https://brown-csci1660.github.io

CS1660: Intro to Computer Systems Security
Spring 2026

Lecture 20: Network Security II

Instructor: **Nikos Triandopoulos**

April 16, 2026

BROWN

Last class

- Cryptography
- Authentication
- Web security
- OS security
  - Access control, OS access control, file-system access control
  - Software security: More on race conditions, isolation, malware
  - More on malware, malware detection
- Network security
  - Cloud security, email security, APTs, SIEM & security analytics
  - Background, specific attacks, …

Today

- Cryptography
- Authentication
- Web security
- OS security
  - Access control, OS access control, file-system access control
  - Software security: More on race conditions, isolation, malware
  - More on malware, malware detection
- Network security
  - Cloud security, email security, APTs, SIEM & security analytics
  - Background, specific attacks, …

20.0 More on APTs

Recall: Cyberweapons

Starting from 2010 several viruses acted as a sort of weapons in international relationships

- Usually this is not confirmed by governments
- Most famous:
  - 2010 Stuxnet
  - 2012 Flame
  - 2017 Equifax
  - 2020 Orion Solarwinds

SolarWinds hack

A supply-chain attack against an IT-management company

- Hackers able to compromise networks of many other companies and deliver malware
- Malware inserted into update of Orion, a network-management product
- Hackers used AWS as a disguise
- At least 100 companies impacted (e.g, Microsoft, US government agencies)
- FireEye, an impacted cybersecurity company discovered the hack
- SolarWinds issued a security advisory + what defensive measures could be taken
- FBI Investigation to find the actors
- “solarwinds123” used as a server password (security researcher warned SolarWinds of this!)

SolarWinds hack (cont.)

Why were basic security practices neglected at SolarWinds? Under-investment in security…

- "Employees say that under [CEO] Mr. Thompson ... every part of the business was examined for cost savings and **common security practices were eschewed because of their expense**. His approach helped almost triple SolarWinds' annual profit margins to more than $453 million in 2019 from $152 million in 2010."
- Bruce Schneier: "The market does not reward security, safety or transparency. It doesn't reward reliability past a bare minimum, and it doesn't reward resilience at all."
- Core problem: Limited economic incentives to invest in cybersecurity
  - Expense with diminishing returns
  - Limited legal liability
  - Small factor in customers' decisions, small effect on share price
  - Negative impact to supply chain security

Investment in security

Gordon-Loeb model

- Even with optimal incentives, firms will never invest more than 37% of expected damage from security breaches in cybersecurity

Reason

- Cybersecurity is not generating profit and having diminishing returns on investment
- If you expect fire damage to cause $10,000 damage, it doesn't make sense to purchase a $10,000 device that reduces the probability of fire damage

Result

- Total damage caused by cybersecurity will always significantly exceed investment in cybersecurity

Legal liability

Having poor cybersecurity is legal

- Limited laws regulating cybersecurity standards

Federal Trade Commission relies on “unfair or deceptive acts” to press charges

Customers and shareholders need extreme cases of negligence or false statements

- Class-action lawsuit against SolarWinds by shareholders only because they allege false and misleading statements

If an honest effort is made, very little legal risk in having bad cybersecurity

Lack of business consequences (or accountability)

Over time, customers tend to forgive and forget data breaches

- Equifax, eBay, Adobe, and Marriott all recovered from their breaches

In corporate context

- Incentives in procurement favor functionality and cost over possible cybersecurity risks
- Difficult to evaluate cybersecurity between companies
- Share prices usually drop heavily after a data breach, but studies show a negligible long-term effect
- More recent data breaches have had smaller share price drops due to “breach fatigue”

Supply chain issues

Centralization of software

- Creates points of vulnerability that dramatically reduce hacker effort
  - SolarWinds allowed hackers to access 18,000 systems
- Vulnerabilities in one company's product have cascading effects
  - Beyond their immediate customers
  - CISA: 30% of SolarWinds victims did not use SolarWinds
- Example: 2017 NotPetya attack
  - Malware deployed by a malicious automatic update in MeDoc (Ukrainian tax preparation software)
  - Caused $10 billion damage
  - Damaged pharmaceutical production, global shipping, hospital systems

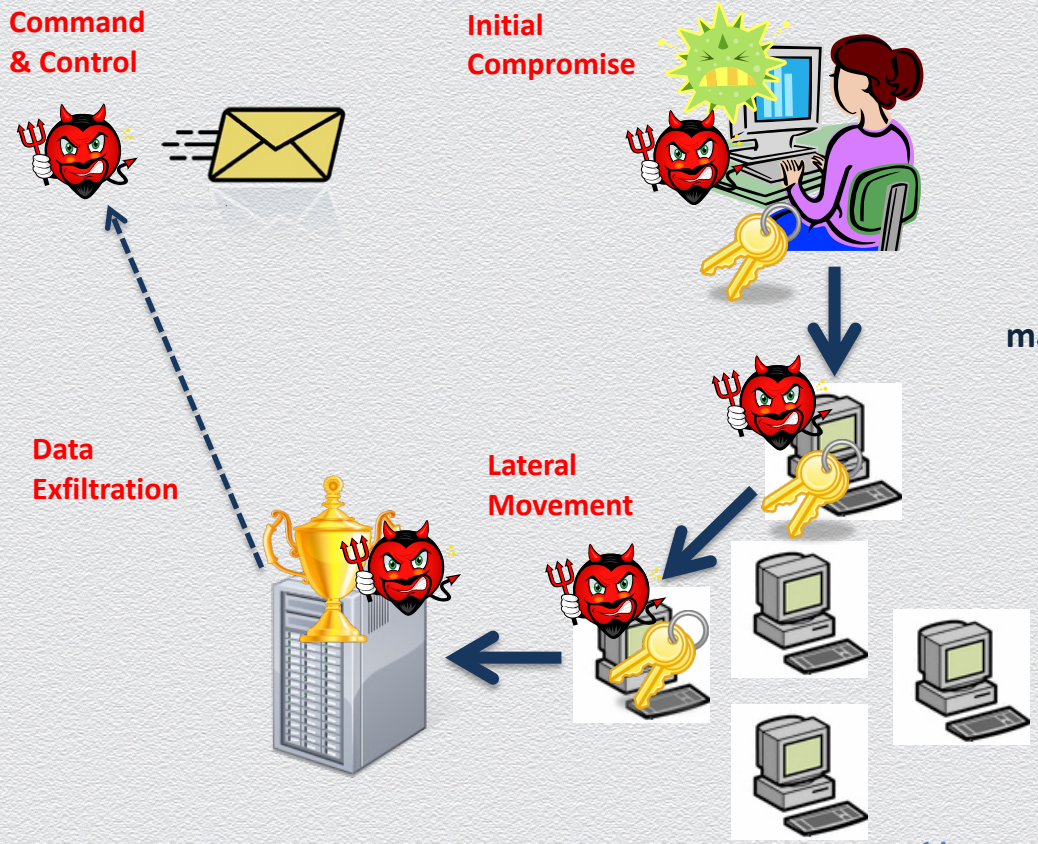
20.1 PillarBox (SIEMs & security analytics)

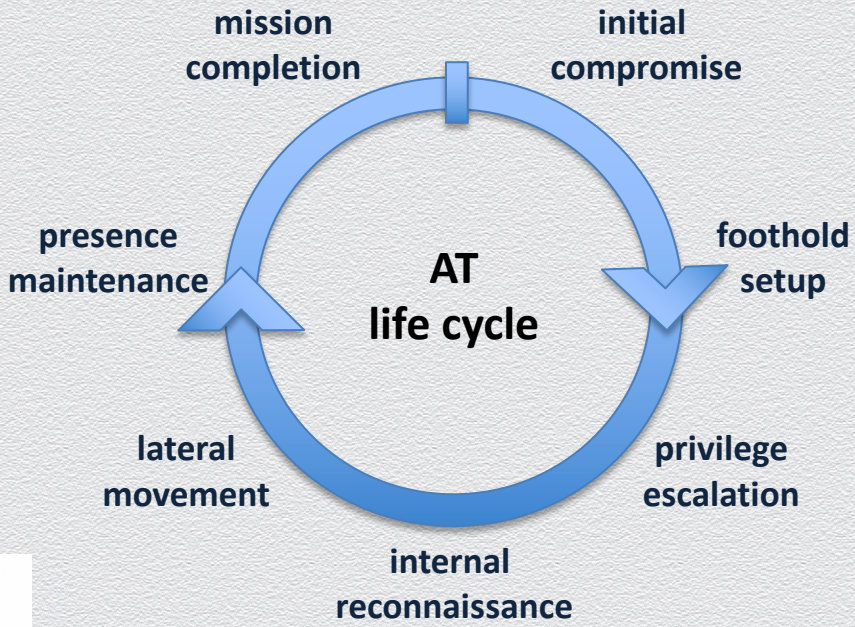
Advanced Threats: Enterprises' toughest enemy

- Advanced Threats (ATs) are a serious risk facing enterprises today
  - comprise well-targeted, persistent attacks
  - aim at unauthorized data manipulation or exfiltration
  - employ rich attack vectors and unknown strategies
    - social engineering
    - zero-day malwares / vulnerabilities
    - low-and-slow progression

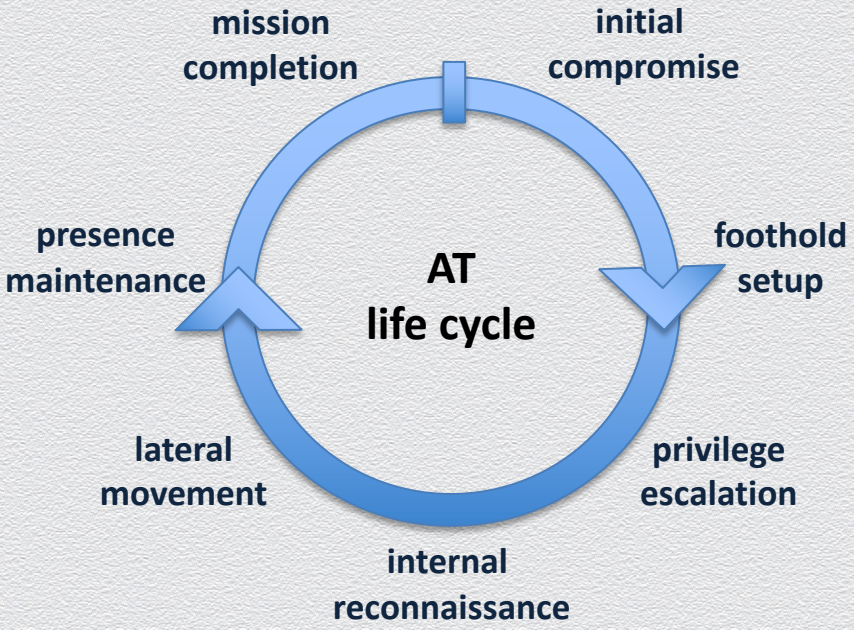
➔ Extremely hard-to-defend, often even hard-to-detect

The “canonical” attack cycle





Best defenses



Best defenses in security industry

- Tighter preventative practices
  - raise the protection fence
    - e.g., multi-factor authentication, data protections, access control, etc.
- **Detection & forensics tools**
  - visibility – analysis – action
    - e.g., security information event management (SIEM) systems, security analytics

Intelligent-based security

Security Operation Center (SOC)

Command & Control

Initial Compromise

Data Exfiltration

Lateral Movement

Security Information Event Management (SIEM)

Intelligent-based security (cont.)

Visibility into enterprise infrastructure

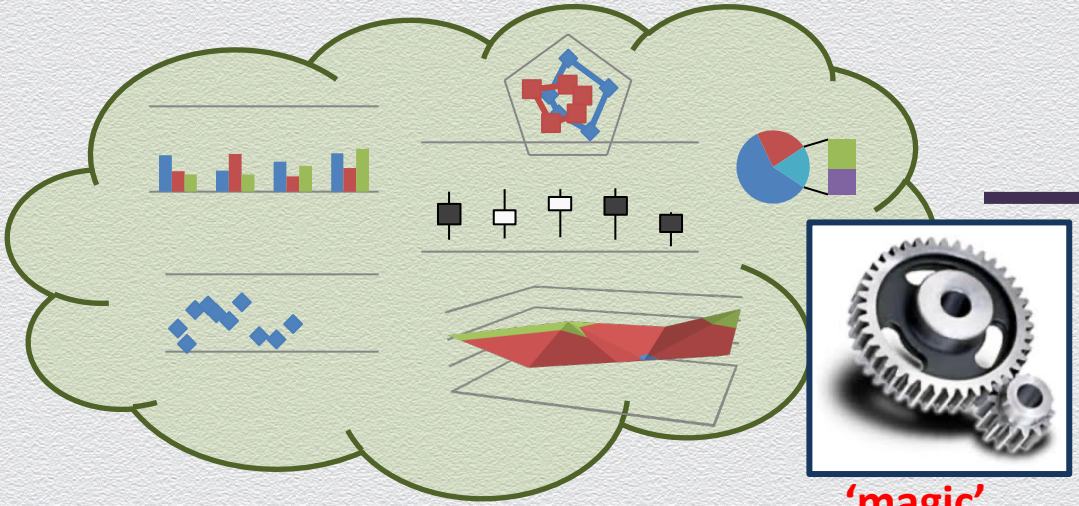
- Monitor complex systems via continuous "health reports"
- Collect device logs and/or special security-related alerts
- Correlate received data across different endpoint devices
- Analyze collected information for incident-respond decision making

Security Analytics Sources (SASs)

- Firewalls
- VPNs
- Endpoint instrumentation
  - Intrusion-detection systems (IDSs)
  - Syslog
  - Anti-virus engine
  - Other alerting facilities

(Big-)Data analytics for security

1. gather evidence from various agents



'magic'

4. detect anomalies

2. profile normal activities

3. label unusual activities/events

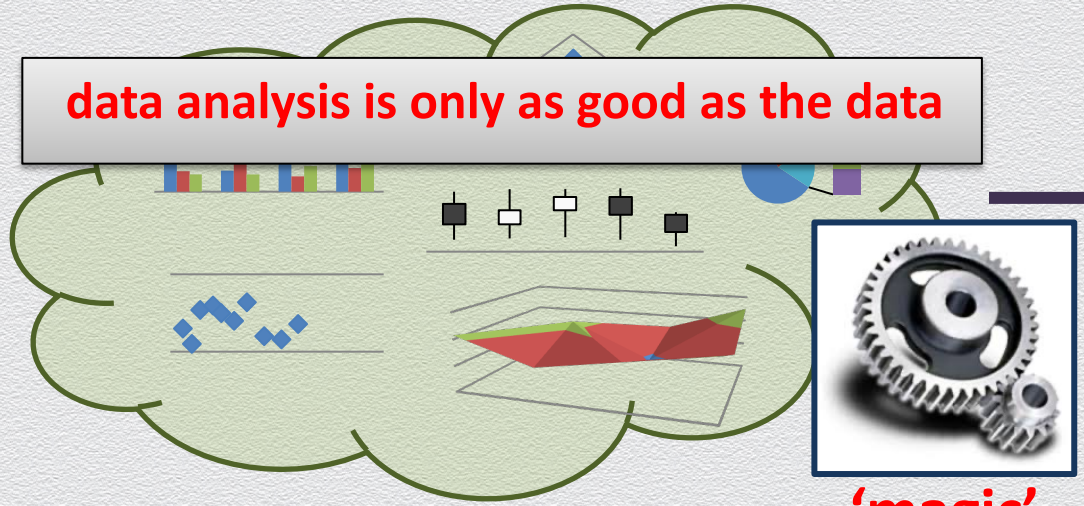
(Big-)Data analytics for security (cont.)

1. gather evidence from various agents

4. detect anomalies

data analysis is only as good as the data

complex big-data problem



‘magic’

2. profile normal activities

3. label unusual activities/events

Next-generation Advanced Threats

- Tougher, evolving adversaries who
  - grow in sophistication to become context aware and target specific
  - know "*what they attack and how it is protected*"
  - shift towards qualitatively stronger attack strategies

Achieve their objective
<u>while trying to evade</u>
defensive tools

(past / current)

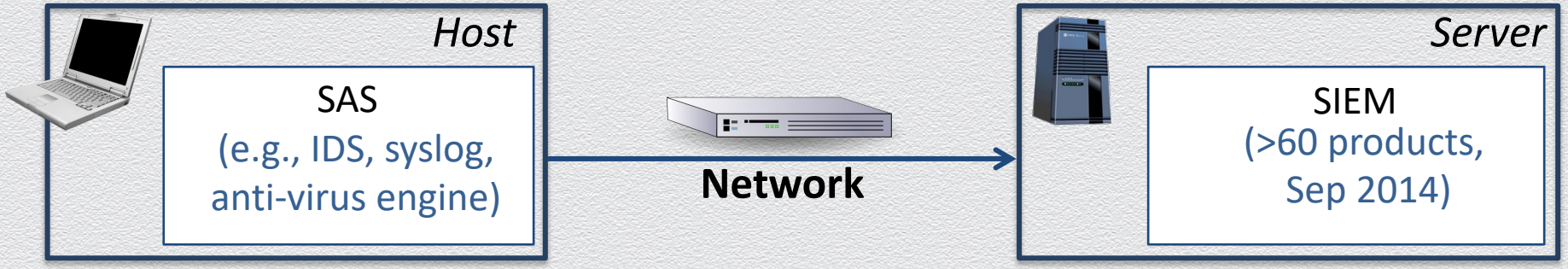
Achieve their objective
<u>by first disarming</u>
defensive tools

(current / future)

In practice this means…

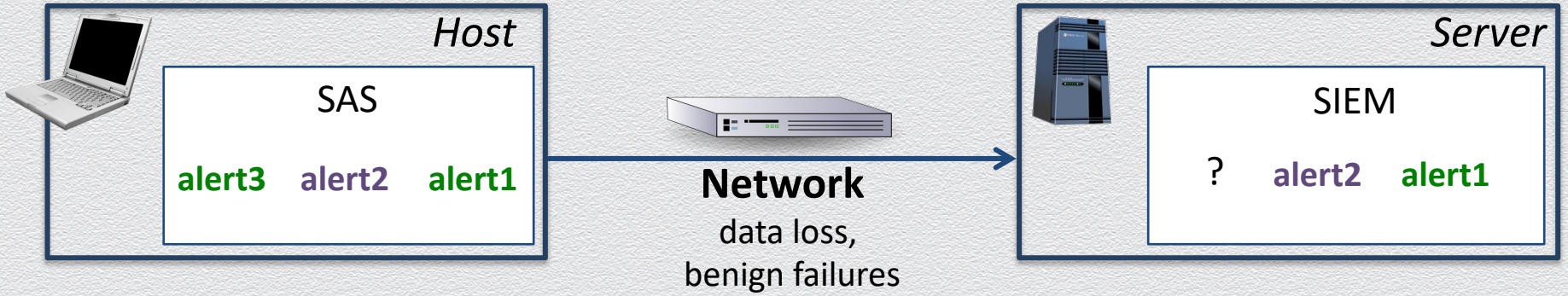
- If strong authentication is used, the attacker will steal
  - stored keys to clone authenticators
  - passwords to impersonate users
  - credentials to forge signatures
- If security logs are collected and analyzed, the attacker will
  - block the stream of reported logs
  - employ log-scrubbing malware to cover its tracks
  - tamper with host-side log generation software

Problem: Secure chain of custody in security analytics



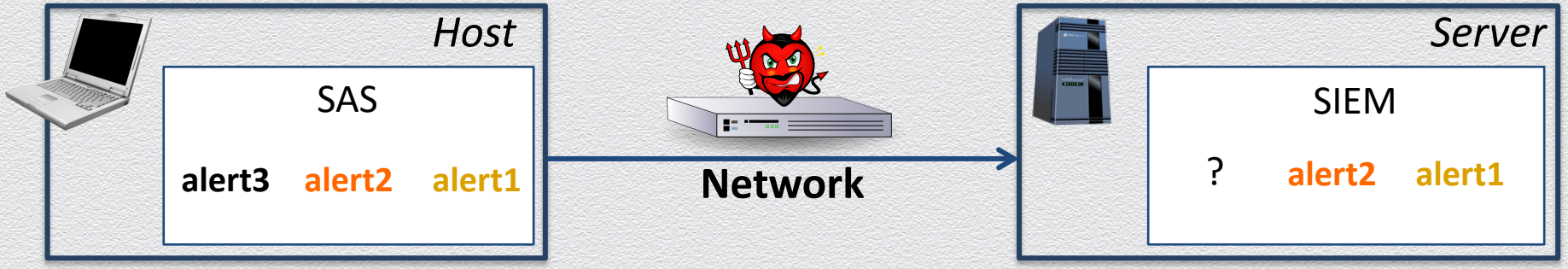
Problem: Secure chain of custody in security analytics

- Security alert systems often employ unreliable channels!



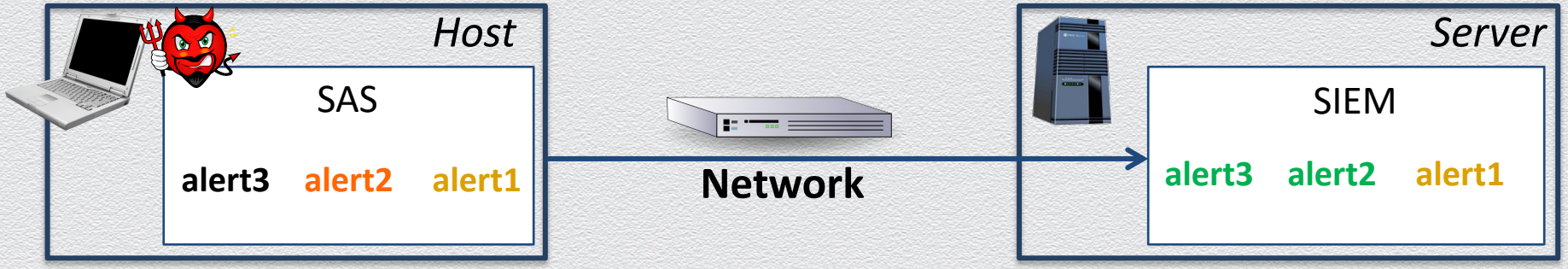
Problem: Secure chain of custody in security analytics

- Security alert systems can employ adversarial channels!



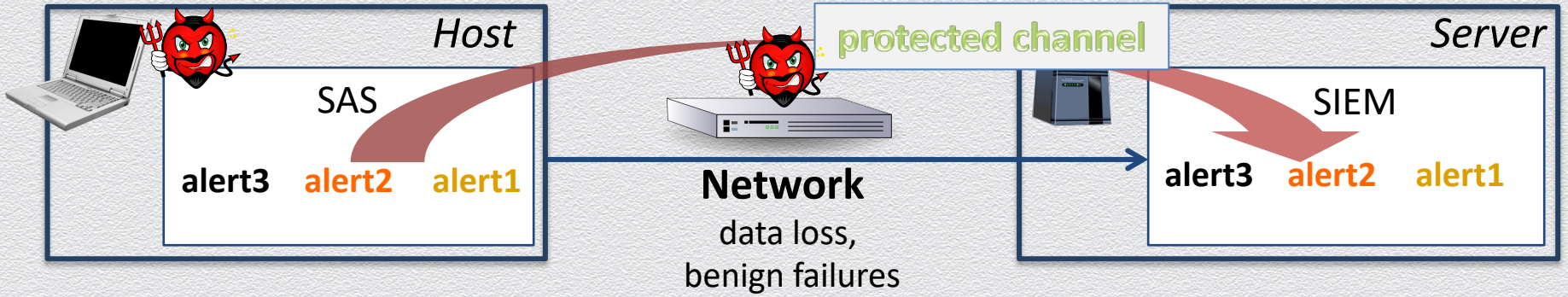
Problem: Secure chain of custody in security analytics

- Security alert systems constitute a direct target of a '2nd-wave' AT!
  - an attacker may discover, observe or read alert transmissions
    - …and accordingly adapt its attack strategy based on SAS behavior!
  - an attacker may tamper, suppress or block alert transmissions
    - …and eventually disrupt SAS functionality (e.g., using log-scrubbing malware)!



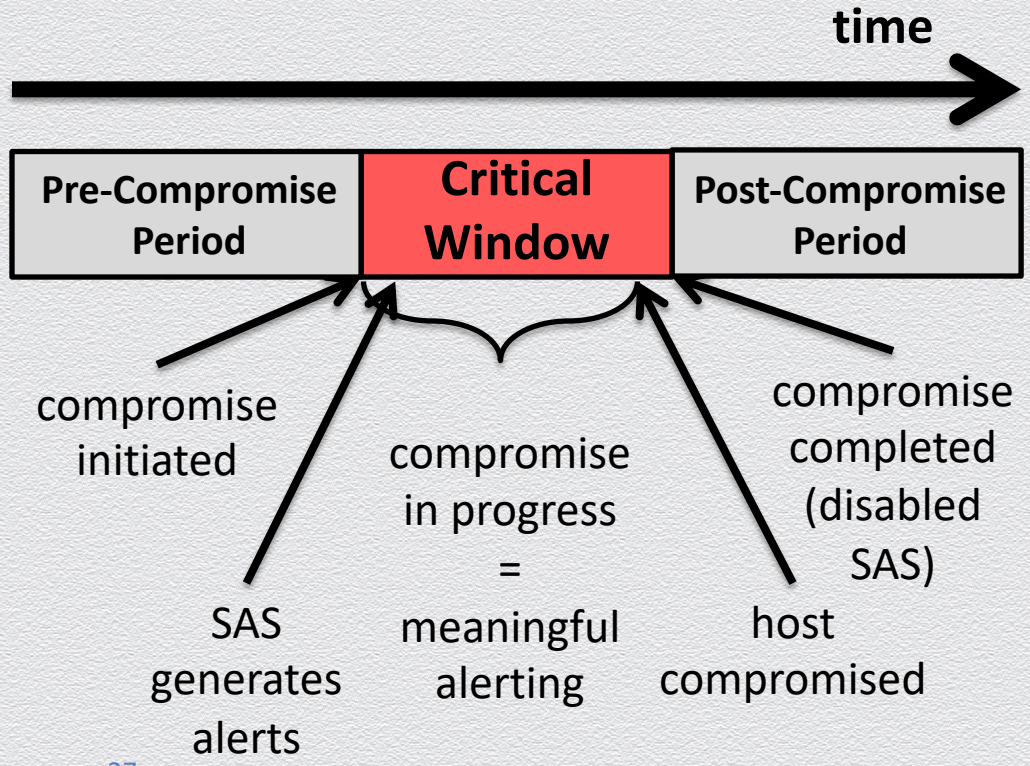
Solution: PillarBox, a secure alert-relaying tool

- Features
  - ensures against alert suppression or tampering
  - conceals alerting activity
  - features self-protection, transmits alerts persistently
  - is agnostic of the exact SAS in use



Which alerts can be relayed and thus protected?

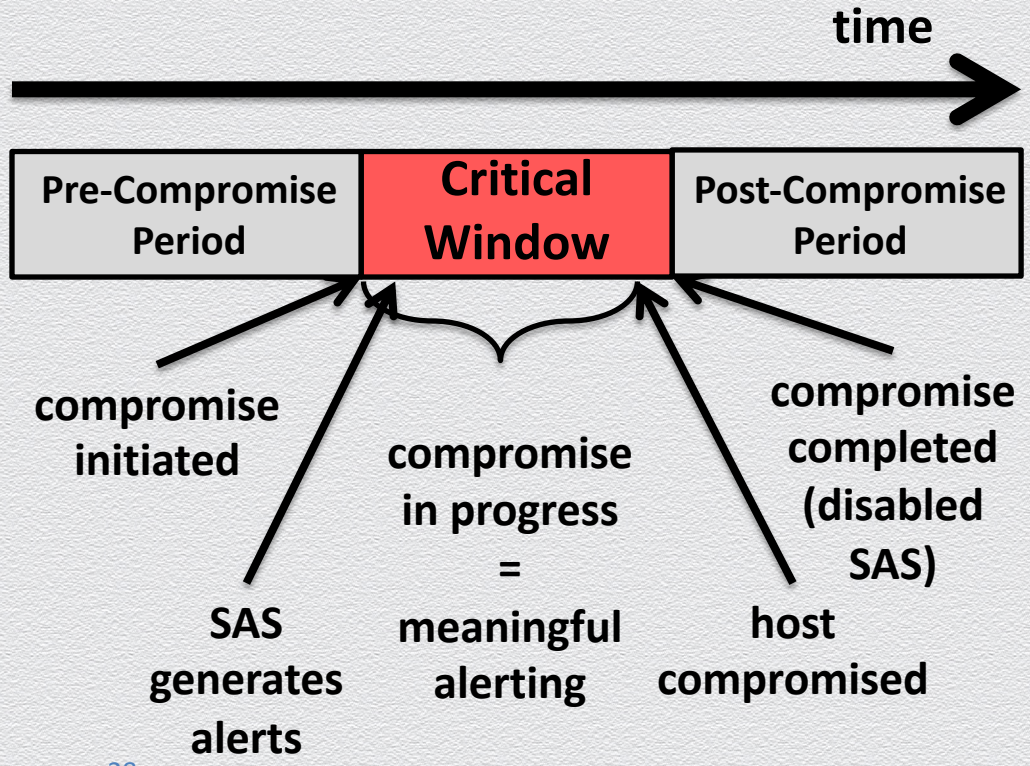
- As a security exploit unfolds on a host, it often produces strong indications visible to resident security software
  - e.g., privilege escalation, registry-entry change, reboot, anti-virus / IDS alert
- A SAS can generate meaningful alerts only while intrusion is in progress
  - e.g., a fully compromised host will automatically shut the SAS down
- We must assume a non-zero critical window of high-indicative alerts



Which alerts can be relayed and thus protected?

race condition between malware & SAS

our goal is to protect alerts **in self-protection** mode, i.e., when SAS is being actively by attacker knowing its operational setting



On-the-fly transmission Vs. buffering of alerts

Push alerts (instantly)

- Unreliable / failed transmissions
  - BYOD hosts, UDP syslog transport
- Network attacks to suppress alerts
  - ARP stack smashing, DoS attack
- SAS intelligence leakage
  - via outbound traffic analysis
- Delayed alert custody
  - prolonged attack window

Buffer alerts

- Persistence
  - alerts are transmitted redundantly
- Regularity / Periodicity
  - in-traffic alert suppression is detected
- Stealth
  - alerting behavior is concealed
- Speed
  - engineered to lock alerts very fast

Vulnerability of on-the-fly alerting

We successfully performed the following attack:

1. use stack overflow exploit in SSH v1 to inject a shell
2. use “Full Nelson” local privilege escalation to gain root access
   - e.g., vulnerabilities CVE-2010-4258, CVE-2010-3849, CVE-2010-3850
3. use automated script to read snort.conf
4. learn if the attack was detectable
5. remove snort log from log file

1. Buffering alerts

(FS) integrity ✓

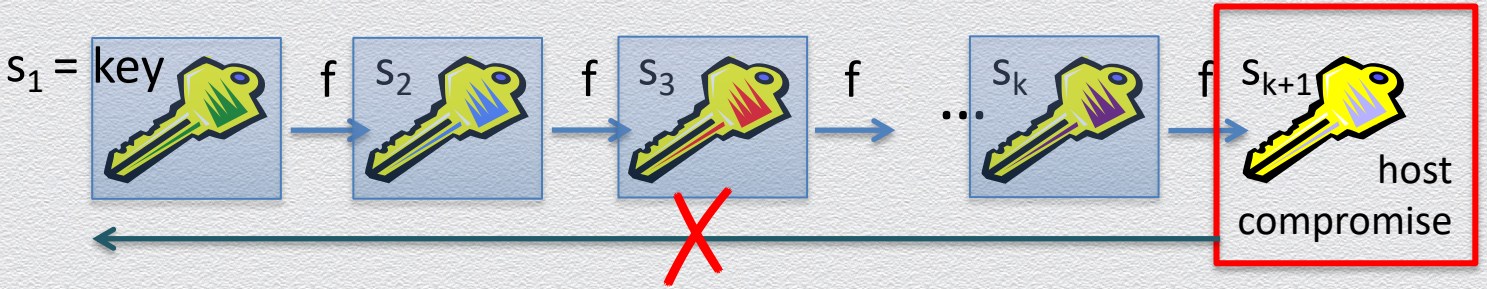
(FS) confidentiality ✓

- As soon as they are generated, alerts are
  - **signed** and **encrypted** using a forward-secure secret key (shared by the server and host) and then stored in a buffer at the host
  - periodically or on demand (e.g., every t alerts) **transferred** to the server

Forward-secure key updates

Attacker can't learn post-compromise cryptographic keys

- update key irreversibly through one-way hashing



2. Retransmitting alerts

(FS) integrity ✓

(FS) confidentiality ✓

persistence ✓

- As before, but now alerts
  - are **not deleted** from buffer but are **transferred redundantly**
    - e.g., when a new alert is generated all buffered alerts

persistence:
missing alerts can only be attributed to an attack, thus allowing to signal a “meta alert”

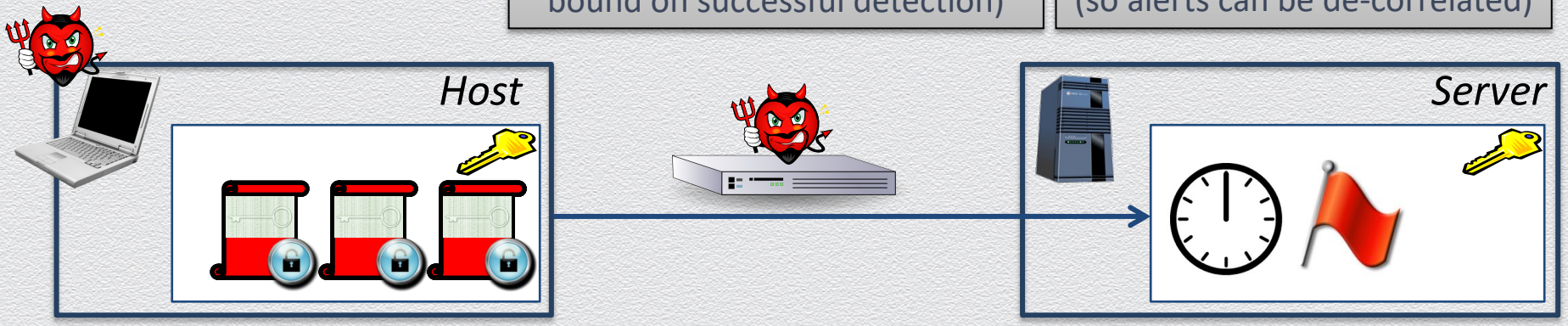
3. Checking heartbeat

(FS) integrity ✓
(FS) confidentiality ✓
persistence ✓
failure detection ✓
traffic concealment ✓

- As before, but now alerts
  - are transmitted **periodically** (in regular time intervals)
  - if failed to reach the

failure detection:
imposes a minimum frequency of transmission (allows an upper bound on successful detection)

traffic concealment:
imposes a regular pattern of transmissions (so alerts can be de-correlated)

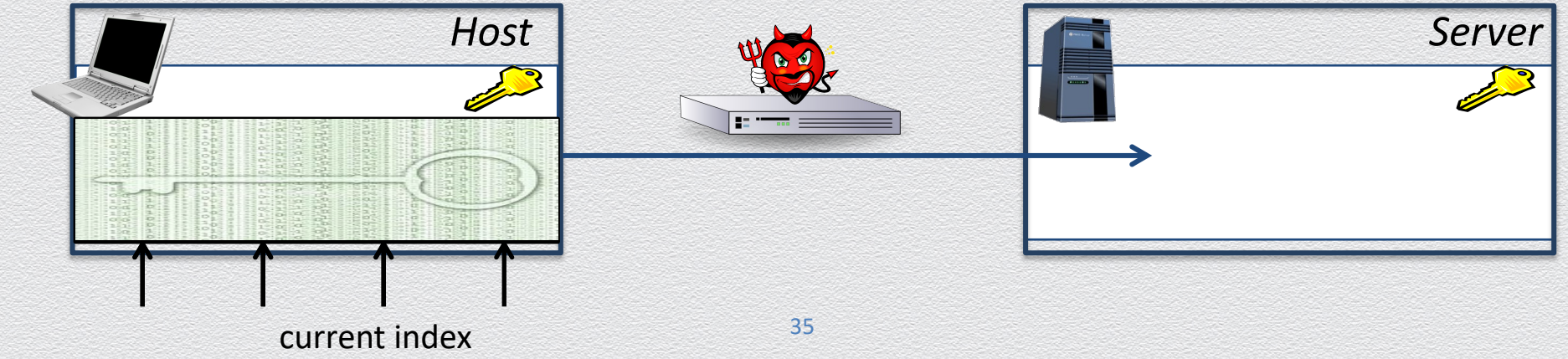


4. Encrypting fixed-size buffers

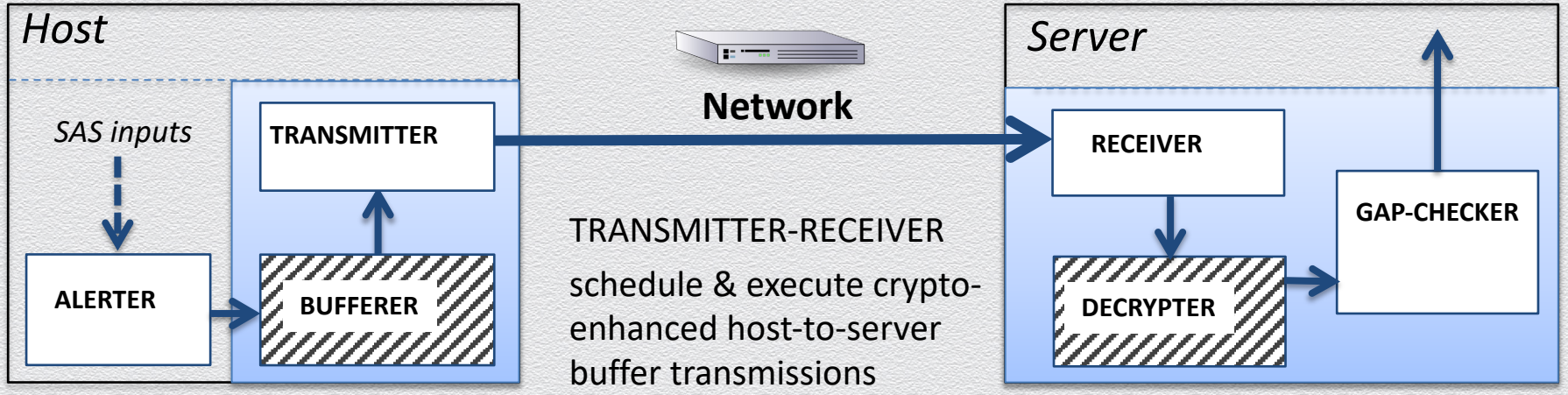
(FS) integrity ✓ failure detection ✓
(FS) confidentiality ✓ traffic concealment ✓
persistence ✓ stealth ✓

- As before, but now alerts
  - are stored in an initially **random, fixed-size** buffe
  - are transmitted periodically **encrypted** as a whol
  - if failed to reach the server, they signal a **"gap al**

stealth:
alerting mechanism is completely hidden from attacker
(at some communication overhead)



PillarBox architecture



TRANSMITTER-RECEIVER

schedule & execute crypto-enhanced host-to-server buffer transmissions

ALERTER

implements SAS, monitors host to identify events against a set of alert rules, creates alert messages and relays them to BUFFERER

BUFFERER-DECRYPTER

implement crypto-assisted reliable channel & report integrity failures

GAP-CHECKER

reconstructs alert stream, checks for missing alerts, reports “heartbeat” or “gap alert” failures

PillarBox prototype

- Implementation in C++, Inter Xeon E5506, 2.13GHz, quad-core, 4MB L2 cache, 3GB RAM, Red Hat Enterprise Linux WS v5.3 x86_64
- ALERTER: Snort, configured to log events indicating impeding compromise
- BUFFERER: data structure residing in main memory
  - buffer size decided at creation time, dynamically updated on based on network disruptions
  - registers a watch on the log file used by ALERTER to grab new logs
- TRANSMITTER: thread waking after a specified interval decided at run time
  - Authenticated encryption using EAX-mode encryption
  - Custom FS pseudorandom generator for efficient irregular key updates
- GAP-CHECKER: two threads checking heartbeat and gap-alert failures

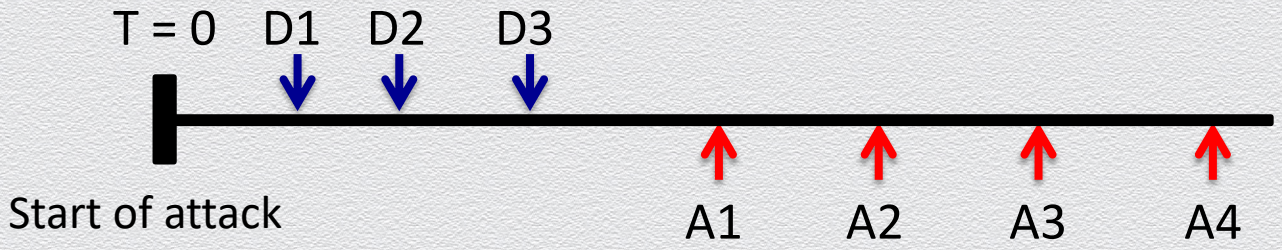
Buffer Vs. on-the-fly alerting

- With PillarBox alerts can be generally locked away in the buffer **faster** than they can be sent over the network to a remote server
  - logs are grabbed from Snort as they are generated and sent directly onto the network using UDP

System	Average	Std. Dev.
PillarBox alert locking	163.06µs	49.09µs
onto-the-wire alert transmission	251.78µs	41.00µs

Race condition: ideal ordering

Non-zero critical window is necessary for PillarBox to be effective



D1 : Attack detected and logged

D2 : Attack secured by PillarBox

D3 : Triggered alert rule deleted

A1 : Remote attack completes

A2 : Privilege escalation

A3 : Attacker learns detection rules

A4 : Attacker deleted logged file

Race condition: results

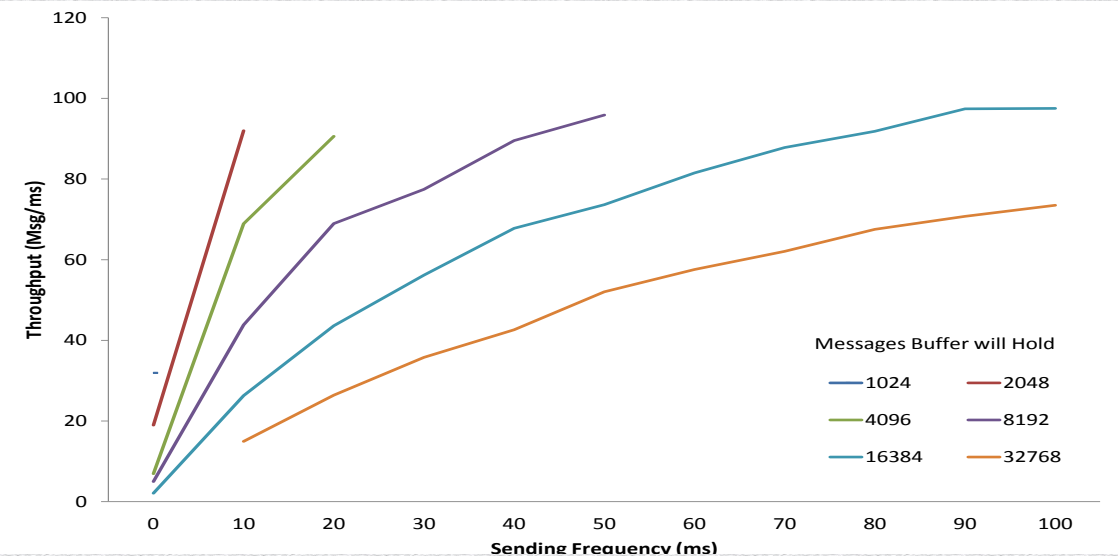
PillarBox consuming alerts from Snort can secure alerts before full host compromise

Event	Average (µs)	Std. Dev. (µs)
D1 : Attack detected and logged	1,645.441	1,069.843
D2 : Attack secured by PillarBox	1,645.609	1,069.842
D3 : Triggered alert rule deleted	1,645.772	1,069.840
A1 : Remote attack completes	2,692.536	1,324.419
A2 : Privilege escalation	2,693.474	1,324.432
A3 : Attacker learns detection rules	2,696.524	1,324.919
A4 : Attacker deleted logged file	2,696.590	1,324.990

Overhead and throughput

PillarBox adds an additional 7% overhead compared to running Snort

PillarBox is fast enough that prevent it from being a bottleneck



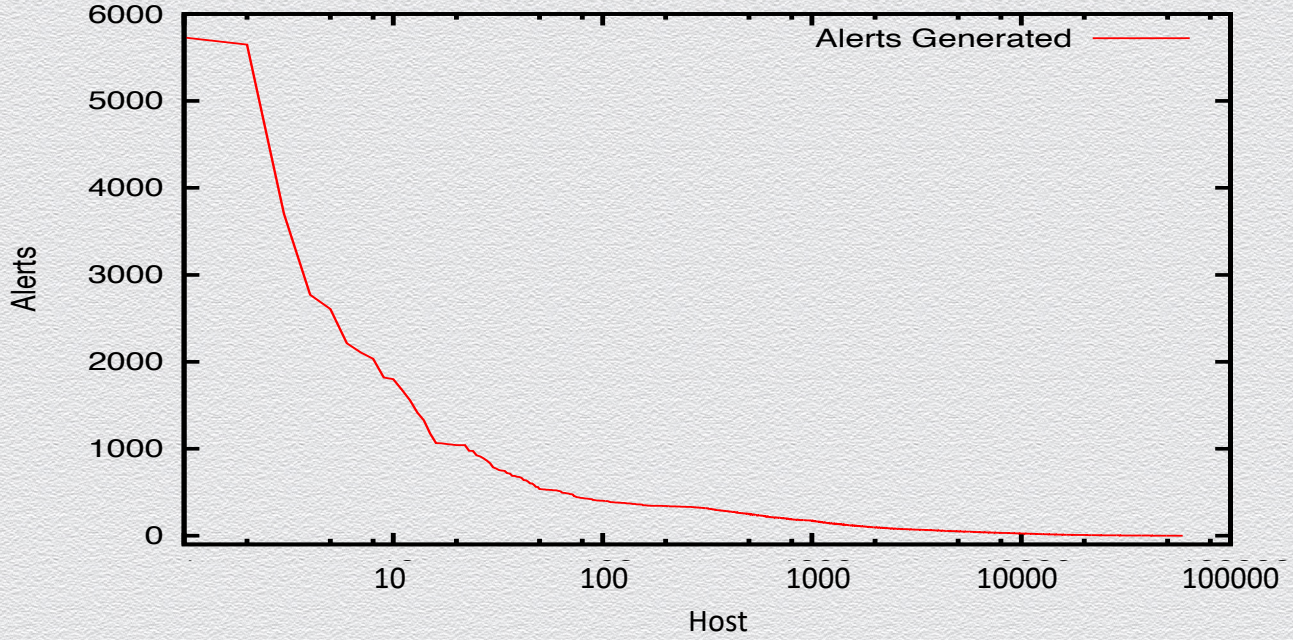
able to process nearly 100,000 alert messages per second,
well above the recording rate achievable by Snort

typical Snort alert messages (a few hundred characters)

Observed alerting frequencies

Analysis of a large enterprise dataset (59,034 users) across period of 7h:

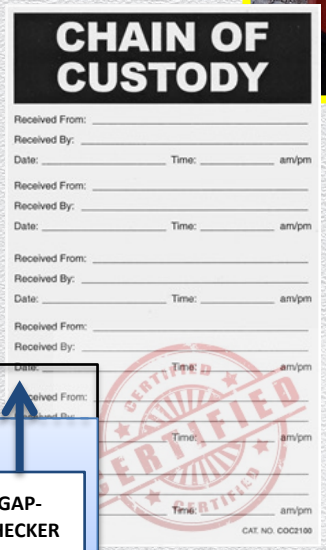
busiest machine: 8603 alerts; average across all machines 18.3 alerts

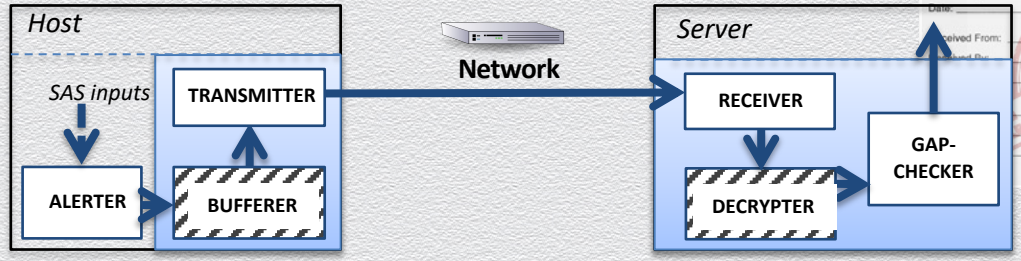


by designing PillarBox to handle a throughput of 1 alert per second (i.e., 3600 alerts per hour), we can handle the busiest ALERTER (e.g., 1707 alerts / hour in our dataset)

Summary

- PillarBox is a general-purpose reliable alert relaying tool, offering intrusion-resilient security in log protection
  - buffering, fast forward-secure logging, secure sequencing
- Features
  - self-protection mode of operation
  - integrity, stealth, persistence





20.1 Networking & Security

Computer networks and security…

Remote Communication

- Networks enable distant interactions

Data Exchange Infrastructure

- Network devices allow the creation of an efficient digital domain

Cyber Attack Vectors

- Networks are common targets needing solid defenses.

There is a dual nature of networks as both enablers and potential risks

Networks

Types

Virtual Circuit

- Legacy phone network
- Single route through sequence of hardware devices established when two nodes start communication
- Data sent along route
- Route maintained until communication ends

Packet Routing

- Internet
- Data split into packets
- Packets transported independently through network
- Each packet handled on a best-effort basis
- Packets may follow different routes

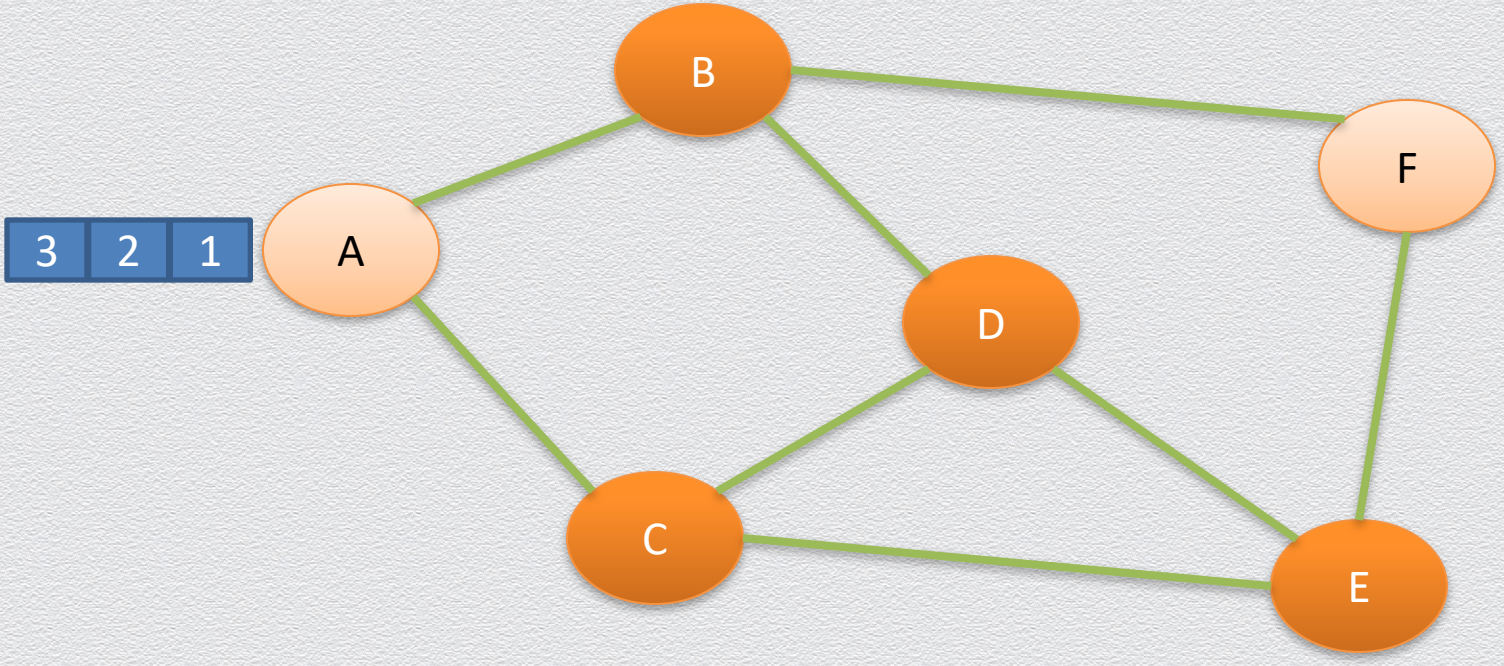
Virtual circuit

- Analog rotary phones

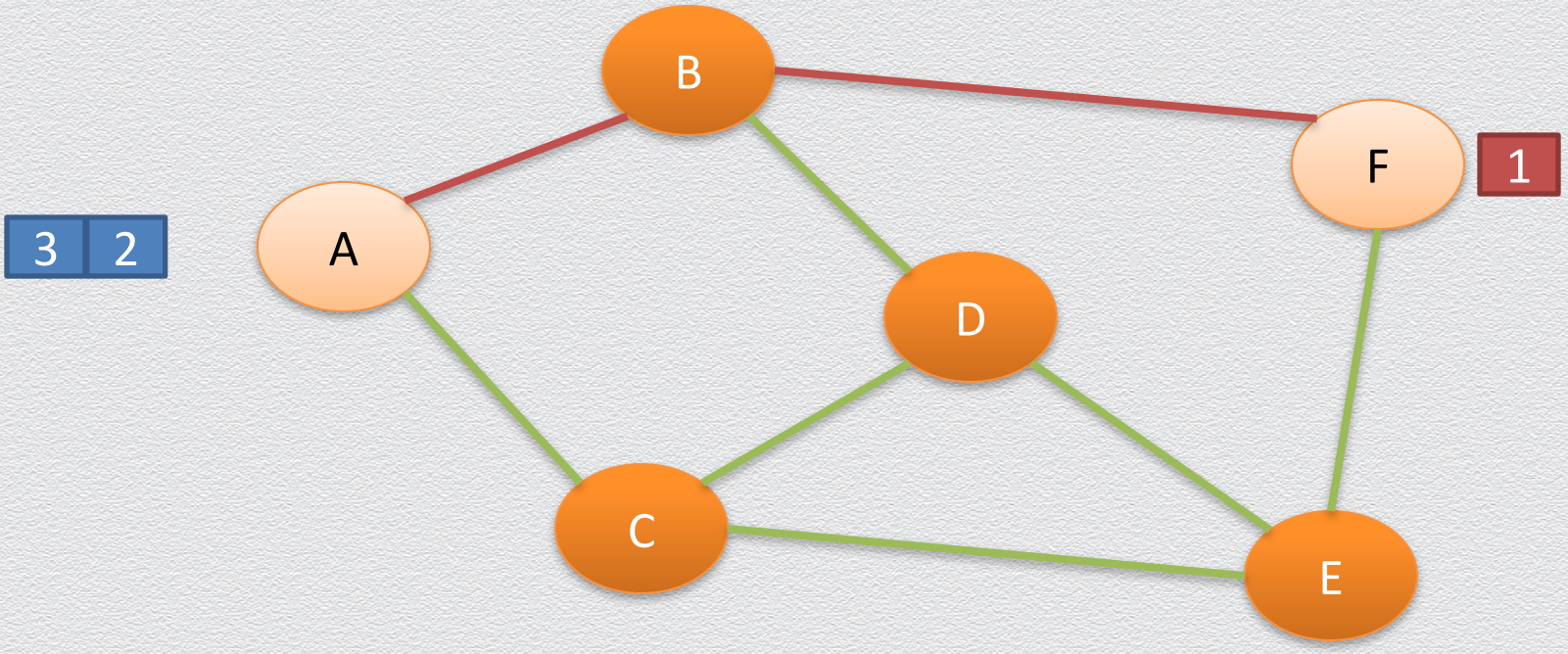
Packet routing (aka switching)

- Data split into packets
- Each packet is
  - Transported independently through network
  - Handled on a best-effort basis by each device
- Packets may
  - Follow different routes between the same endpoints
  - Be dropped by an intermediate device and never delivered

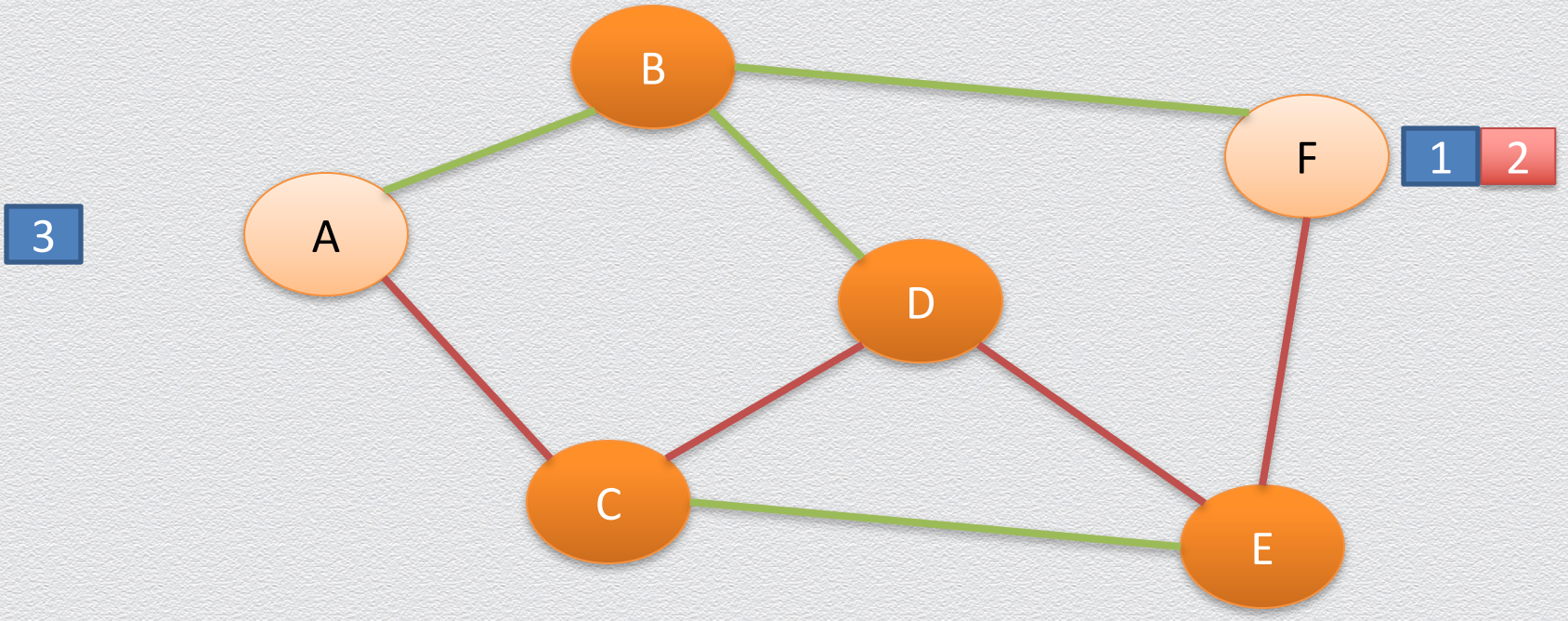
Packet routing: Example



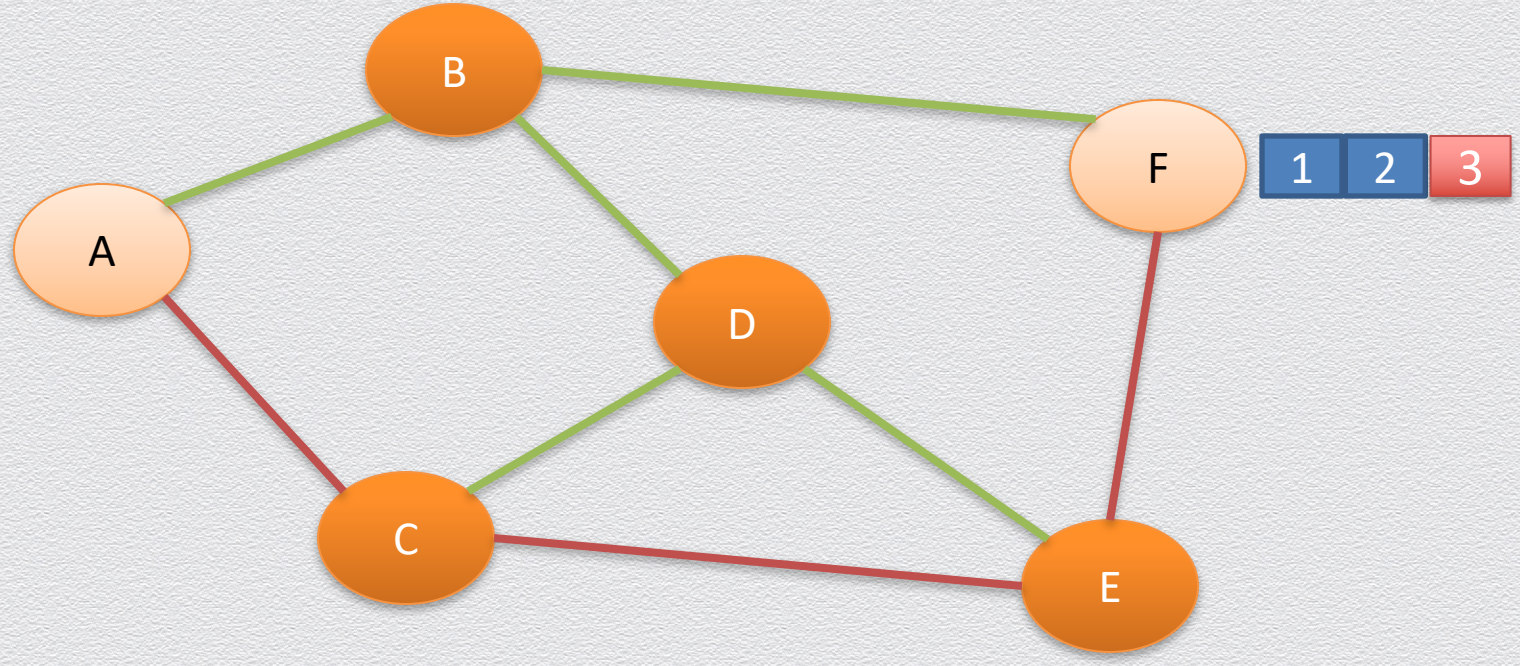
Packet routing: Example



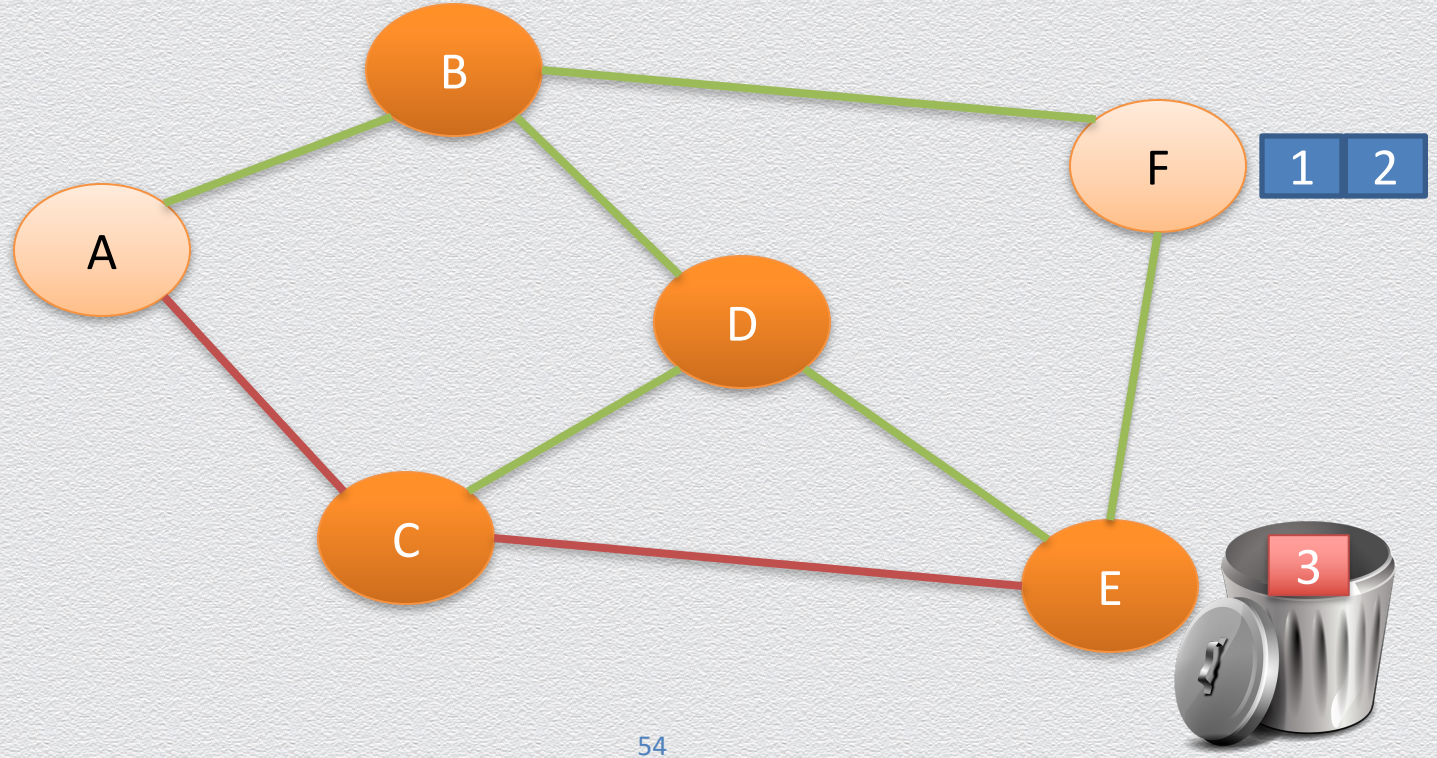
Packet routing: Example



Packet routing: Example



Packet routing: Problem



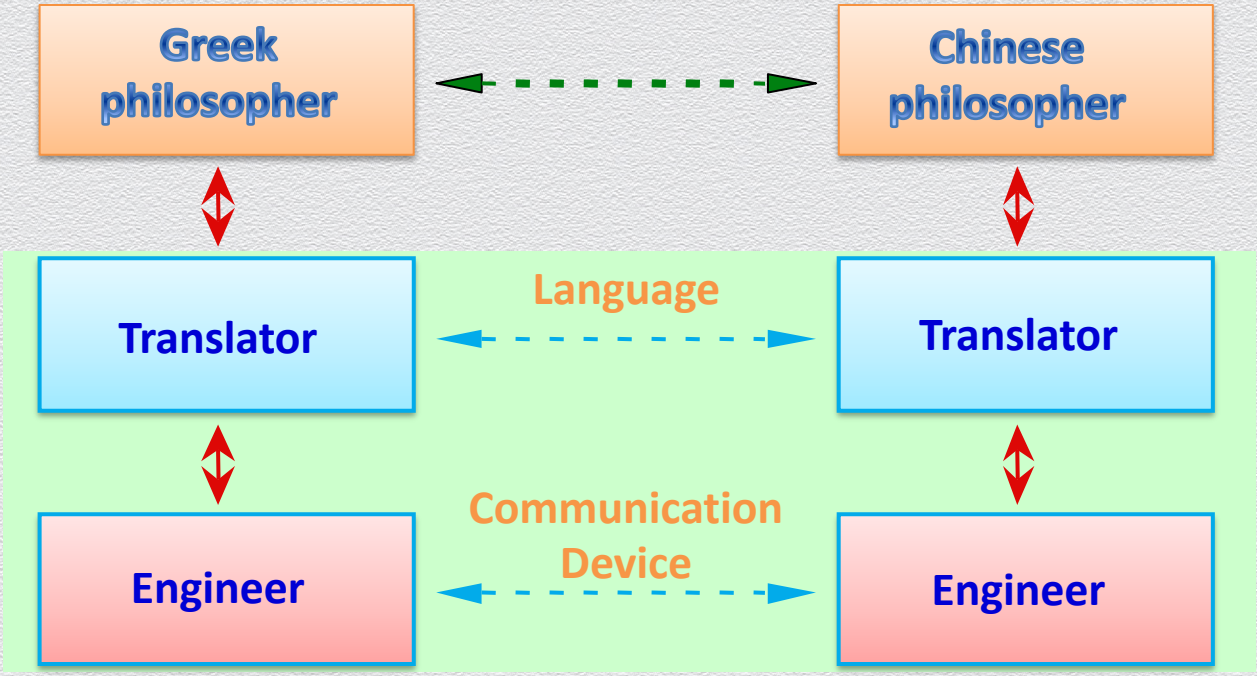
Communication

Characterized by the following fundamental principles

- Packet routing/switching
- Stack of layers (virtual layers)
- Encapsulation

20.2 Protocol layers and encapsulation

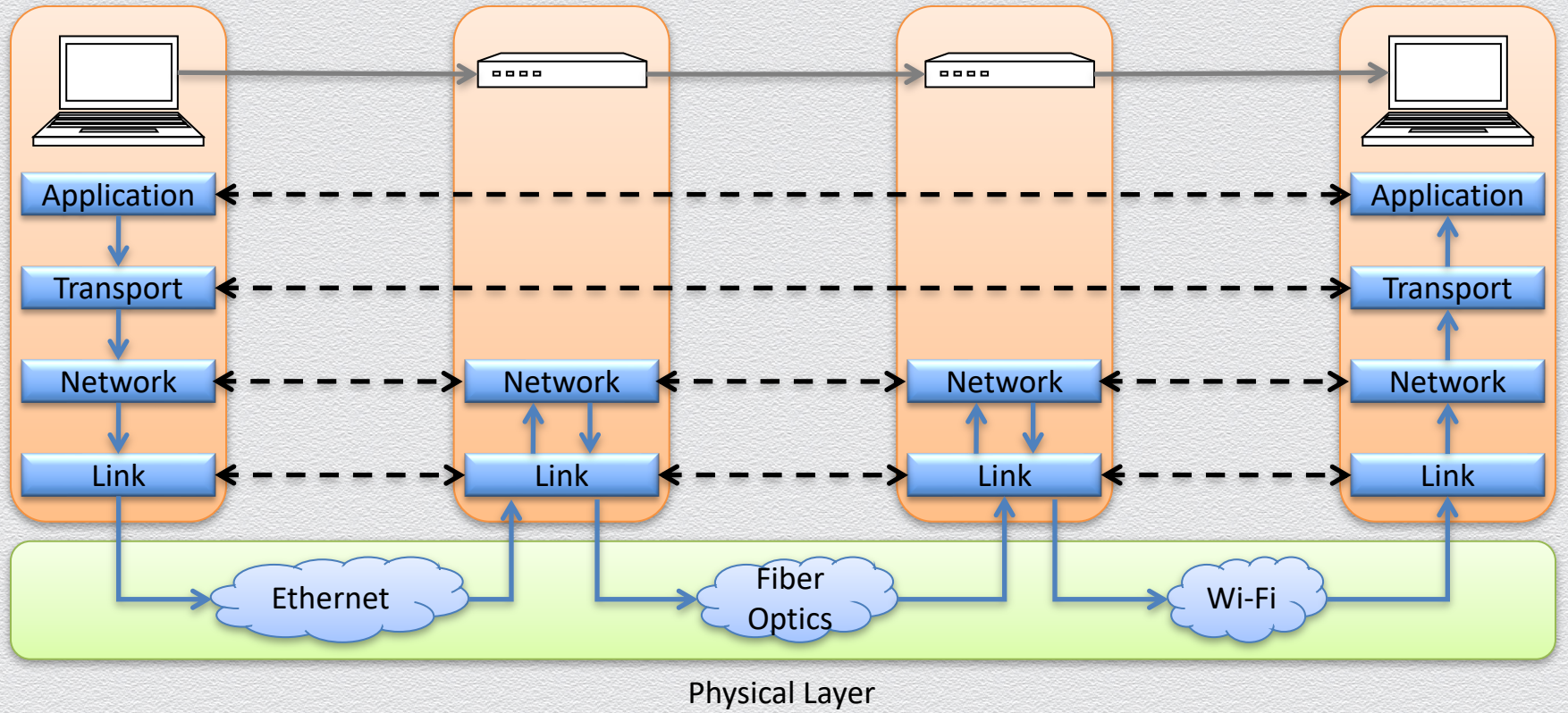
Two philosophers example



Stack of layers

- Network communication models use a **stack of layers**
  - Higher layers use services of lower layers
  - Physical channel at the bottommost layer
- A network device implements several layers
- A communication channel between two devices is established for each layer
  - Actual channel at the bottom layer
  - Virtual channel at higher layers

Internet Layers: How your computer talks to a website



Encapsulation

Packets

- A packet typically consists of
  - Control information
    - header and footer
  - Data
    - Payload

Protocols

- A protocol P uses the services of another protocol Q through **encapsulation**
  - A packet p of protocol P is encapsulated into a packet q of protocol Q
  - The payload of q is p
  - The control information of q is derived from that of p