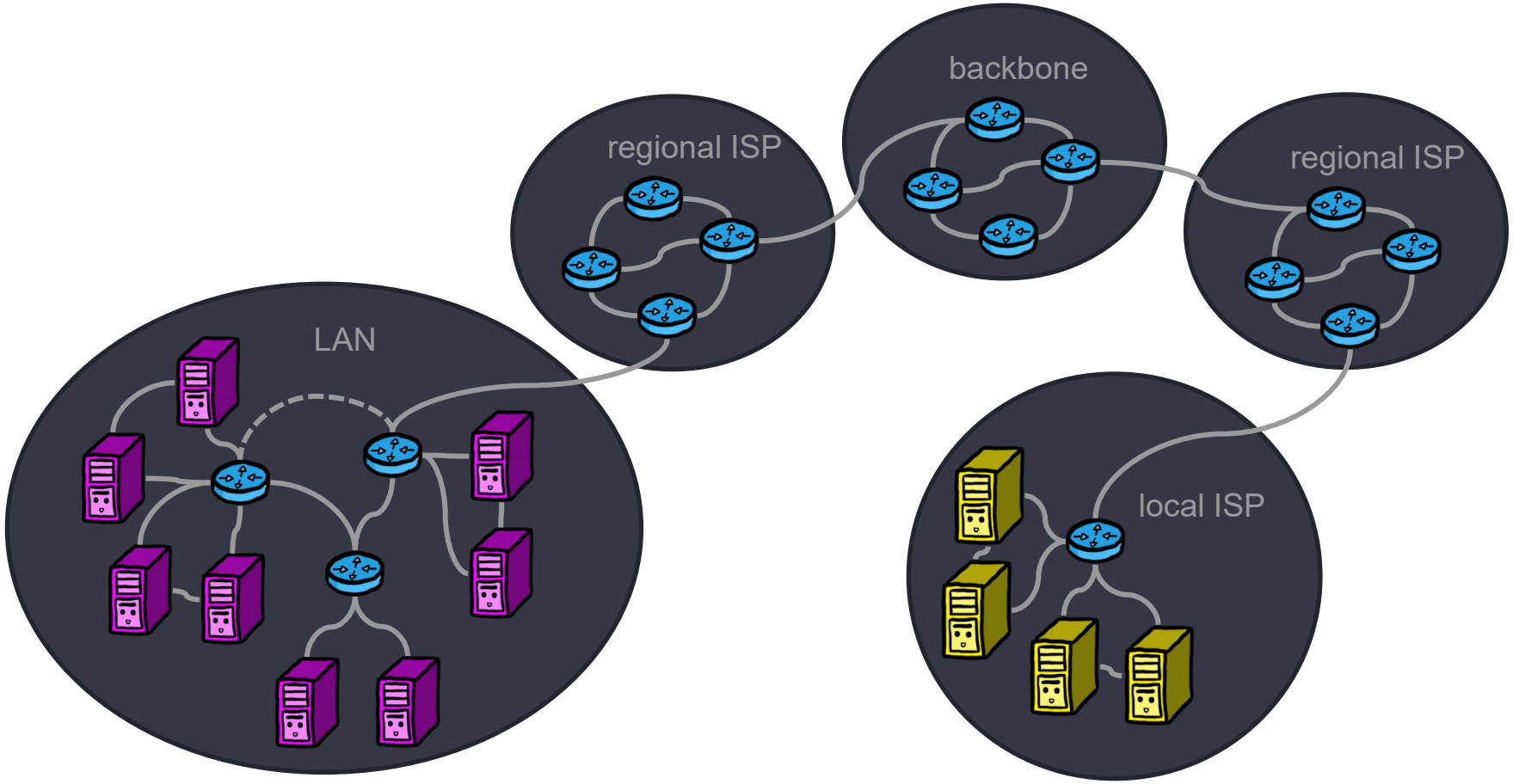
Fall 2025, Tuesday/Thursday 8:30-9:50am

Today's Class

- Networking background
  - Internet's architecture
  - UDP/TCP
  - IP

- How to protect the network at the system level
  - Barriers to protect entry
    - Firewalls
    - DMZ
  - Detection systems to support these barriers
    - Honeypots
    - Intrusion Detection

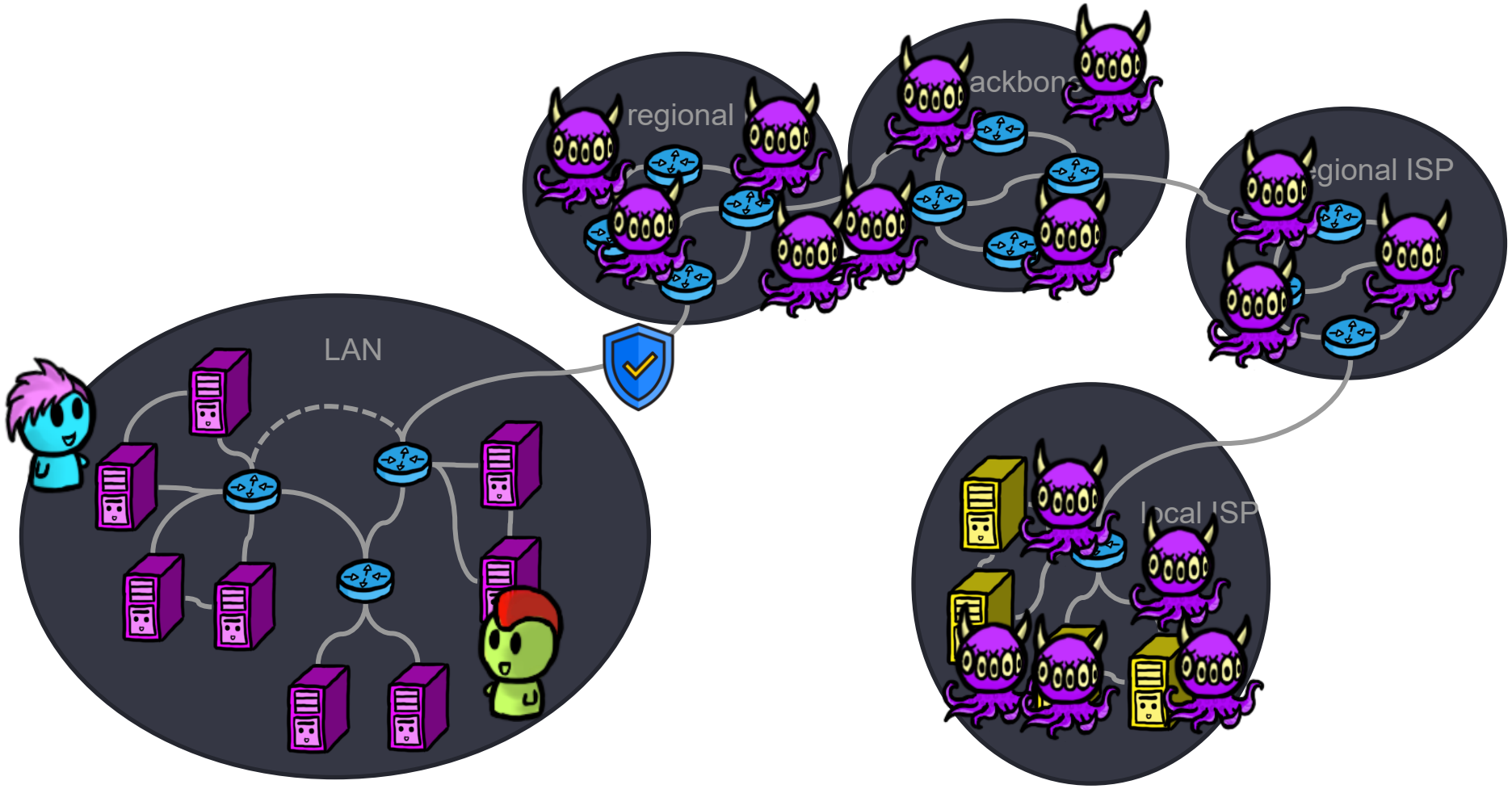
Architecture of the Internet



Characteristics of the Internet (that make security hard)

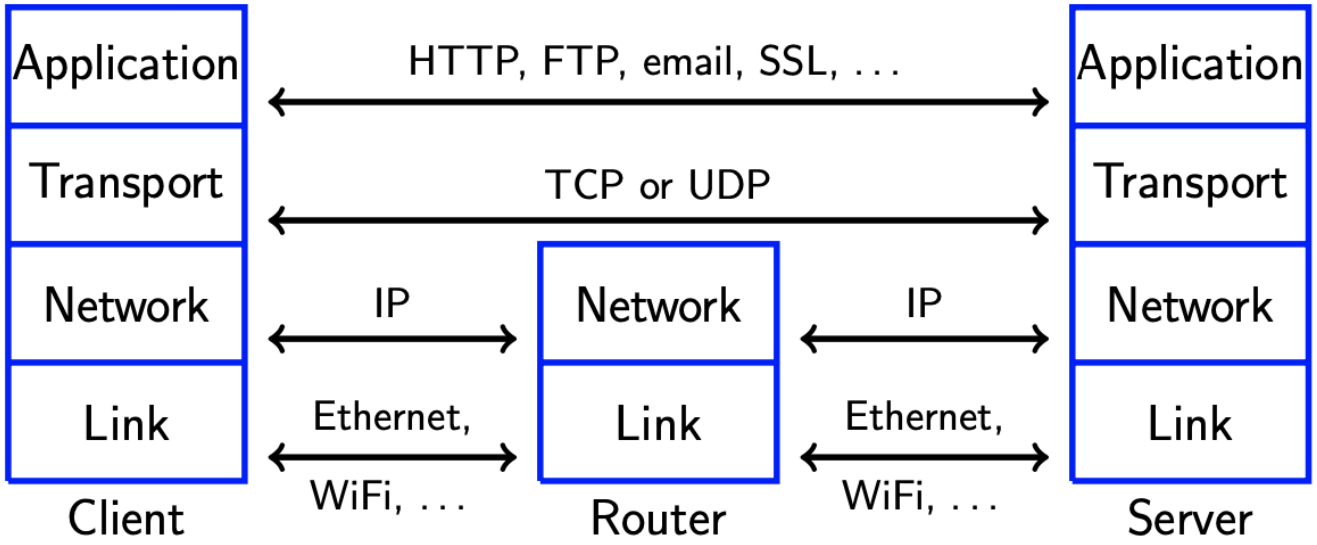
- Traffic from a source to a destination flows through nodes controlled by different entities
- End nodes cannot control through which nodes traffic flows
  - Worse, all traffic is split up into individual packets, and each packet could be routed along a different path
- Different types of nodes
  - 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,. . .

How can we be safe?



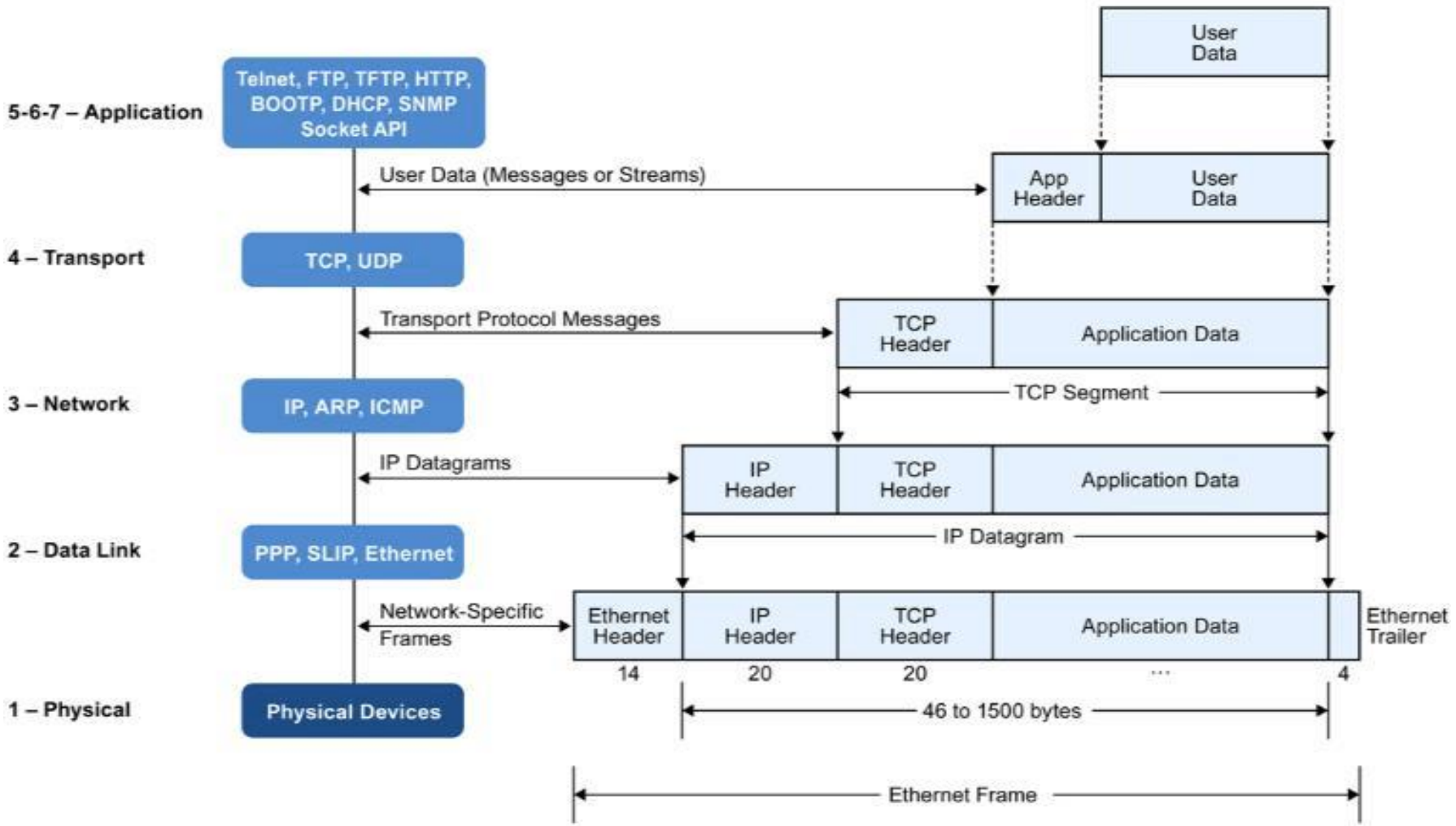
Networking Basics

TCP/IP Model



- 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

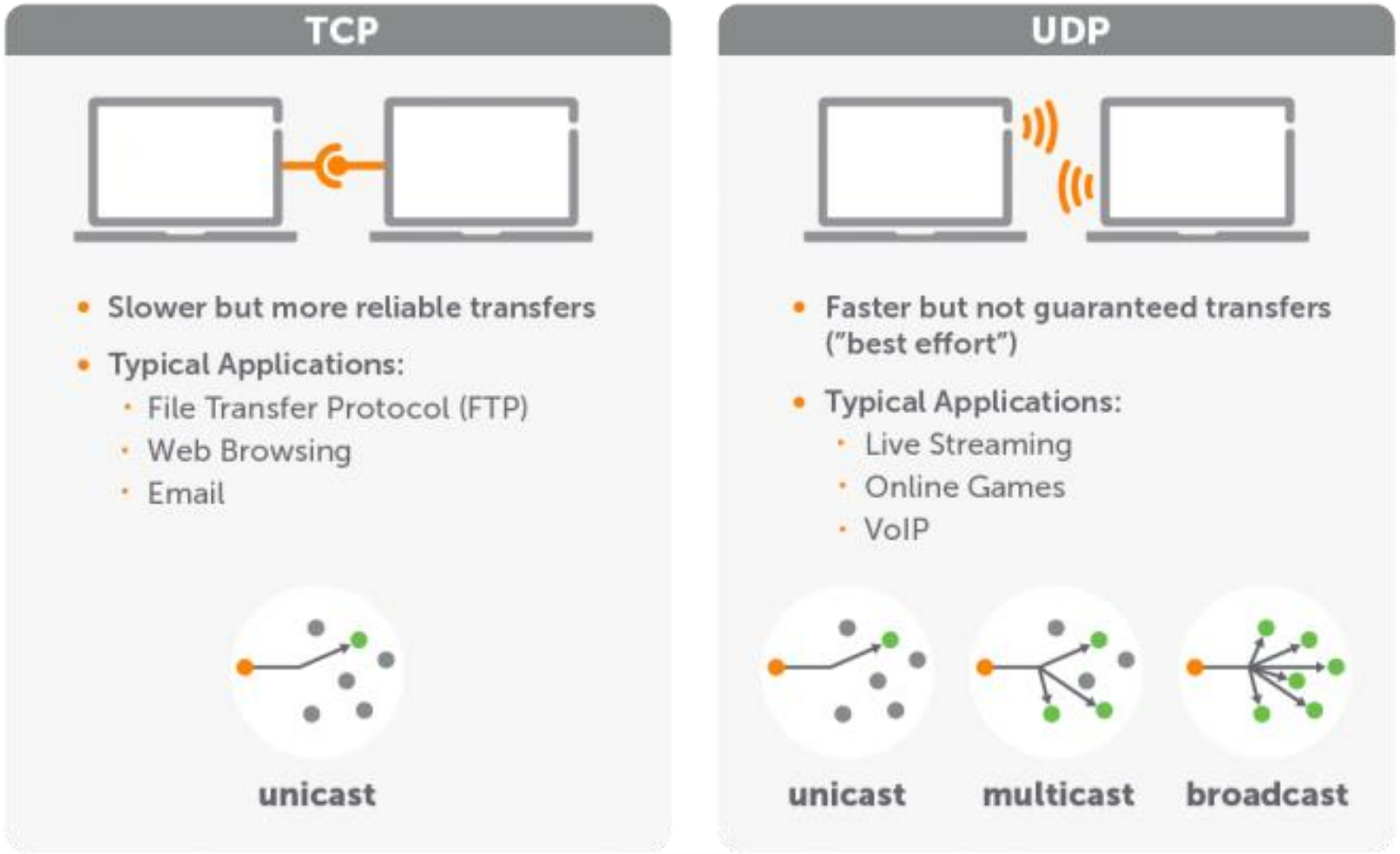
Packet Formats



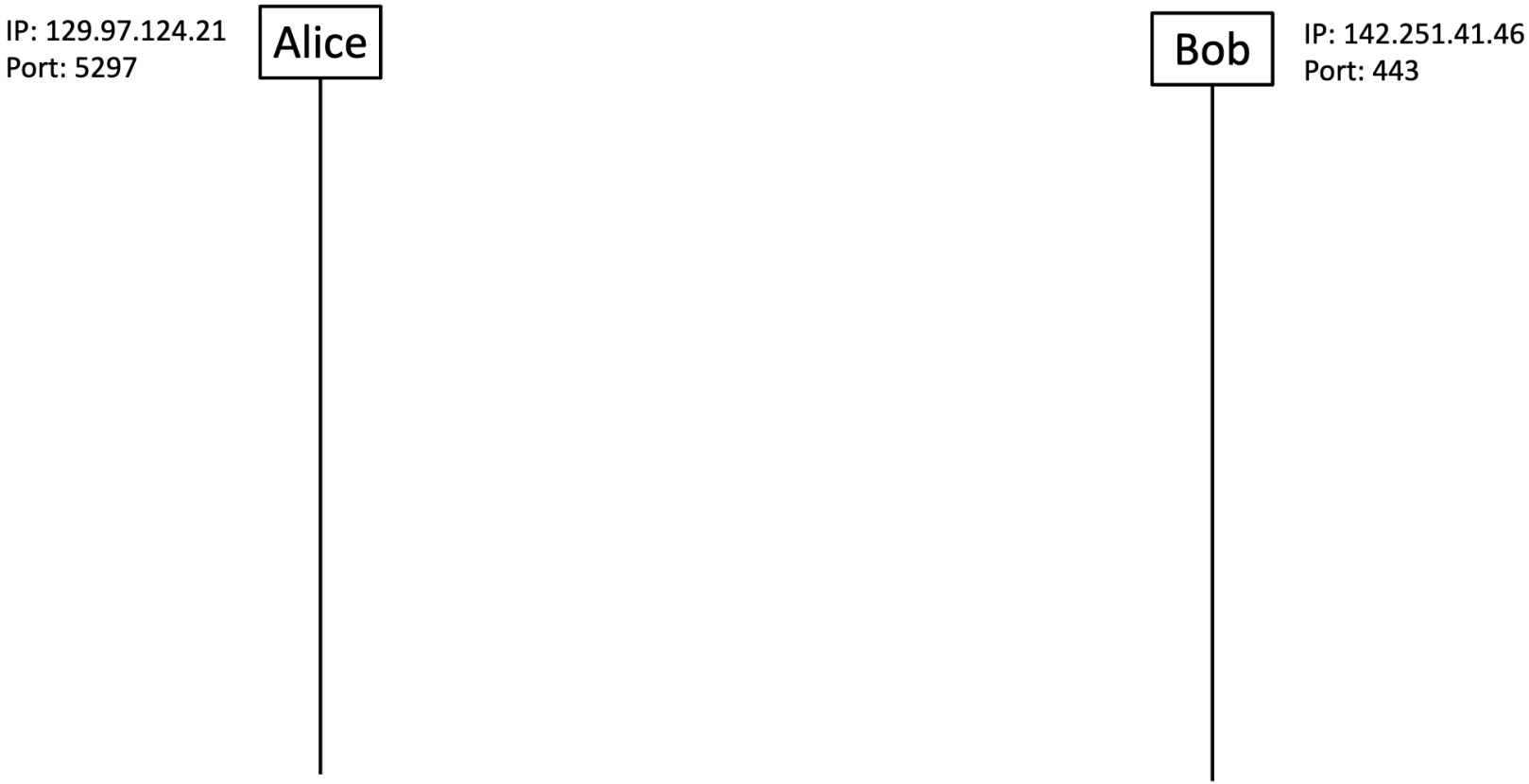
IP


Sender of packet
Recipient of packet
192.16.00.12
192.16.00.12
To: 192.00.00.75
The Internet
192.00.00.75

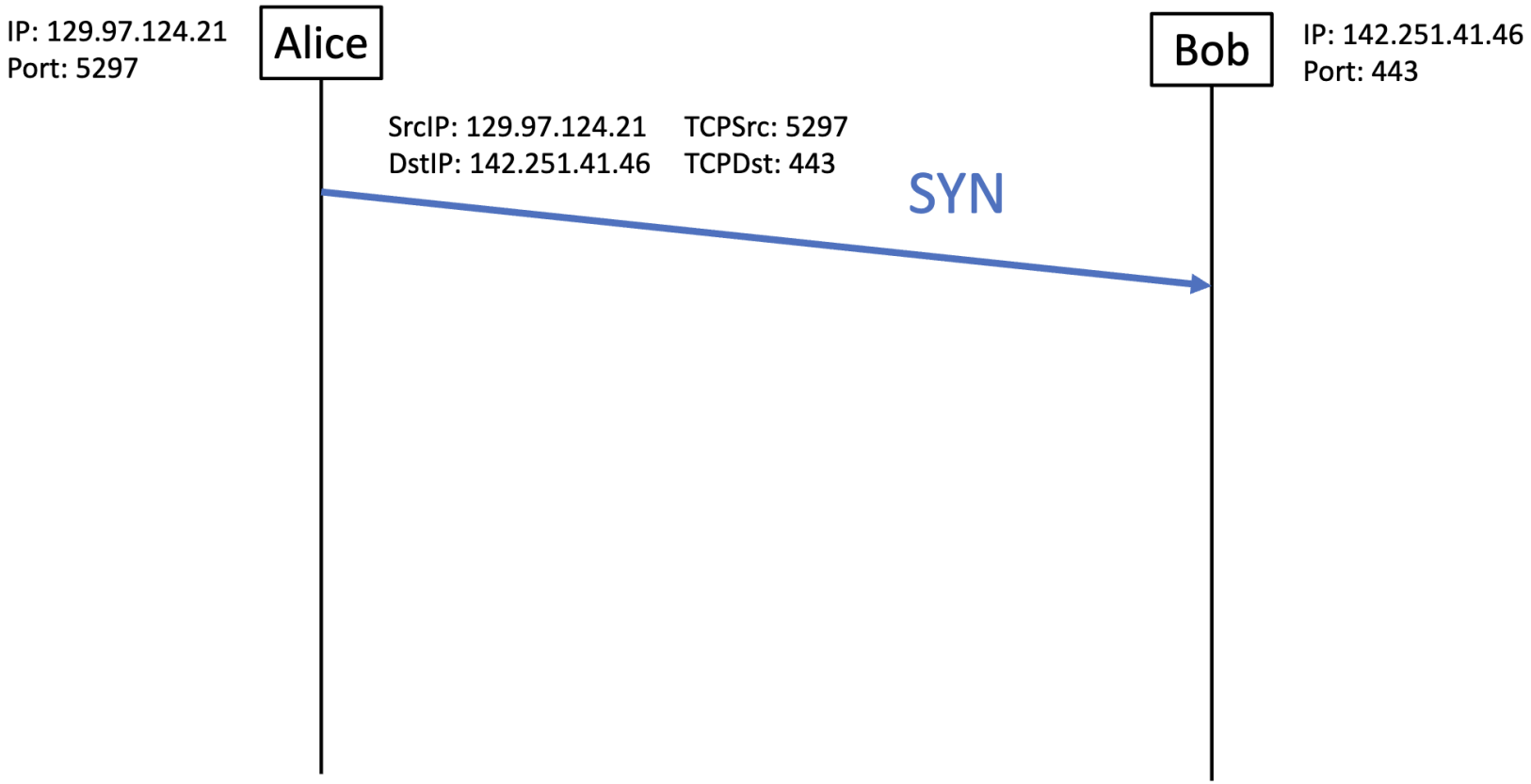
UDP vs TCP


TCP
Slower but more reliable transfers
Typical Applications:
File Transfer Protocol (FTP)
Web Browsing
Email
unicast
UDP
Faster but not guaranteed transfers ("best effort")
Typical Applications:
Live Streaming
Online Games
VoIP
unicast
multicast
broadcast

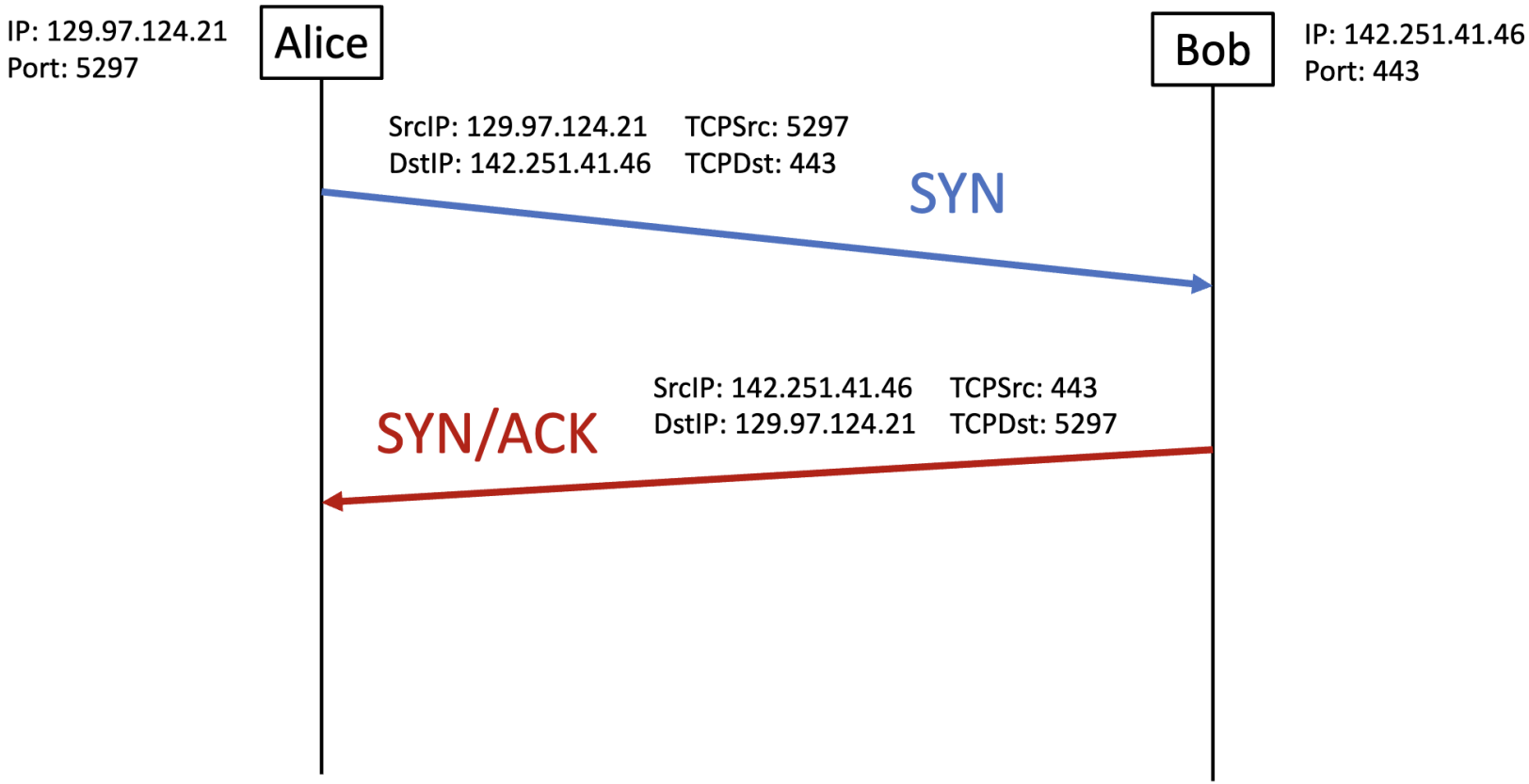
The TCP Handshake



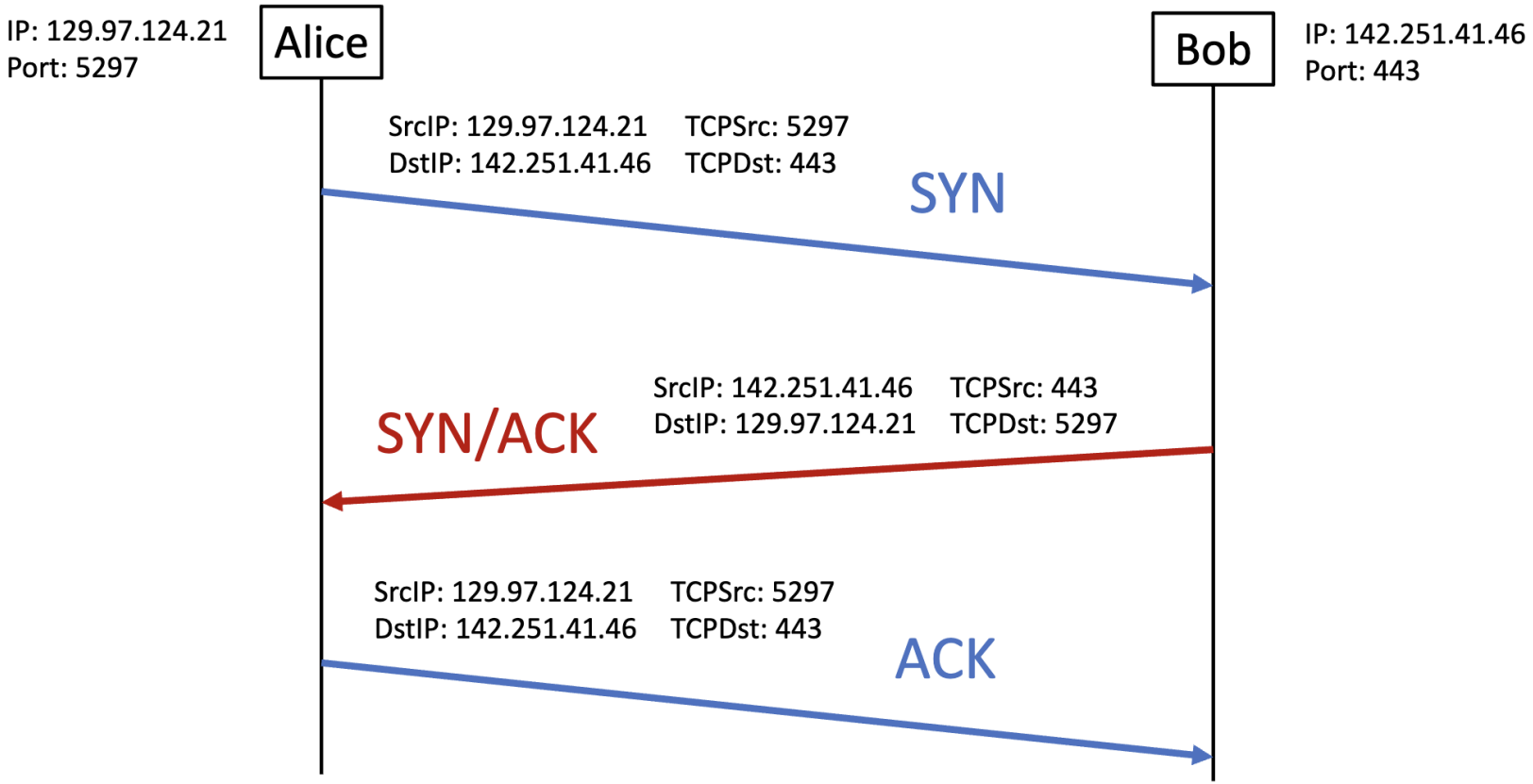
The TCP Handshake



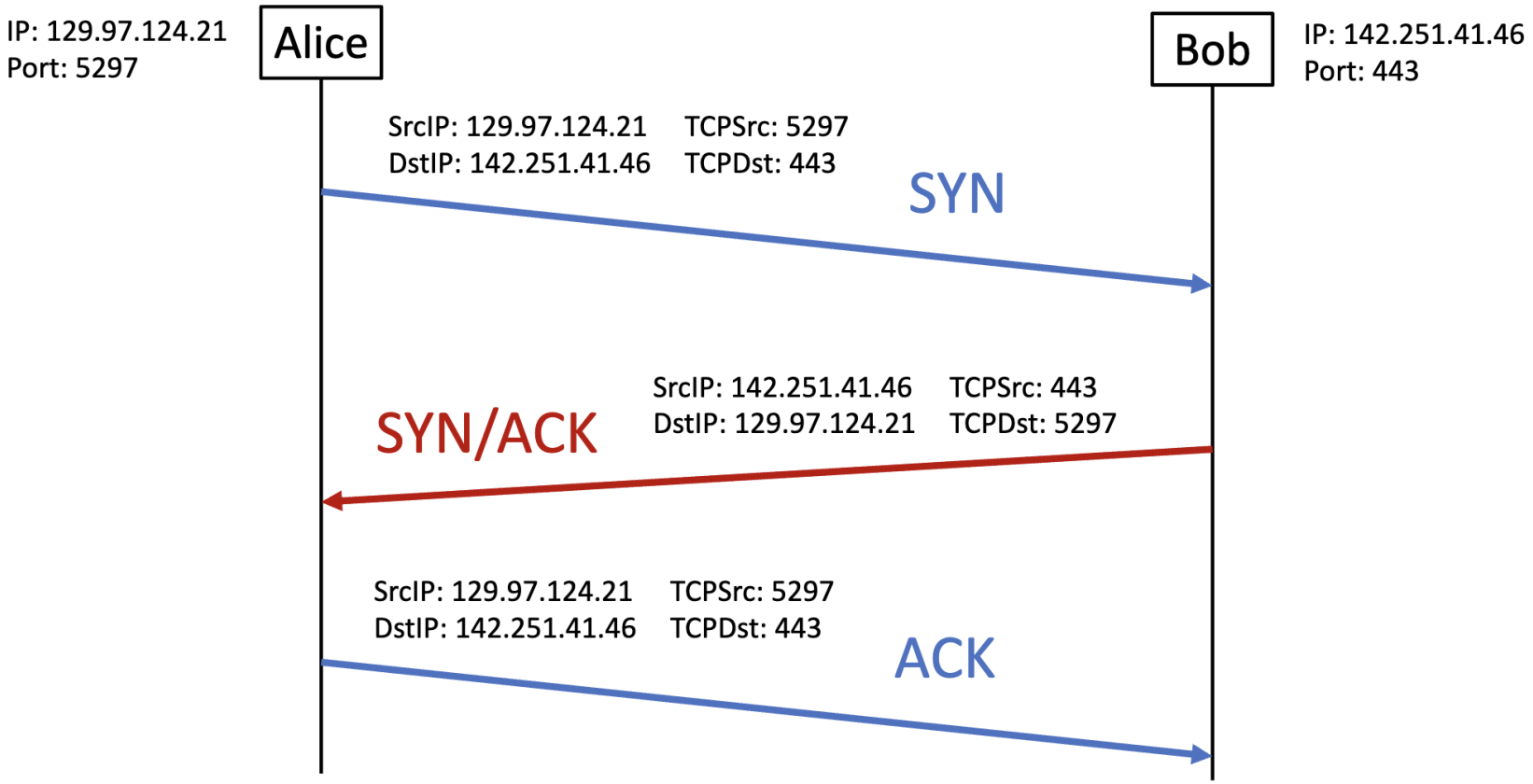
The TCP Handshake



The TCP Handshake

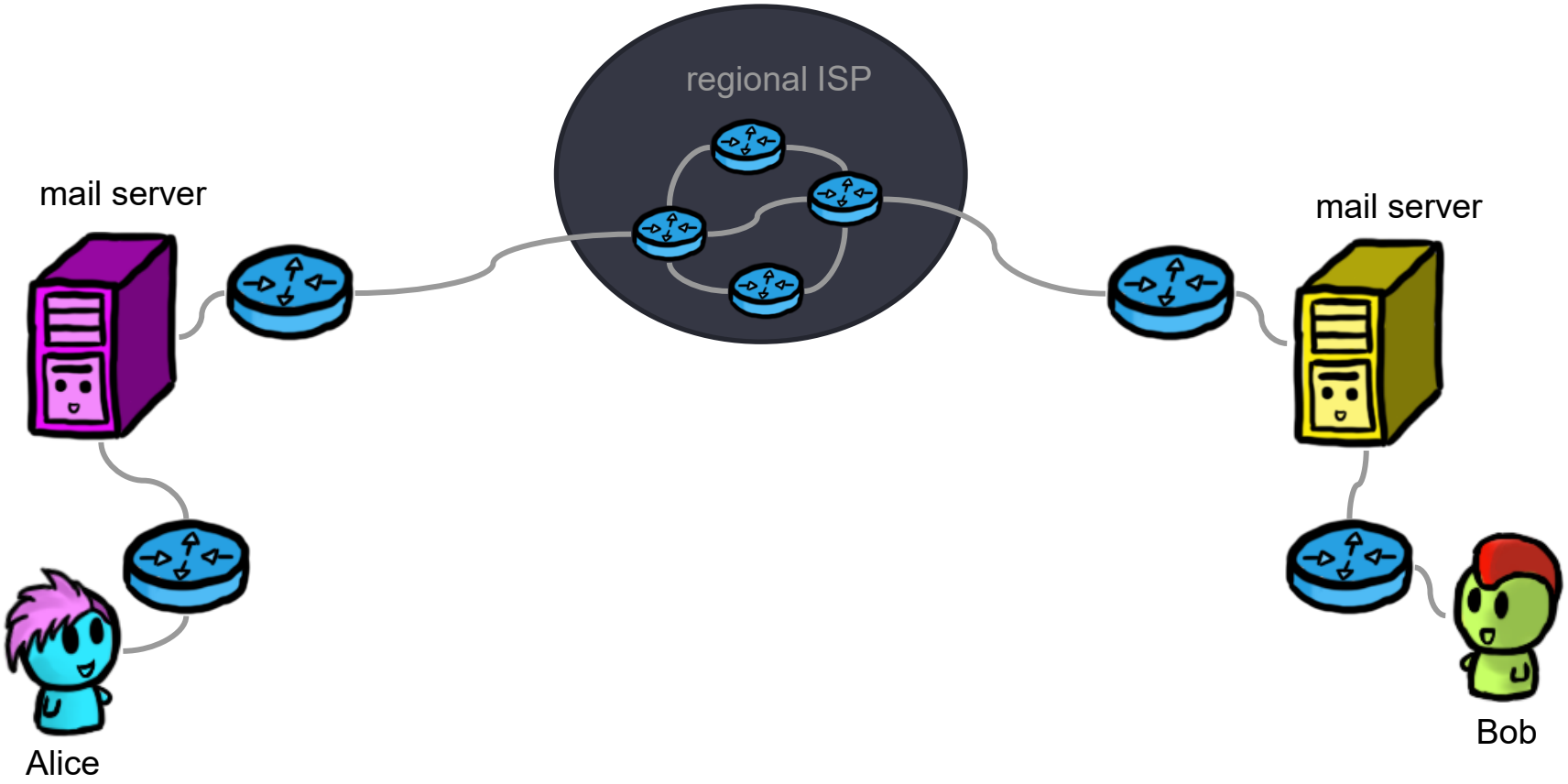


The TCP Handshake

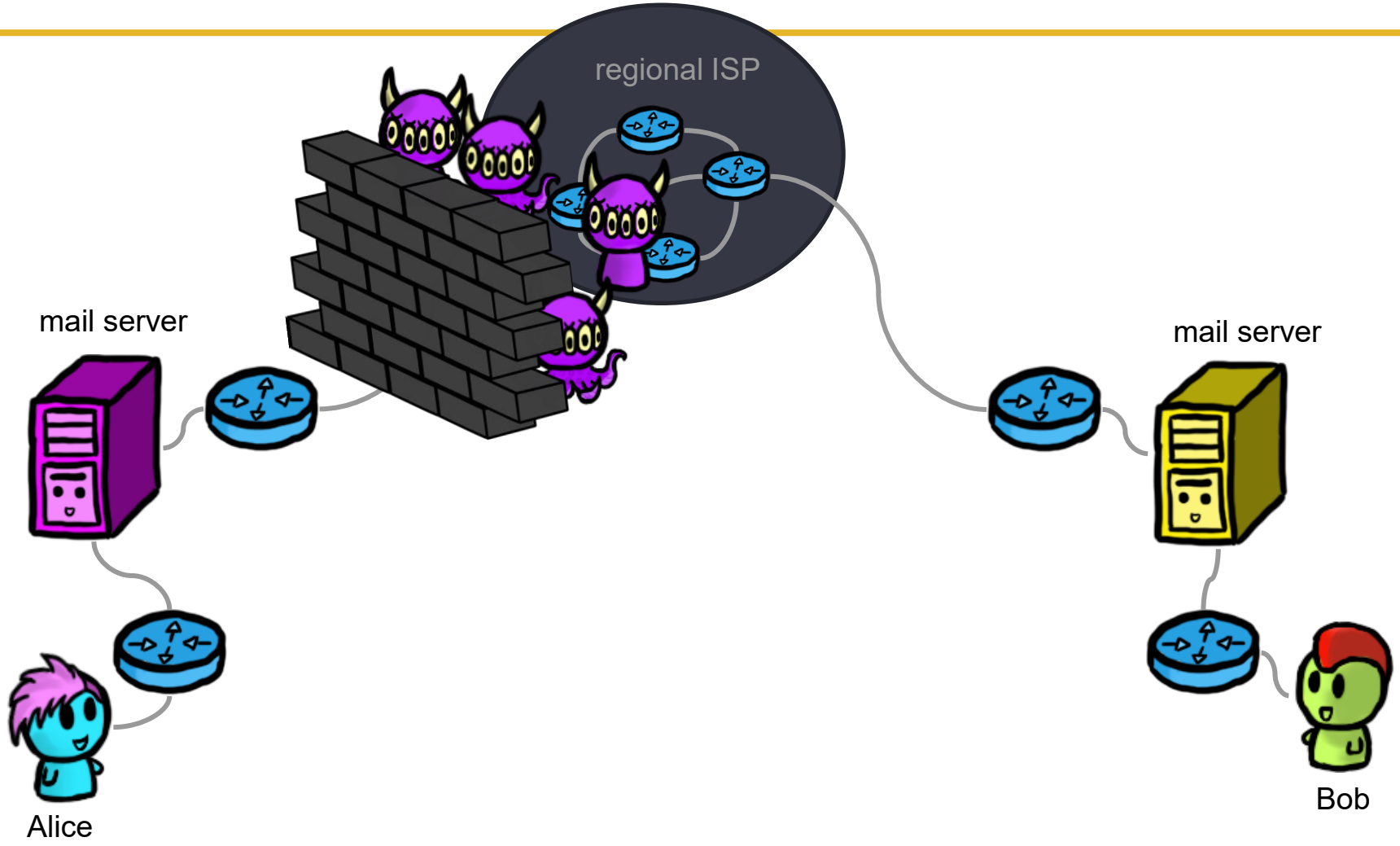


Disclaimer: omitting several other details (SEQ numbers, MSS, window size, additional flags, etc.)

Working Example



Keep Unauthorized Individuals Out of the LAN



Redundancy (in case we fail)

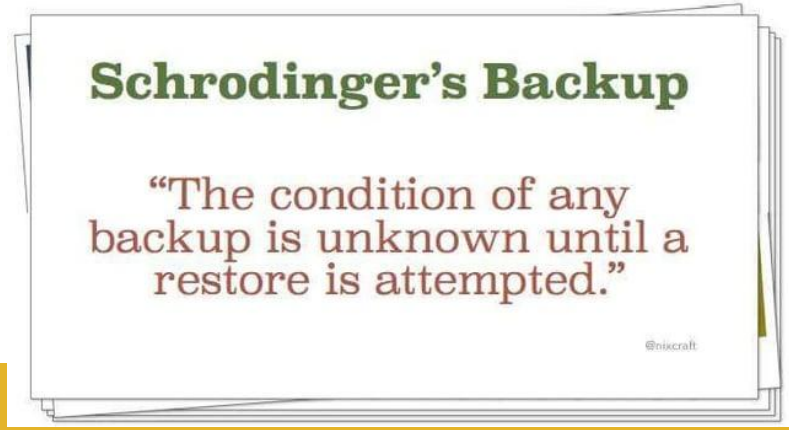
- Avoid **single points of failure**
  - Even if you don’t have to worry about attackers
  - Disk crash, power failure, earthquake…
- Servers should be deployed in a **redundant** way on multiple machines, ideally with different software to get **genetic diversity** and at **different locations**
- Redundant servers should be kept in (close) **sync** so that backup servers can take over easily

Redundancy (in case we fail)

- Avoid **single points of failure**
  - Even if you don't have to worry about attackers
  - Disk crash, power failure, earthquake…
- 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

Redundancy (in case we fail)

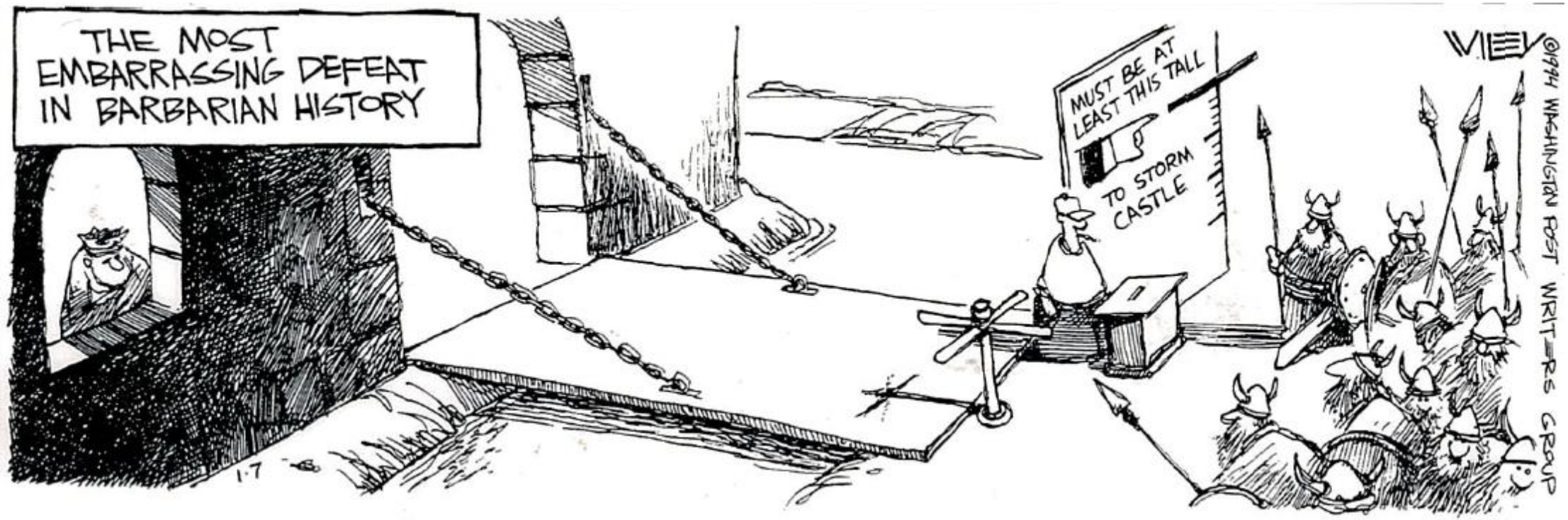
- Avoid **single points of failure**
  - Even if you don't have to worry about attackers
  - Disk crash, power failure, earthquake…
- Servers should be deployed in a **redundant** way on multiple machines, ideally with different software to get **genetic diversity** and at **different locations**
- Redundant servers should be kept in (close) **sync** so that backup servers can take over easily
  - Test this! Beware of **Schrödinger backups**



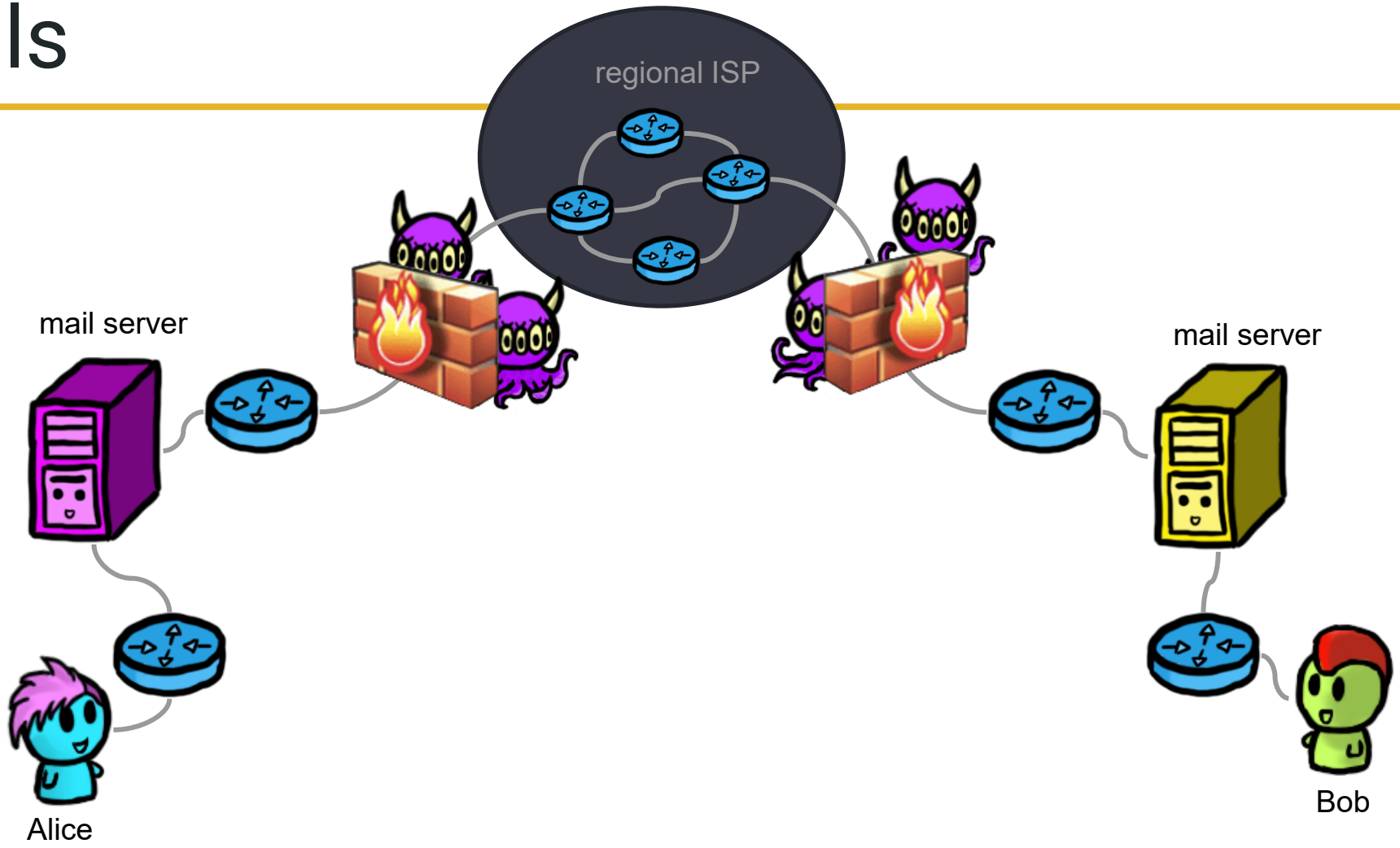
Access controls

- Access Control Lists (ACLs) on routers
  - All traffic to a company typically goes through a single (or a few) routers
  - Filter traffic based on allow/block list
  - In case of flooding attack, define router ACL that drops packets with particular source and destination address
  - ACLs are expensive for high-traffic routers
  - Source addresses of packets in a flood are typically spoofed and dynamic
- We need something more sophisticated.

Firewalls


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

Firewalls



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:
    - Blocklist: "permit everything unless explicitly forbidden"
    - Allowlist: "forbid everything unless explicitly allowed"

Blocklist vs allowlist

Limitations of Firewalls

- Firewalls do not protect against internal attacks
  - Employee who colludes with external party
  - Laptop that is infected while outside the LAN
  - Poorly secured WIFI allowing an attacker inside the LAN.
- Vulnerable to IP Spoofing (more later)
- 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 are typically deployed on designated computers that have been stripped of all unnecessary functionality to limit attack surface

Packet filtering gateways

- Simplest type – very fast and transparent
- Make decision based on network 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

Example

Table 12.1 Packet-Filtering Example

Rule	Direction	Source Address	Destination Address	Protocol	Destination Port	Action
A	In	External	Internal	TCP	25	Permit
B	Out	Internal	External	TCP	> 1023	Permit
C	Out	Internal	External	TCP	25	Permit
D	In	External	Internal	TCP	> 1023	Permit
E	Either	Any	Any	Any	Any	Deny

A. Allow inbound mail from an external source (port 25 is for SMTP incoming)
B. Allow a response to an inbound SMTP connection
C. Allow outbound mail to an external source
D. Allow a response to an outbound SMTP connection
E. Explicit statement of default policy (deny all)

Example

Table 12.1 Packet-Filtering Example

Rule	Direction	Source Address	Destination Address	Protocol	Destination Port	Action
A	In	External	Internal	TCP	25	Permit
B	Out	Internal	External	TCP	> 1023	Permit
C	Out	Internal	External	TCP	25	Permit
D	In	External	Internal	TCP	> 1023	Permit
E	Either	Any	Any	Any	Any	Deny

A. Allow inbound mail from an external source (port 25 is for SMTP incoming)
B. Allow a response to an inbound SMTP connection
C. Allow outbound mail to an external source
D. Allow a response to an outbound SMTP connection
E. Explicit statement of default policy (deny all)

Q: Problems?

Example

Table 12.1 Packet-Filtering Example

Rule	Direction	Source Address	Destination Address	Protocol	Destination Port	Action
A	In	External	Internal	TCP	25	Permit
B	Out	Internal	External	TCP	> 1023	Permit
C	Out	Internal	External	TCP	25	Permit
D	In	External	Internal	TCP	> 1023	Permit
E	Either	Any	Any	Any	Any	Deny

A. Allow inbound mail from an external source (port 25 is for SMTP incoming)
B. Allow a response to an inbound SMTP connection
C. Allow outbound mail to an external source
D. Allow a response to an outbound SMTP connection
E. Explicit statement of default policy (deny all)

Rule D is too permissive!

Example

Table 12.1 Packet-Filtering Example

Rule	Direction	Source Address	Destination Address	Protocol	Destination Port	Action
A	In	External	Internal	TCP	25	Permit
B	Out	Internal	External	TCP	> 1023	Permit
C	Out	Internal	External	TCP	25	Permit
D	In	External	Internal	TCP	> 1023	Permit
E	Either	Any	Any	Any	Any	Deny

Let's work on a fix… is this good enough?

Rule	Direction	Source Address	Source Port	Dest Address	Protocol	Dest Port	Action
D	In	External	25	Internal	TCP	> 1023	Permit

Example

Table 12.1 Packet-Filtering Example

Rule	Direction	Source Address	Destination Address	Protocol	Destination Port	Action
A	In	External	Internal	TCP	25	Permit
B	Out	Internal	External	TCP	> 1023	Permit
C	Out	Internal	External	TCP	25	Permit
D	In	External	Internal	TCP	> 1023	Permit
E	Either	Any	Any	Any	Any	Deny

Let's work on a fix… is this good enough?

Rule	Direction	Source Address	Source Port	Dest Address	Protocol	Dest Port	Action
D	In	External	25	Internal	TCP	> 1023	Permit

Example

Table 12.1 Packet-Filtering Example

Rule	Direction	Source Address	Destination Address	Protocol	Destination Port	Action
A	In	External	Internal	TCP	25	Permit
B	Out	Internal	External	TCP	> 1023	Permit
C	Out	Internal	External	TCP	25	Permit
D	In	External	Internal	TCP	> 1023	Permit
E	Either	Any	Any	Any	Any	Deny

Ah!

Rule	Direction	Source Address	Source Port	Dest Address	Protocol	Dest Port	Flag	Action
D	In	External	25	Internal	TCP	> 1023	ACK	Permit

What about spoofing?

- A firewall 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
    - from outside of uWaterloo whose source address is of the form 129.97.x.y

What about spoofing?

- A firewall 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
    - from outside of uWaterloo whose source address is of the form 129.97.x.y

Q: Does this eliminate spoofed traffic completely?

What about spoofing?

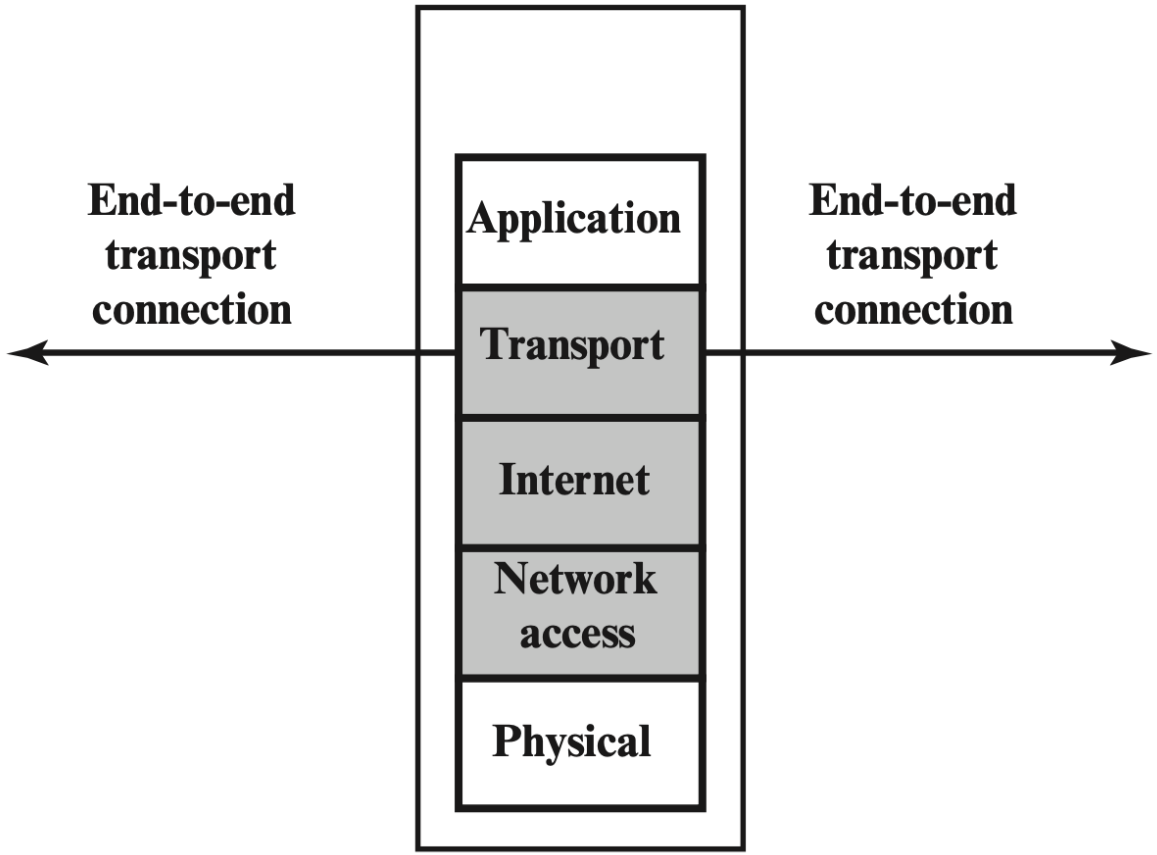
- A firewall can drop spoofed traffic
  - UWaterloo's firewall could drop all packets originating:
    - from uWaterloo whose source address is not of the form 129.97.x.y
    - from outside of uWaterloo whose source address is of the form 129.97.x.y

Q: Does this eliminate spoofed traffic completely?

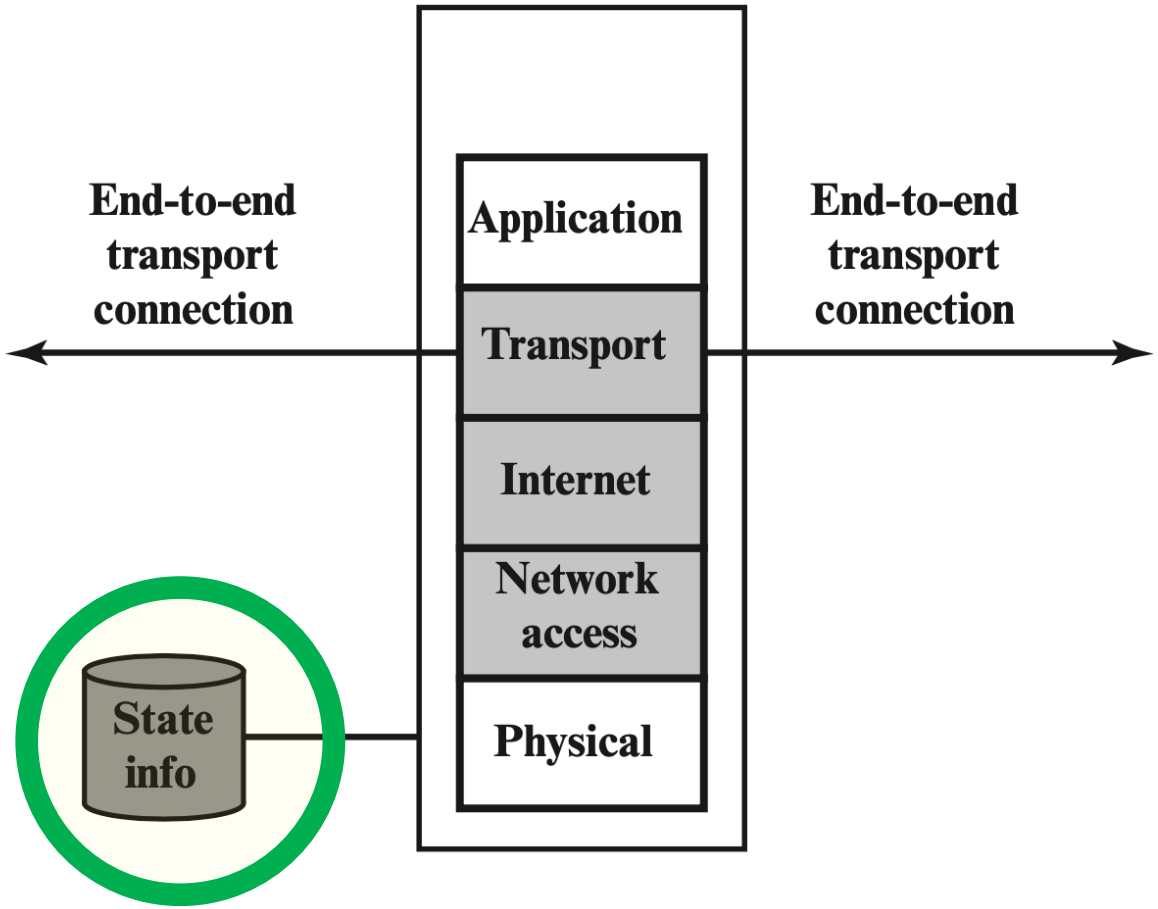
A: No. Internal spoofing may still occur.

i.e., we cannot drop traffic that does not cross the firewall at all…

From Stateless to Stateful



(b) Packet filtering firewall



(c) Stateful inspection firewall

Stateful inspection firewalls

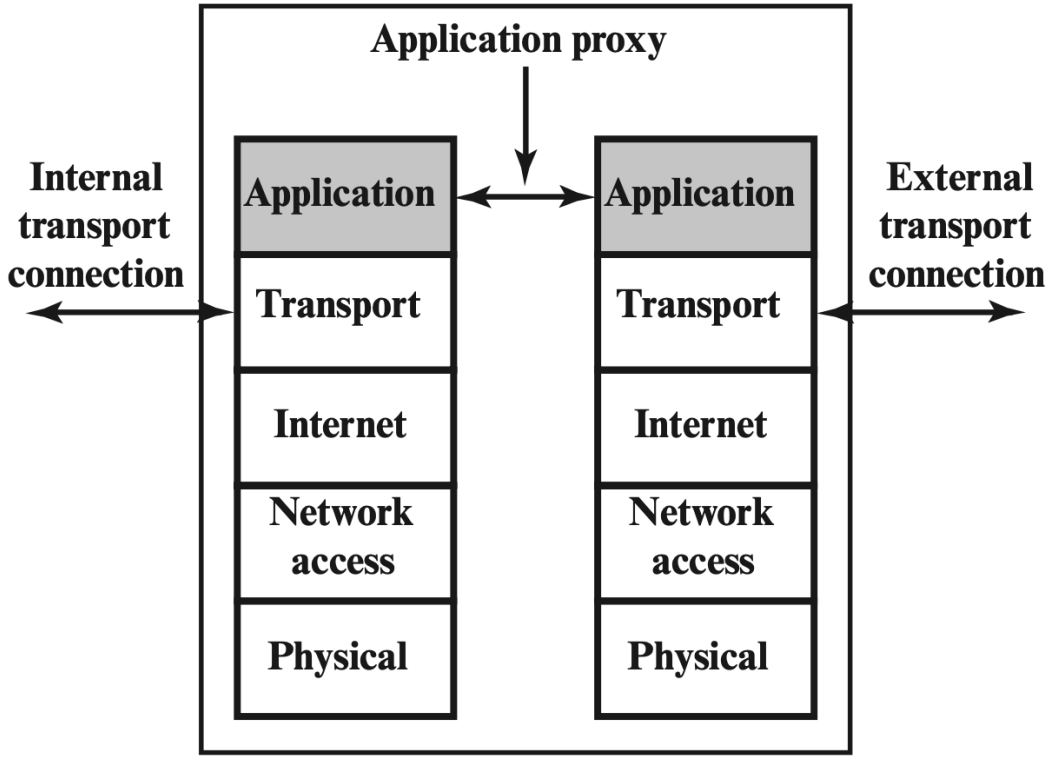
- More expensive than packet filtering
- Keep state to identify packets that belong together
  - When a client within the company opens a TCP connection to a server outside the company, firewall must recognize response packets from server and let (only) them through
  - Some application-layer protocols (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

Table 12.2 Example Stateful Firewall Connection State Table (SP 800-41-1)

Source Address	Source Port	Destination Address	Destination Port	Connection State
192.168.1.100	1030	210.22.88.29	80	Established
192.168.1.102	1031	216.32.42.123	80	Established
192.168.1.101	1033	173.66.32.122	25	Established
192.168.1.106	1035	177.231.32.12	79	Established
223.43.21.231	1990	192.168.1.6	80	Established
2122.22.123.32	2112	192.168.1.6	80	Established
210.922.212.18	3321	192.168.1.6	80	Established
24.102.32.23	1025	192.168.1.6	80	Established
223.21.22.12	1046	192.168.1.6	80	Established

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



(d) Application proxy firewall

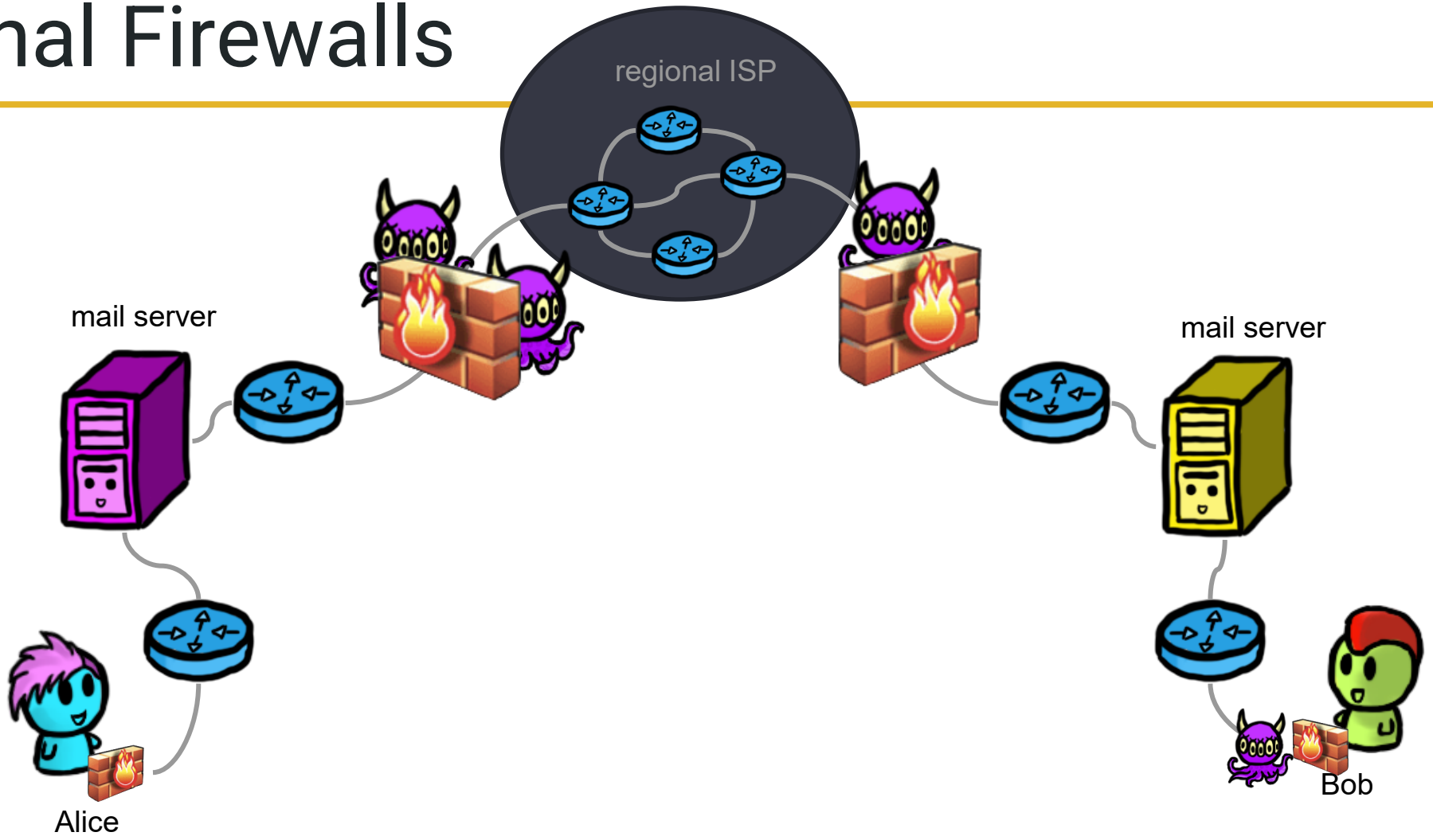
Application Proxy

- 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



Personal Firewalls

- Firewall that runs on a (home) user's computer
  - Usually software based
  - Especially important for computers that are always online
- Primarily for denying unauthorized remote access
- 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?

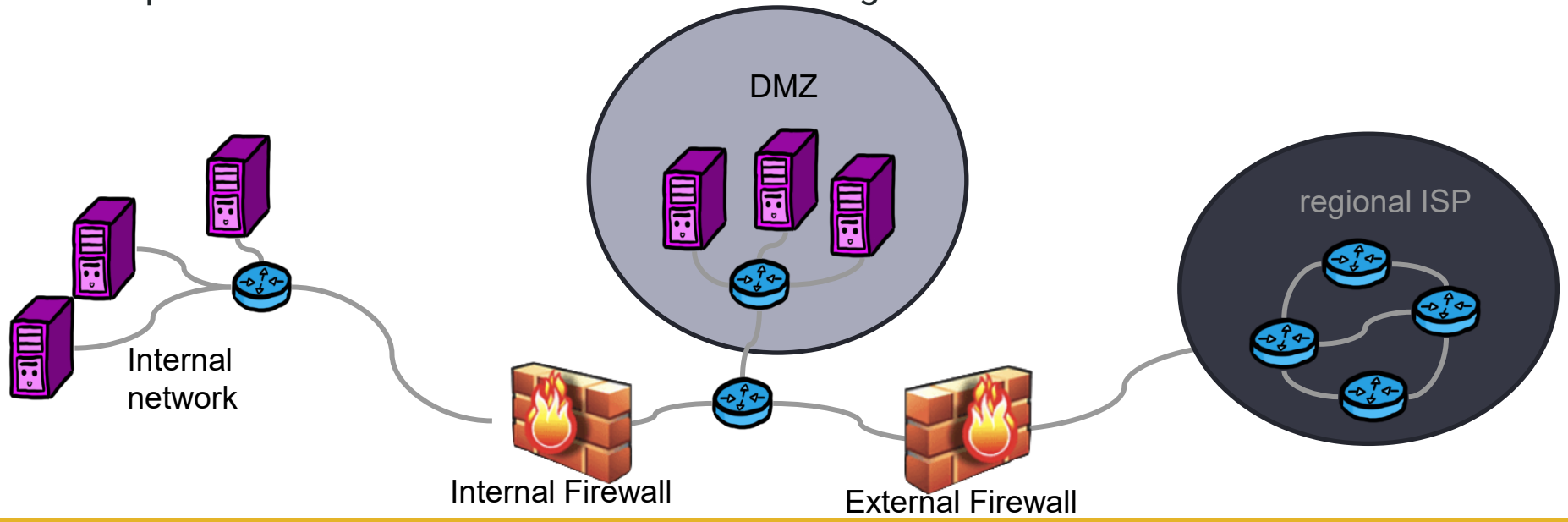
Segmentation and separation

- Don’t put all a company’s servers on a single machine
  - Deploy them on **multiple** machines, depending on their functional and access requirements

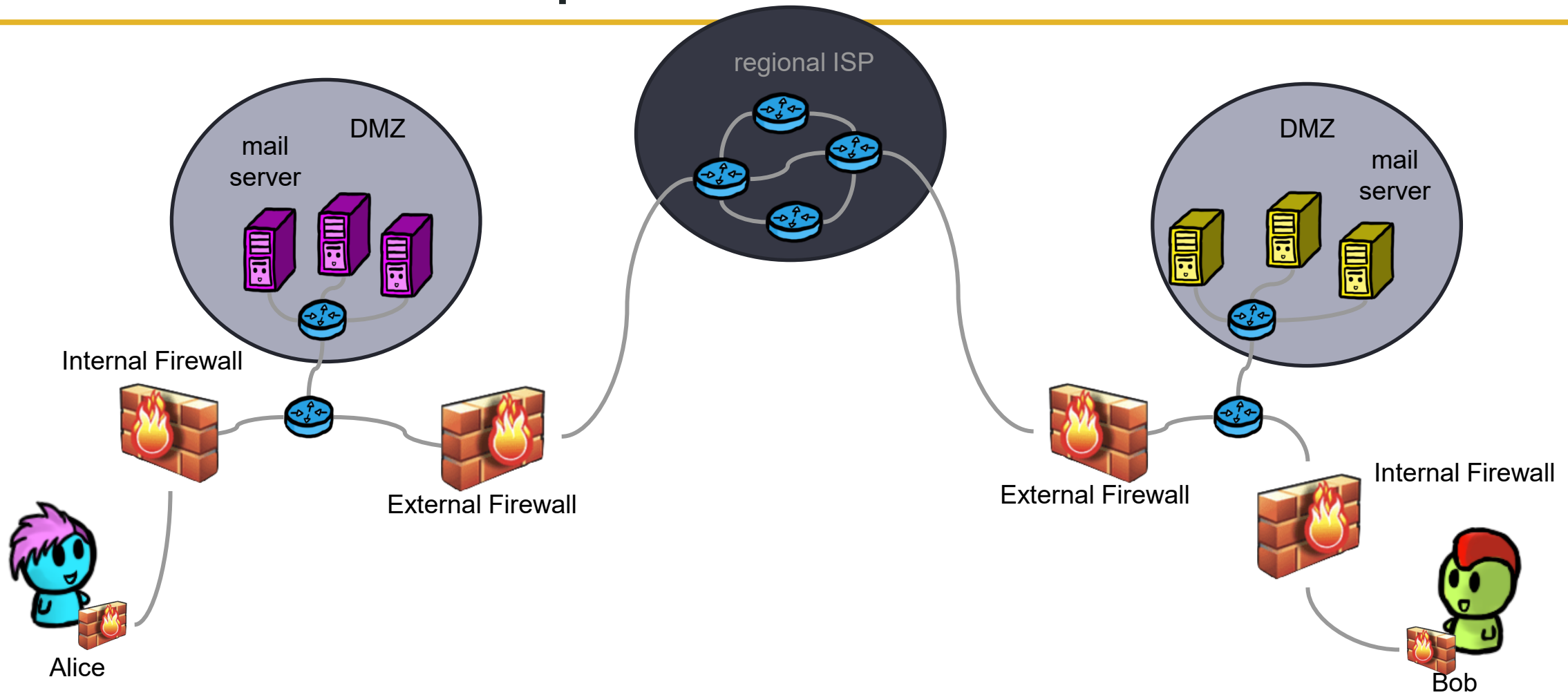
- If a machine gets broken into, only some services will be affected
  - E.g., the web server of a company needs to be accessible from the outside and is therefore more **vulnerable**
  - Therefore, it shouldn’t be trusted by other servers of the company, and it should be **deployed outside** the company firewall

Demilitarized Zone (DMZ)

- Subnetwork that contains an organization's external services, accessible to the Internet
  - E.g. Webserver, email sever, DNS server
- Deploy external and internal firewall
  - External firewall protects DMZ
  - Internal firewall protects internal network from attacks lodged in DMZ



DMZ in our example



A real-life DMZ

Recall redundancy?

- You don’t want to just rely on the firewall.
- What if something gets through?
- **Q:** How can we improve our firewall rules over time?
- **A:** Detection-based defenses as a second line of defence

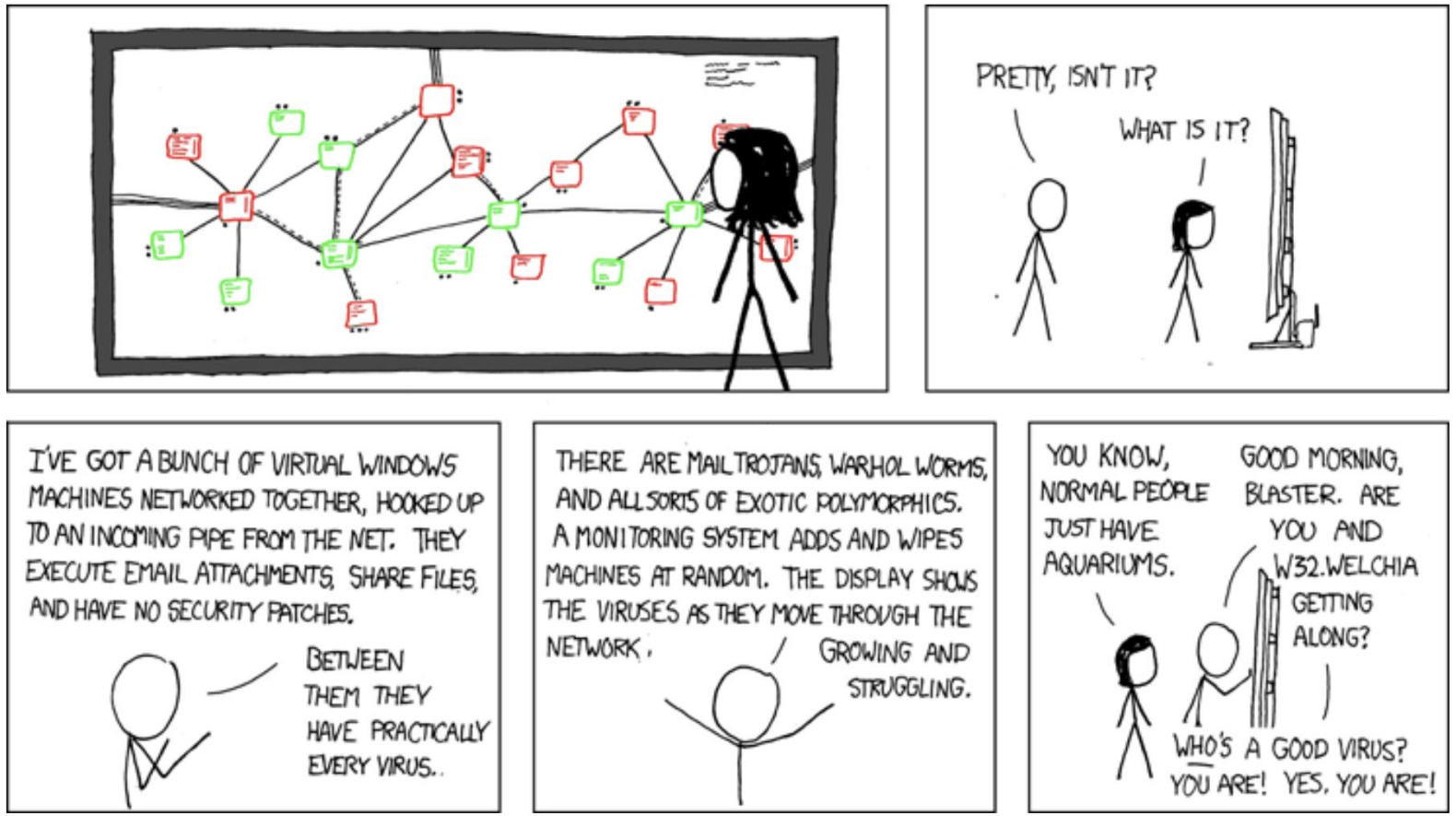
Intrusion Detection Advantages

- If detected fast enough can remove the threat
- Can act as a deterrent as it can be a pain to get around
- Gather intel about the attackers to improve first line of defense

Honeypots / honeynets

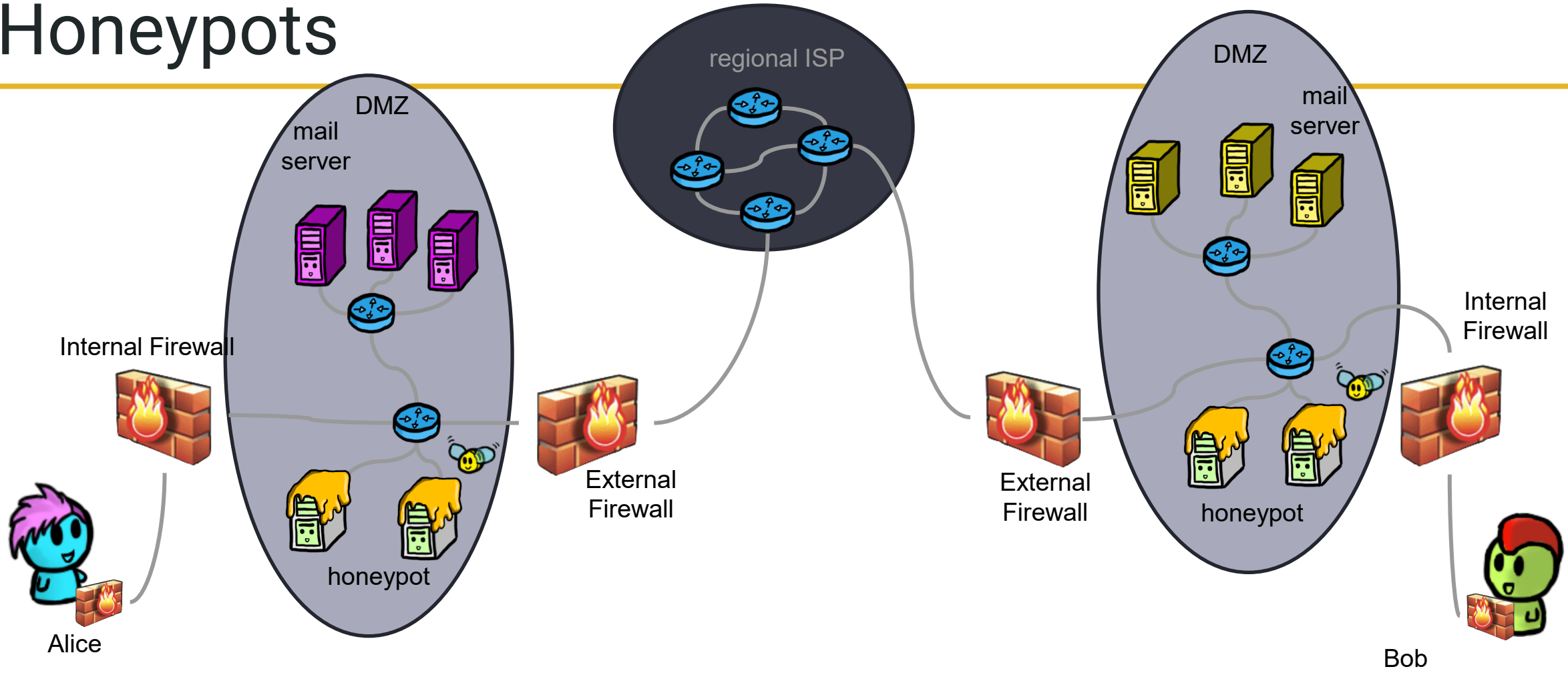
- Set up an (unprotected) computer or an entire network as a trap for an attacker
- System has no production value, so any activity is suspicious
  - Any received email is considered spam
- Observe attacker to learn about new attacks, to identify and stop attacker, or to divert attacker from attacking real system
- The attacker should not be able to learn it targeted a honeypot
- The attacker may use the honeypots to break into real system

Honeypots / honeynets



https://xkcd.com/350/

Honeypots


regional ISP
DMZ
mail
server
Internal Firewall
External
Firewall
honeypot
Alice
DMZ
mail
server
External
Firewall
Internal
Firewall
honeypot
Bob

Types of honeypots/-nets

- Low interaction
  - Server 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 for logging data
  - More complex to deploy
  - Can capture lots of information
  - Can capture unexpected behaviour by attacker

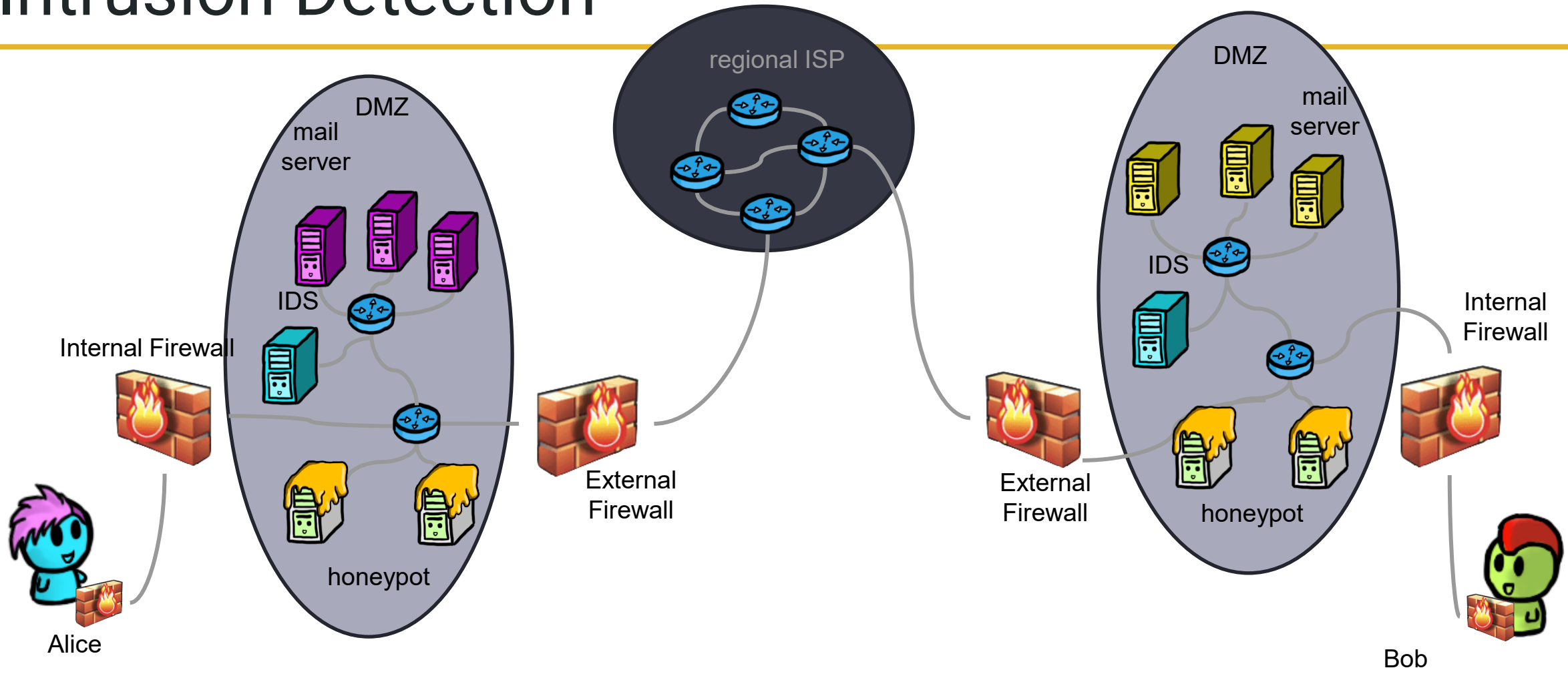
Types of Honeypots

- External
  - Pros: Less risk to internal network, most data collected
  - Cons: Easier to detect
- DMZ
  - Pros: More realistic without endangering the internal network
  - Cons: Needs proper protection for other DMZ services
- Internal
  - Pros: Detects internal attacks
  - Cons: If compromised it’s inside the network

Intrusion detection systems (IDSs)

- Firewalls do not protect against inside attackers and are not perfect
- 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

Intrusion Detection



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

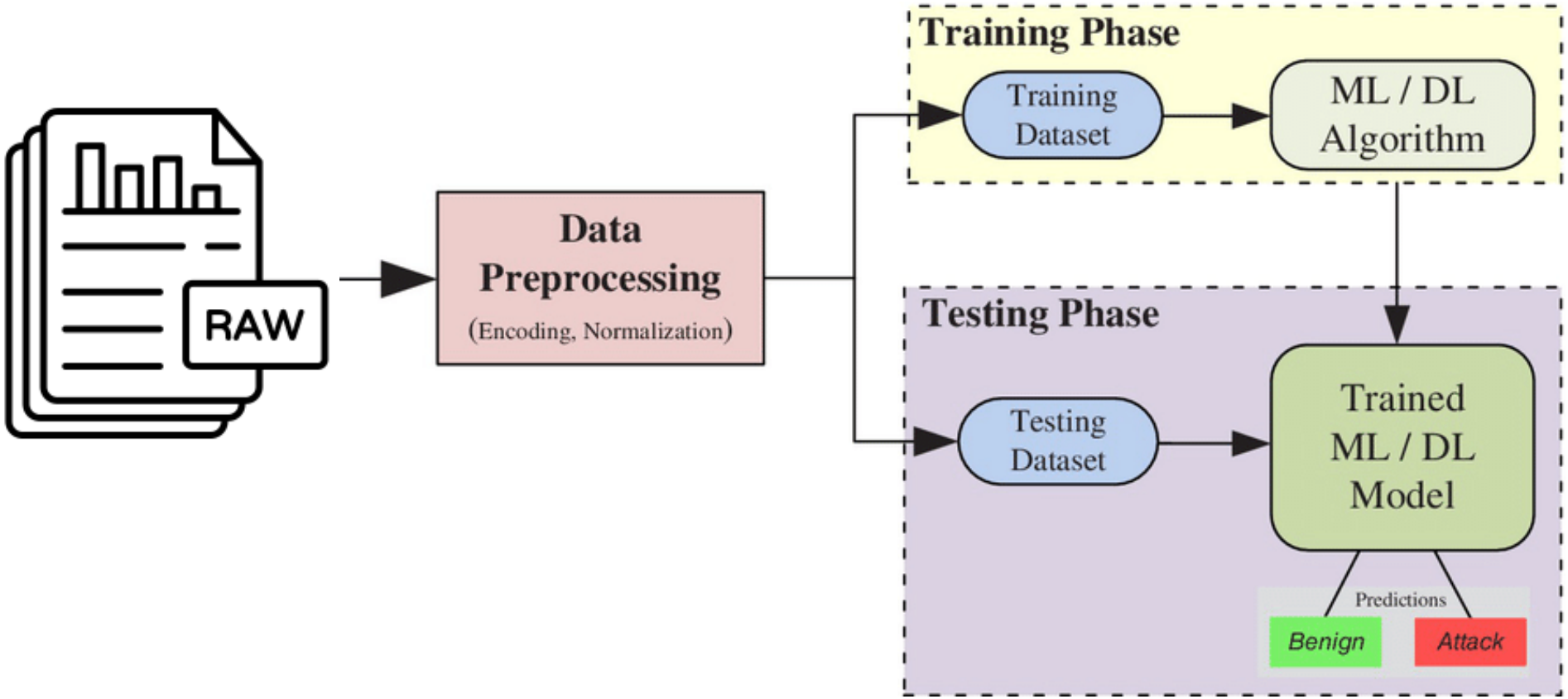
Signature-based IDSs

- 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
  - Hypothesis test or threshold for frequency of certain events

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

ML Based Intrusion Detection



ML Based Intrusion Detection

- Just because we can does not always mean we should!
- Typically, interpretable classifiers (trees, KNN)
- Need representative training set
  - Out of distribution samples can be a problem
- Keep parameters secret
  - Adversarial Examples (more later)
- Privacy issues (more later)

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

- False positives 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

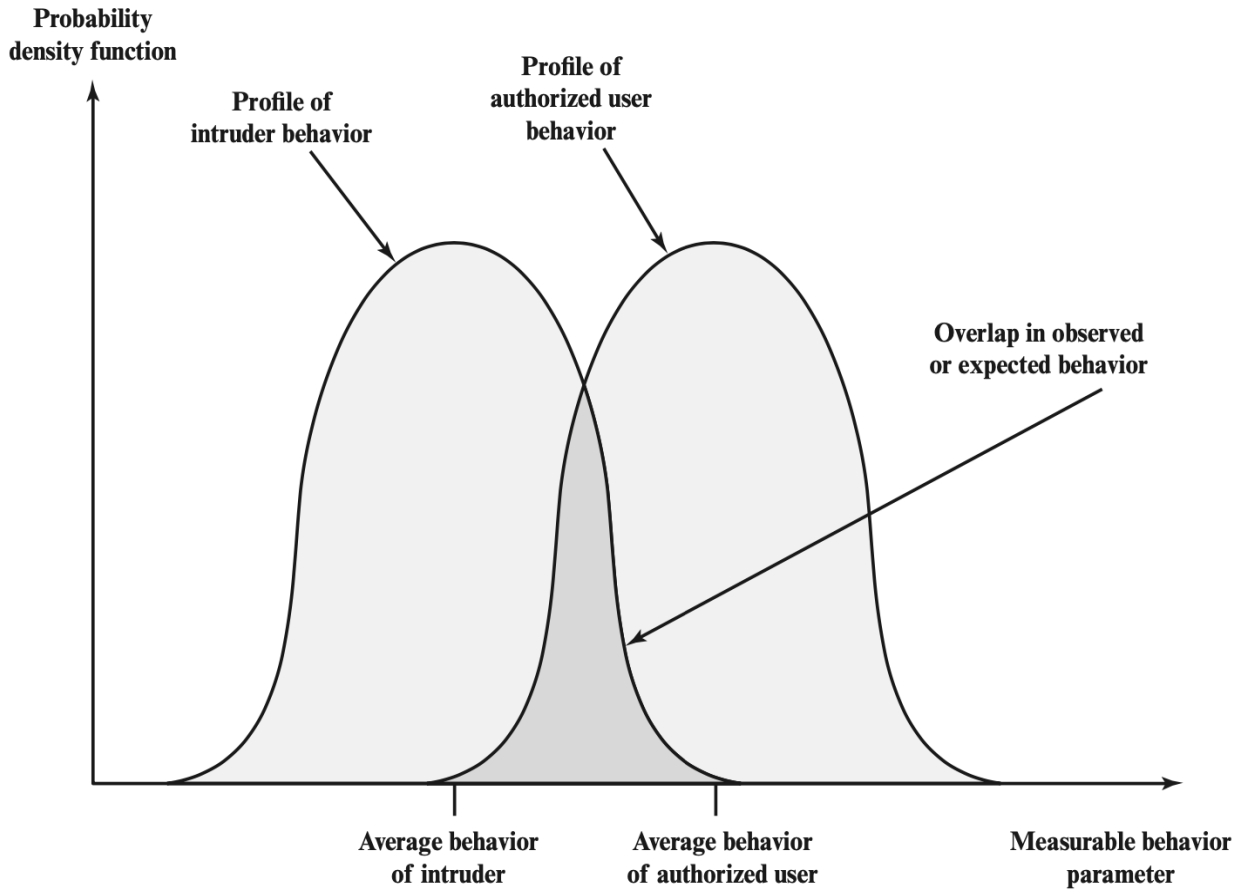


Figure 11.1 Profiles of Behavior of Intruders and Authorized Users

Example: Snort

- 3 modes
  - Sniffer, Logger, IDS
- IDS is based on a list of rules
  - Can be similar to firewall or more sophisticated
  - Each rule an alert, log, etc


```
alert tcp $EXTERNAL_NET 80 -> $HOME_NET any
(
    msg:"Attack attempt!";
    flow:to_client,established;
    file_data;
    content:"1337 hackz 1337",fast_pattern,nocase;
    service:http;
    sid:1;
)
```

Example: Tripwire

- Anomaly-based, host-based IDS, detects file modifications
- **1st step:** Initially, compute digital fingerprint of each system file and store fingerprints at a safe place
- **2nd step:** Periodically, re-compute fingerprints and compare them to stored ones
- (Malicious) file modifications will result in mismatches
  - Q: Why is it not a good idea to perform the second step directly on the production system?