Advanced Network Security

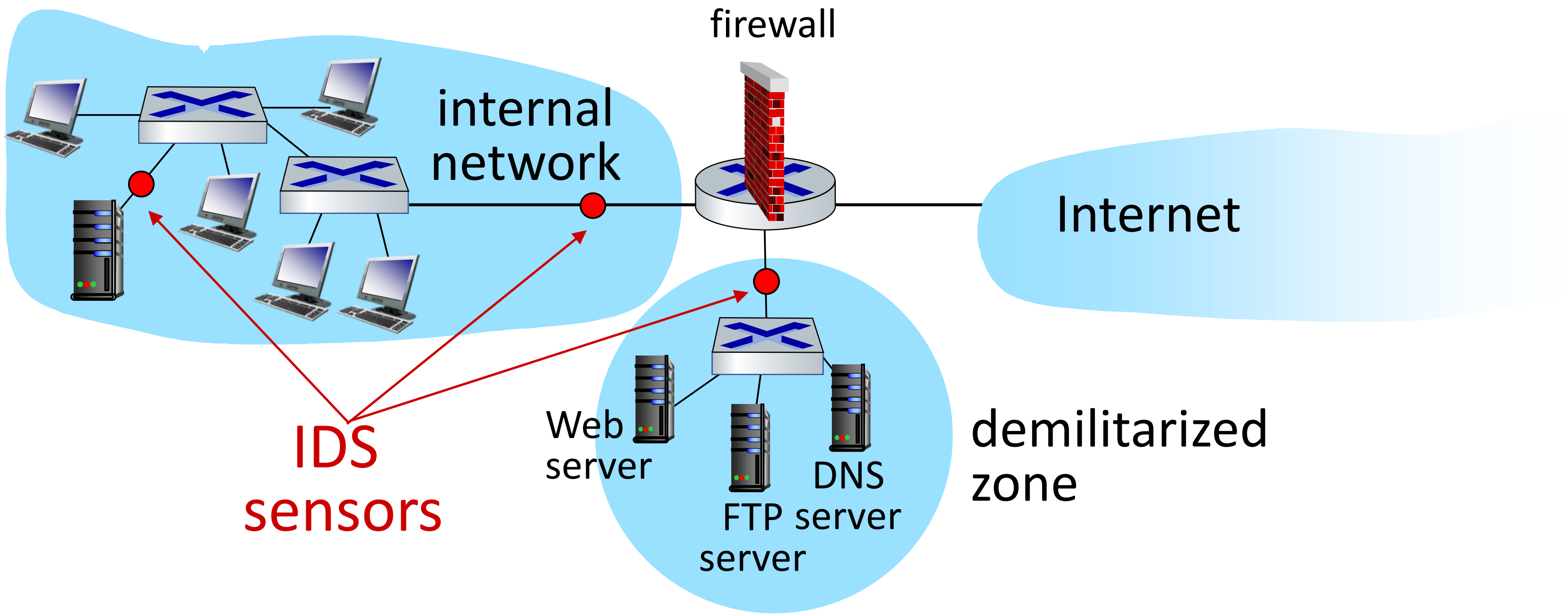
Firewalls and IDS

Amir Mahdi Sadeghzadeh, Ph.D.

Most slides have been adapted from Steven Bellovin, COMS W4180, Columbia University, 2006, and Mehdi Kharrazi, CE40-817, Sharif University of Technology, 2015

Intrusion detection systems

multiple IDSs: different types of checking at different locations



What is IDS?

- An Intrusion Detection System (IDS) is a system that attempts to identify intrusions.
- Intrusion detection is the process of identifying and responding to malicious activity targeted at computing and networking resources.

- The goal of IDS is to detect fingerprints of malicious activity.

Elements of Intrusion Detection

- Primary assumptions:
  - System activities are observable
  - Normal and intrusive activities have distinct evidence
- Components of intrusion detection systems:
  - From an algorithmic perspective:
    - Features - capture intrusion evidence from audit data
    - Models - piece evidence together; infer attack
  - From a system architecture perspective:
    - Audit data processor, knowledge base, decision engine, alarm generation and responses

Where Are IDS Deployed?

- Host-based
  - Monitor activity on a single host
  - Advantage: better visibility into behavior of individual applications running on the host and network traffic
- Network-based (NIDS)
  - Often placed on a **router or firewall**
  - Monitor traffic, examine packet headers and payloads
  - Advantage: single NIDS can protect many hosts and look for global patterns

Requirements of Network IDS

- High-speed, large volume monitoring
  - No packet filter drops
  - Why is it hard?
- Real-time notification
- Broad detection coverage
  - Precision, Recall, F-score
- Economy in resource usage
- Resilience to stress
- Resilience to attacks upon the IDS itself!

Knowledge-based IDS

- Good accuracy, bad completeness
  - Drawback
    - need regular update of knowledge
    - Difficulty of gathering the information
    - Maintenance of the knowledge is a time-consuming task
- Knowledge-based IDS
  - Misuse Detection
  - Specification-based Detection

Misuse Detection

- The system is equipped with a number of attack descriptions ("signature").
  - Then matched against the audit data to detect attacks.
- Signature
  - Sequences of system calls, patterns of network traffic, etc

- Pro: less false positives (But there still some!)
- Con: cannot detect novel attacks, need to update the signatures often.
- Approaches: pattern matching, security rule specification.

Behavior-based IDS

- Good completeness, bad accuracy
- Detect intrusion by observing a deviation from the normal or expected behavior of the system or the users
- Can detect attempts to exploit new and unforeseen vulnerabilities
- Behavior-based IDS
  - Statistics
  - Machine Learning

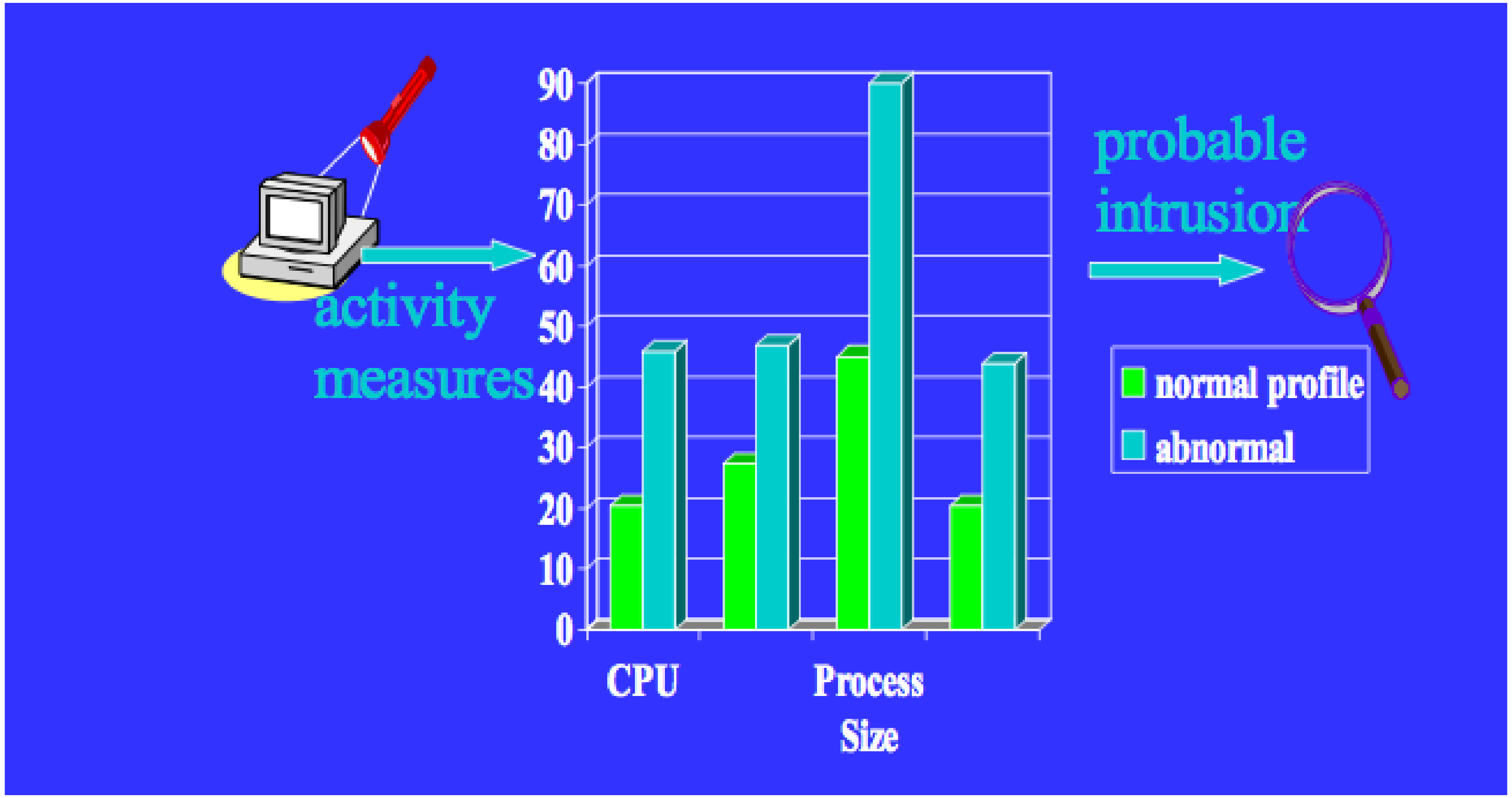
Anomaly Detection

- Using a model of normal system behavior, try to detect deviations and abnormalities
- Any large deviation from the model is thought as anomaly.
  - E.g., raise an alarm when a statistically rare event(s) occurs

- Pro: can detect previous unseen attacks
- Con: have higher false positives, and hard to train a system for a very dynamic environment.
- Approaches: statistical methods, Machine Learning

Anomaly Detection

- Relatively high false positive rate - anomalies can just be new normal activities.



Misuse or Anomaly?

Root pwd modified, admin not logged in

Four failed login attempts

Failed connection attempts on 50 sequential ports

User who usually logs in around 10am from a UT dorm logs in at 4:30am from a Russian IP address

UDP packet to port 1434

“DEBUG” in the body of an SMTP message

Misuse or Anomaly?

Root pwd modified, admin not logged in	Misuse
Four failed login attempts	Anomaly
Failed connection attempts on 50 sequential ports	Anomaly
User who usually logs in around 10am from a UT dorm logs in at 4:30am from a Russian IP address	Anomaly
UDP packet to port 1434	Misuse
“DEBUG” in the body of an SMTP message	Not an attack! (most likely)

Anomaly Detection

- Define a profile describing "normal" behavior
  - Works best for "small", well-defined systems (single program rather than huge multi-user OS)
- Profile may be statistical
  - Build it manually (this is hard)
  - Use machine learning techniques
    - Log system activities for a while, then "train" IDS to recognize normal and abnormal patterns
  - Risk: attacker trains IDS to accept his activity as normal
    - Daily low-volume port scan may train IDS to accept port scans

Host-Based Anomaly Detection

- Compute statistics of certain system activities
  - Login and location frequency; last login; password fails; session elapsed time, output, CPU, I/O; frequency of commands and programs, file read/write/create/delete
- Report an alert if statistics outside range
- Example: IDES (Denning, mid-1980s)
  - For each user, store daily count of certain activities
    - For example, fraction of hours spent reading email
  - Maintain list of counts for several days
  - Report anomaly if count is outside weighted norm
- Problem: most unpredictable user is the most important

Tripwire

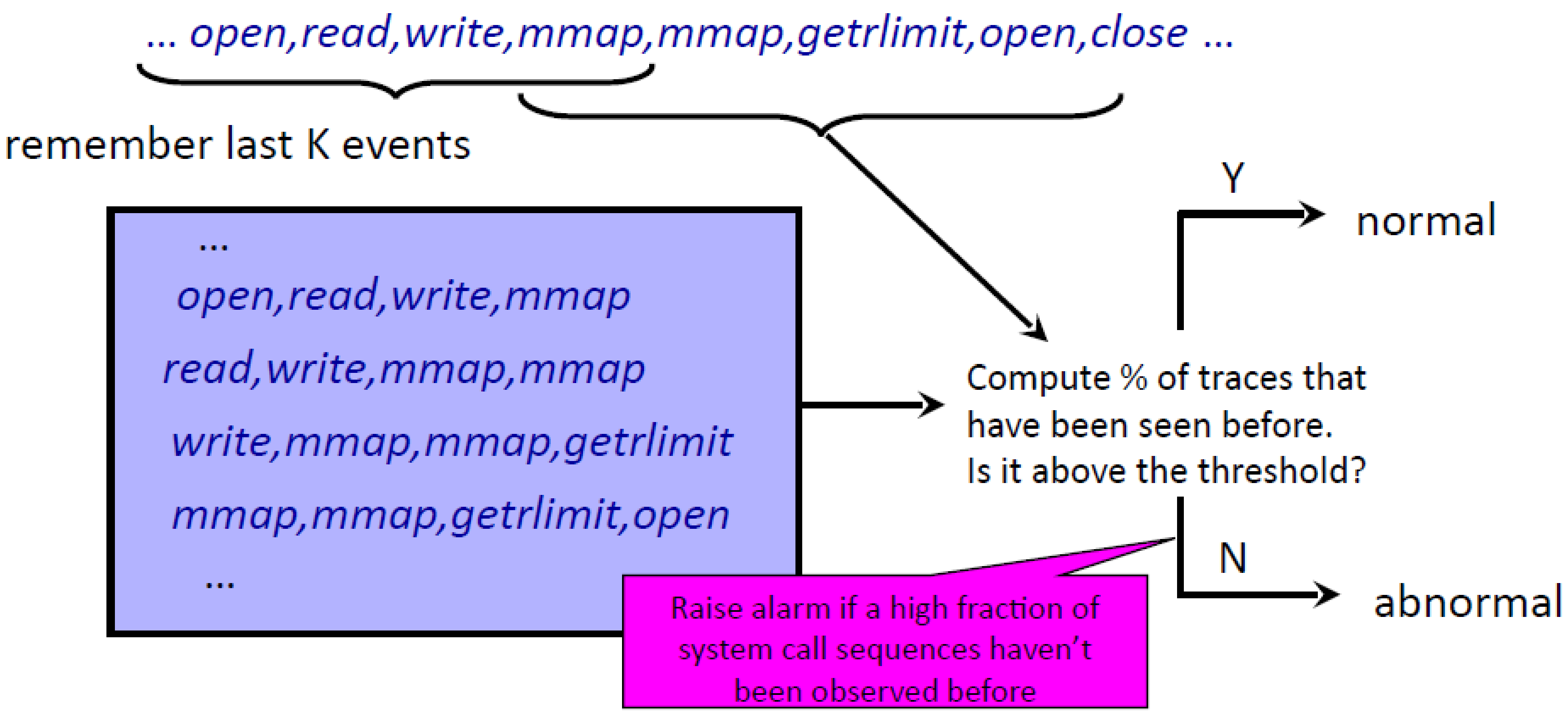
- File integrity checker
  - Records hashes of critical files and binaries
    - Hashes must be stored in read-only memory (why?)
  - Periodically checks that files have not been modified, verifies sizes, dates, permissions
- Good for detecting rootkits, but may be subverted by a clever rootkit
  - Install a backdoor inside a continuously running system process (no changes on disk!)
  - Copy old files back into place before Tripwire runs

System Call Interposition

- Observation: all sensitive system resources are accessed via OS system call interface
  - Files, sockets, etc.
- Idea: monitor all system calls and block those that violate security policy
  - Modify program code to “self-detect” violations
  - Language-level: Java runtime environment inspects the stack of the function attempting to access a sensitive resource and checks whether it is permitted to do so
  - Common OS-level approach: system call wrapper

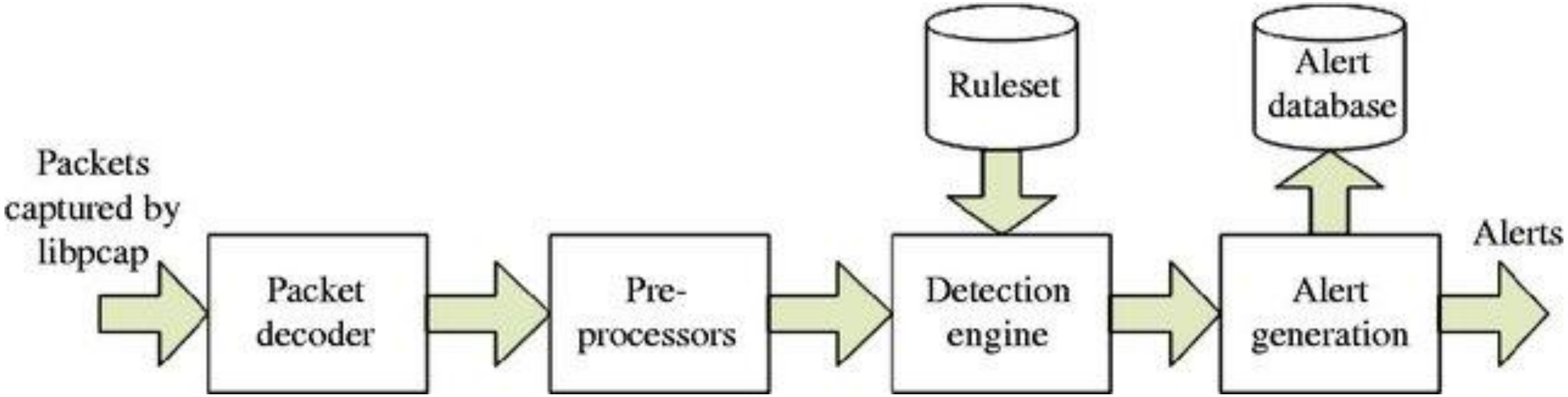
"Self-Immunology" Approach

- Normal profile: short sequences of system calls
  - • Use strace on UNIX



Snort

- Popular open-source network-based intrusion detection tool
- Large, constantly updated sets of rules for common vulnerabilities
- Occasionally had its own vulnerabilities
  - IBM Internet Security Systems Protection Advisory (Feb 19, 2007): Snort IDS and Sourcefire Intrusion Sensor IDS/IPS are vulnerable to a stack-based buffer overflow, which can result in remote code execution



Snort Preprocessors

- Preprocessors are designed to preprocess and normalize traffic, making it ready for detection.
  - HTTP Inspect Preprocessor
  - Portscan Detection Preprocessor
  - SSL/TLS Preprocessor
  - SMTP Preprocessor
  - Stream5 Preprocessor
  - FTP Preprocessor

Snort Rule Examples

- This rule will create an alert if it sees a TCP connection on port 80 (HTTP) with a GET request to the domain “example.com.”

  alert tcp anyany -> any 80 (msg: “Possible HTTP GET request”; content: “GET”; http_method; content: “example.com”; http_host; sid:1000001; rev:1;)

- This rule will create an alert if it sees a TCP connection with a POST request to a web application’s “/login.php” page with a username and password parameter followed by a single quote, a common indicator of a SQL injection attempt.

  alert tcp anyany -> anyany (msg: “Possible SQL Injection attempt”; flow:to_server, established; content: “POST”; nocase; content:”/login.php”; nocase; content: “username=”; nocase; content: “password=”; nocase; content:”‘”; sid:1000003; rev:1;)

https://www.sapphire.net/security/snort-rules-examples/

Port Scanning

- Many vulnerabilities are OS-specific
  - Bugs in specific implementations (specific version), default configuration
- Port scan is often a prelude to an attack
  - Attacker tries many ports on many IP addresses
    - For example, looking for an old version of some daemon with an unpatched buffer overflow
  - If characteristic behavior detected, mount attack
    - Example: SGI IRIX responds on TCPMUX port (TCP port 1); if response detected, IRIX vulnerabilities can used to break in

Scanning Defense

- block traffic from addresses that previously produced too many failed connection attempts
  - Requires maintaining state
  - Can be subverted by slow scanning
- False positives are common, too
  - Website load balancers, stale IP caches
    - E.g., dynamically get an IP address that was used by P2P host

Detecting Attack Strings Is Hard

- Want to detect "USER root" in packet stream
- Scanning for it in every packet is not enough
  - Attacker can split attack string into several packets; this will defeat stateless NIDS
- Recording previous packet's text is not enough
  - Attacker can send packets out of order
- Full reassembly of TCP state is not enough
  - Attacker can use TCP tricks so that certain packets are seen by NIDS but dropped by the receiving application
    - Manipulate checksums, TTL (time-to-live), fragmentation

Anomaly Detection with NIDS

- High false positive rate
  - False identifications are very costly because sys admin will spend many hours examining evidence
- Training is difficult
  - Lack of training data with real attacks
  - Network traffic is very diverse, the definition of "normal" is constantly evolving
    - What is the difference between **a flash crowd** and a **denial of service** attack?

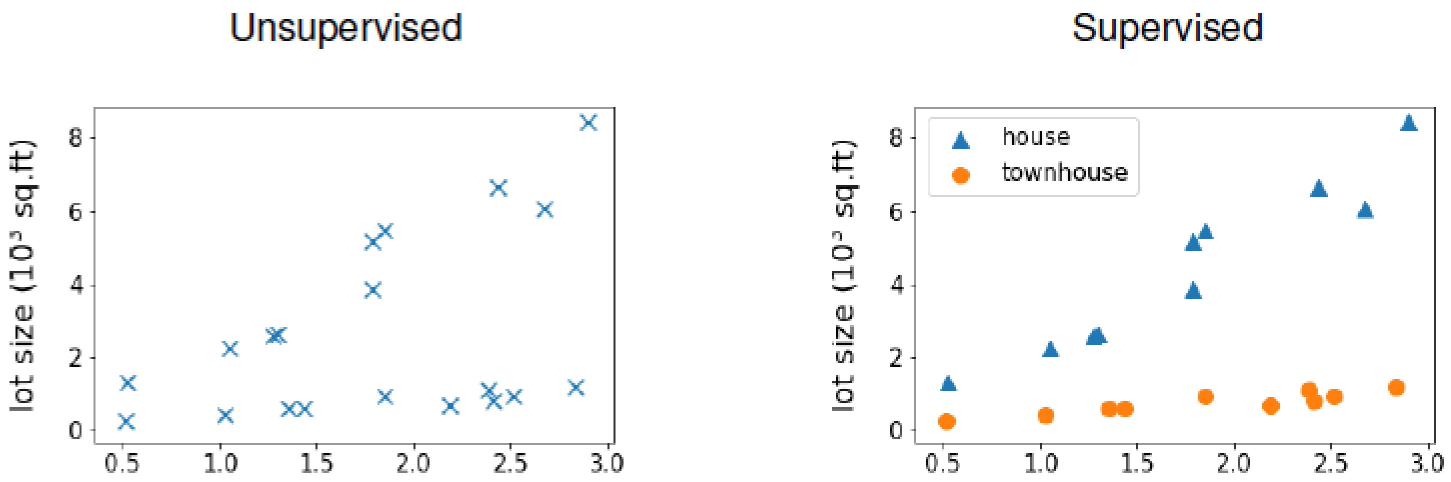
Machine Learning

- A computer program is said to learn from **experience E** with respect to some class of **tasks T** and **performance measure P**, if its performance at tasks in T, as measured by P, improves with experience E.

- Example
  - Task: Classifying emails as spam or not spam
  - Experience: Watching label of emails as spam or not spam
  - Performance metric: The number (or fraction) of emails correctly classified as spam

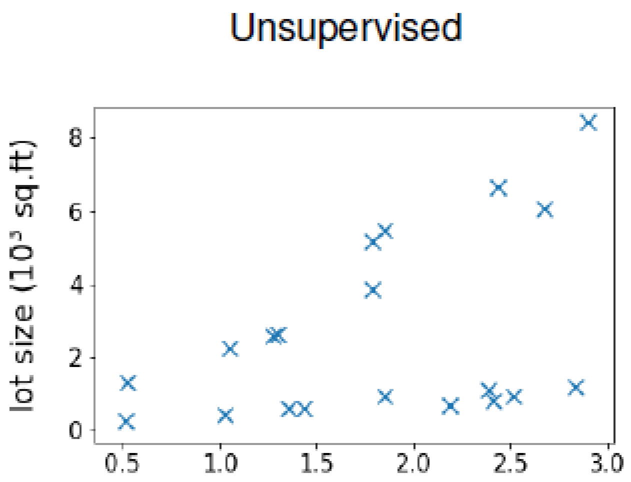
Supervised vs. Unsupervised Learning

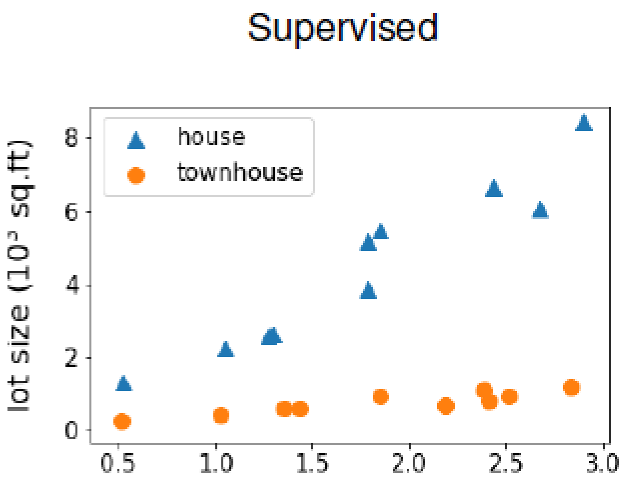
- Supervised learning
  - The model is trained on a labeled dataset, which means it's provided with input-output pairs (features and corresponding target labels).
  - Examples
    - Classification (e.g., spam detection, image recognition), Regression (e.g., price prediction).
  - Anomaly Detection with known anomaly samples



Supervised vs. Unsupervised Learning

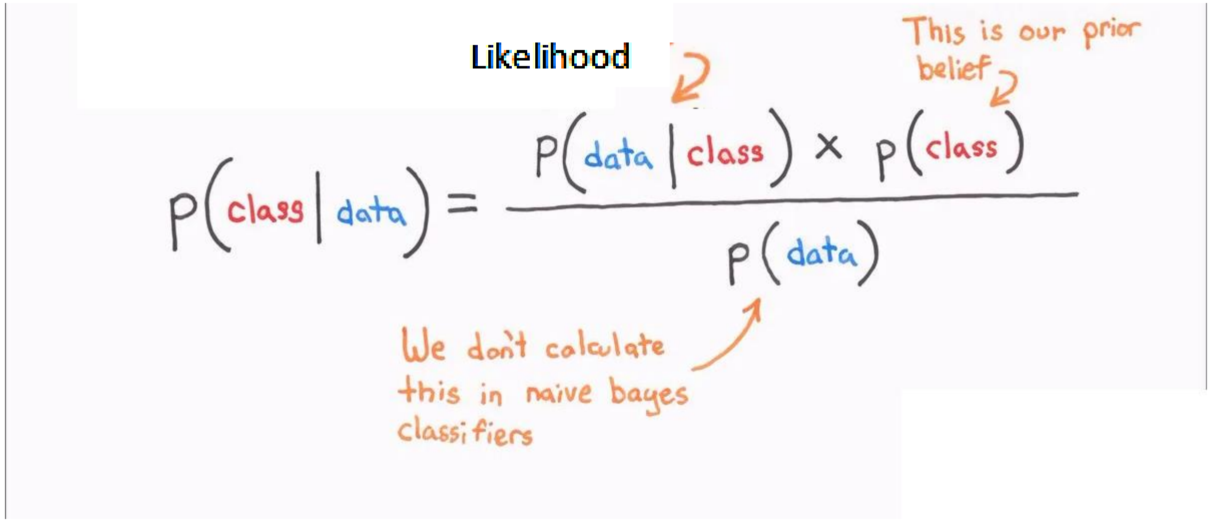
- Unsupervised learning
  - the model is not provided with labeled data. It discovers patterns, structures, or clusters in the data without explicit guidance.
  - Examples
    - Clustering (e.g., customer segmentation), Dimensionality Reduction (e.g., PCA).
  - Anomaly Detection with only normal samples





Naïve Bayes Classifier (supervised learning)

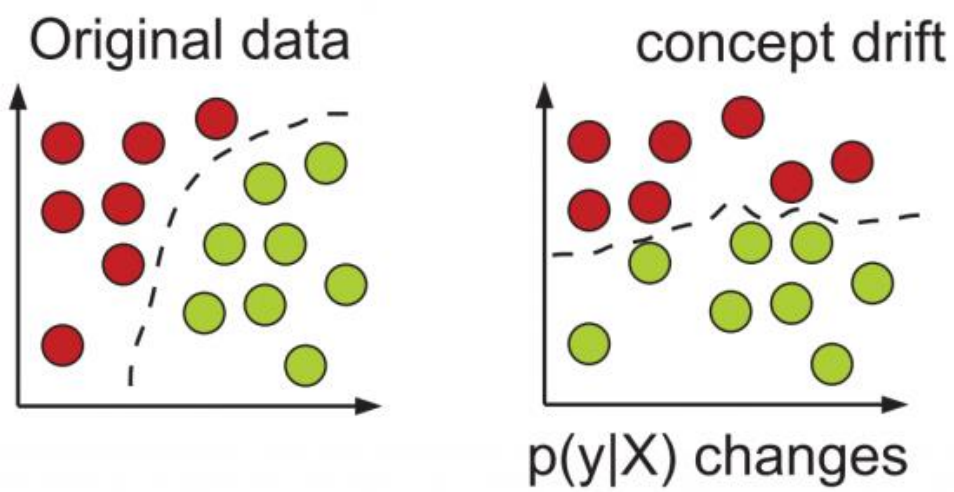
- A Bayes Classifier is a probabilistic model that uses Bayes' theorem to classify data into different categories or classes.



- **P(Class | Data):** Probability of the data belonging to a specific class.
- **P(Data | Class):** Probability of observing the data given the class.
- **P(Class):** Prior probability of the class.
- **P(Data):** Probability of the data.

Concept Drift

- Concept drift occurs when there is a change in the functional relationship between a model's input and output data.
- The model continues to function the same despite the changed context, unaware of the changes.
  - Thus, the patterns it has learned during training are no longer accurate.



Concept Drift

- Concept stands for the joint probability distribution of a Machine Learning model's inputs (X) and outputs (Y).
  - $P(X,Y) = P(Y)\,P(X|Y) = P(X)\,P(Y|X)$
- Concept drift can originate from any of the concept components.

- Why It Occurs
  - Concept drift can occur due to changes in the underlying data distribution, external factors, or evolving user preferences.

IDS Evaluation

- Accuracy: false positives and false negatives should be minimized.
- Performance: the rate at which audit events are processed.
- Completeness: to detect all attacks.
- Fault tolerance: resistance to attacks.
- Timeliness: time elapsed between intrusion and detection.

Intrusion Detection Errors

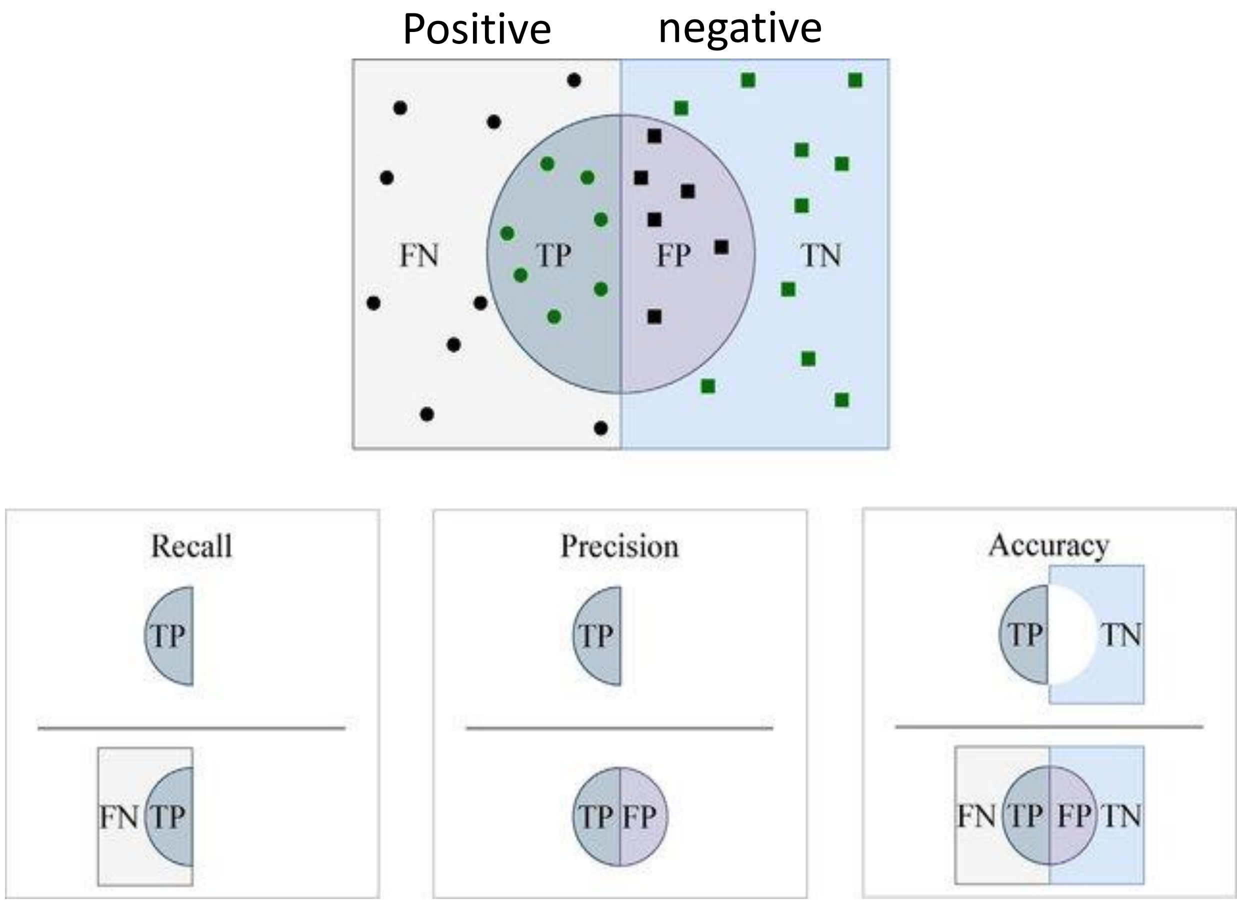
		Actual	
		Positive	Negative
Predicted	Positive	**True Positive**	**False Positive**
	Negative	**False Negative**	**True Negative**

- False negatives: attack is not detected
  - Big problem in signature-based misuse detection
- False positives: harmless behavior is classified as an attack
  - Big problem in statistical anomaly detection

- All intrusion detection systems (IDS) suffer from errors of both types
- Which is a bigger problem?
  - Attacks are fairly rare events, thus IDS often suffer from the base-rate fallacy

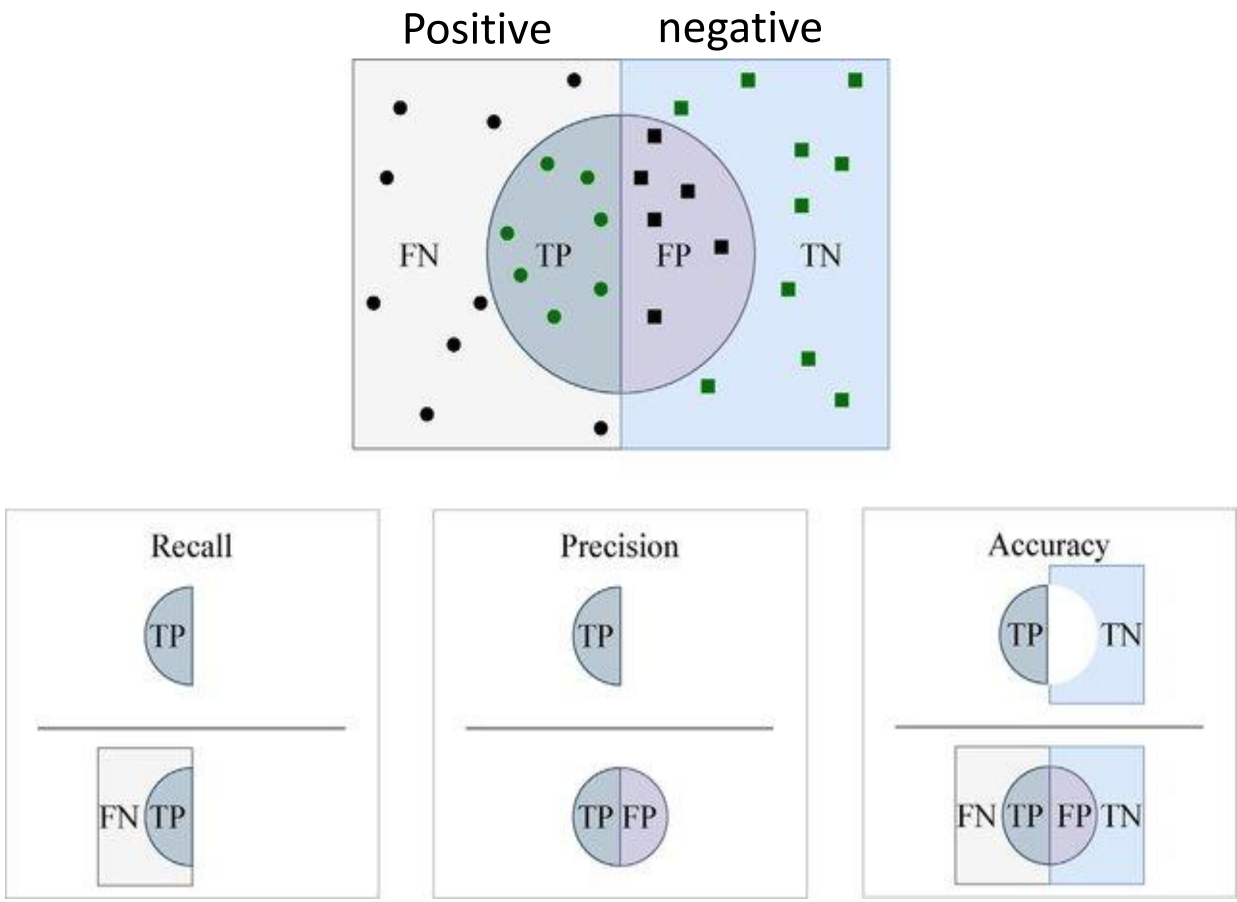
Precision and Recall

- Accuracy
  - Formula: A = (TP + TN) / (TP + TN + FP + FN)
  - Interpretation: High accuracy indicates the overall effectiveness of the IDS in correctly identifying both intrusions and non-intrusions.



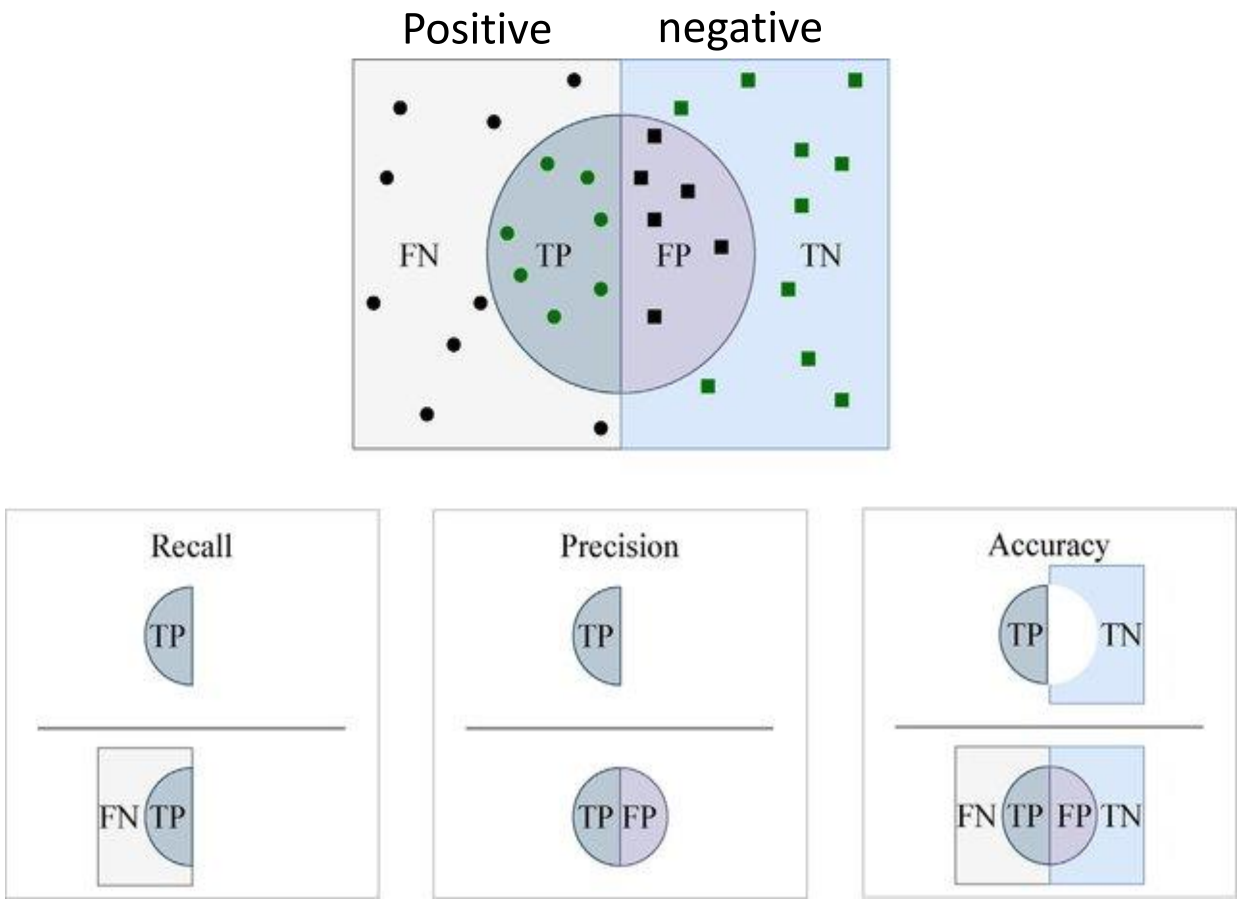
Precision and Recall

- Recall:
  - Formula: R = TP / (TP + FN)
  - Interpretation: High recall means the IDS effectively detects a large portion of intrusions, reducing the chances of false negatives (missed intrusions).



Precision and Recall

- Precision:
  - Formula: P = TP / (TP + FP)
  - Interpretation: High precision indicates that when the IDS raises an alert, it's likely a true intrusion, minimizing false alarms.



Conditional Probability

- Suppose two events A and B occur with probability Pr(A) and Pr(B), respectively
  - Let Pr(A,B) be probability that both A and B occur
- What is the conditional probability that A occurs assuming B has occurred?

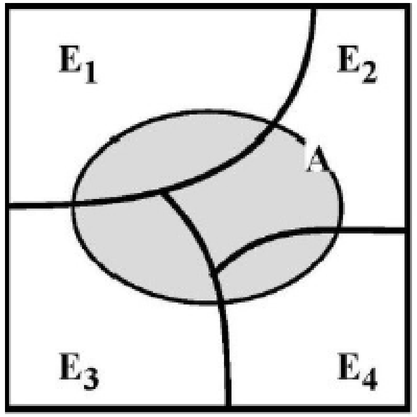
$$\mathrm{Pr}(A|B) = \frac{Pr(A, B)}{\mathrm{Pr}(B)}$$

Bayes' Theorem

Suppose mutually exclusive events $E_1, \ldots ,E_n$ together cover the entire set of possibilities

Then the probability of <u>any</u> event A occurring is

$$Pr(A) = \sum_{1 \le i \le n} Pr(A \mid E_i) \bullet Pr(E_i)$$



- Intuition: since $E_1, \ldots ,E_n$ cover the entire probability space, whenever A occurs, some event E_i must have occurred

Can rewrite this formula as

$$Pr(E_i \mid A) = \frac{Pr(A \mid E_i) \bullet Pr(E_i)}{Pr(A)}$$

Base-Rate Fallacy

- 1% of traffic is SYN floods; IDS accuracy is 90%
  - IDS classifies a SYN flood as attack with prob. 90%, classifies a valid connection as attack with prob. 10%
- What is the probability that the connection flagged as a SYN flood by IDS is actually valid?

$$\Pr(\text{valid} \mid \text{alarm}) = \frac{\Pr(\text{alarm} \mid \text{valid}) \bullet \Pr(\text{valid})}{\Pr(\text{alarm})}$$

$$= \frac{\Pr(\text{alarm} \mid \text{valid}) \bullet \Pr(\text{valid})}{\Pr(\text{alarm} \mid \text{valid}) \bullet \Pr(\text{valid}) + \Pr(\text{alarm} \mid \text{SYN flood}) \bullet \Pr(\text{SYN flood})}$$

$$= \frac{0.10 \bullet 0.99}{0.10 \bullet 0.99 + 0.90 \bullet 0.01}$$

= 92% chance raised alarm is false!!!

slide

References

- Kurose, James F., and Keith W. Ross. "Computer networking: A top-down approach edition." Addision Wesley (2007), chapter 8.
- Steven Bellovin, COMS W4180, Columbia University, 2006
- Mehdi Kharrazi, CE40-817, Sharif University of Technology, 2015
- Vitaly Shmatikov, CS 361S, UT Austin, 2014
- Amir Mahdi Sadeghzadeh, Security and Privacy in Machine Learning, Sharif University of Technology, 2023