Kubernetes

Contents

- Why do we need Kubernetes?
- What is Kubernetes?
- Features of Kubernetes
- How Kubernetes Works?
- Architecture of Kubernetes
- Deploy Application to Kubernetes

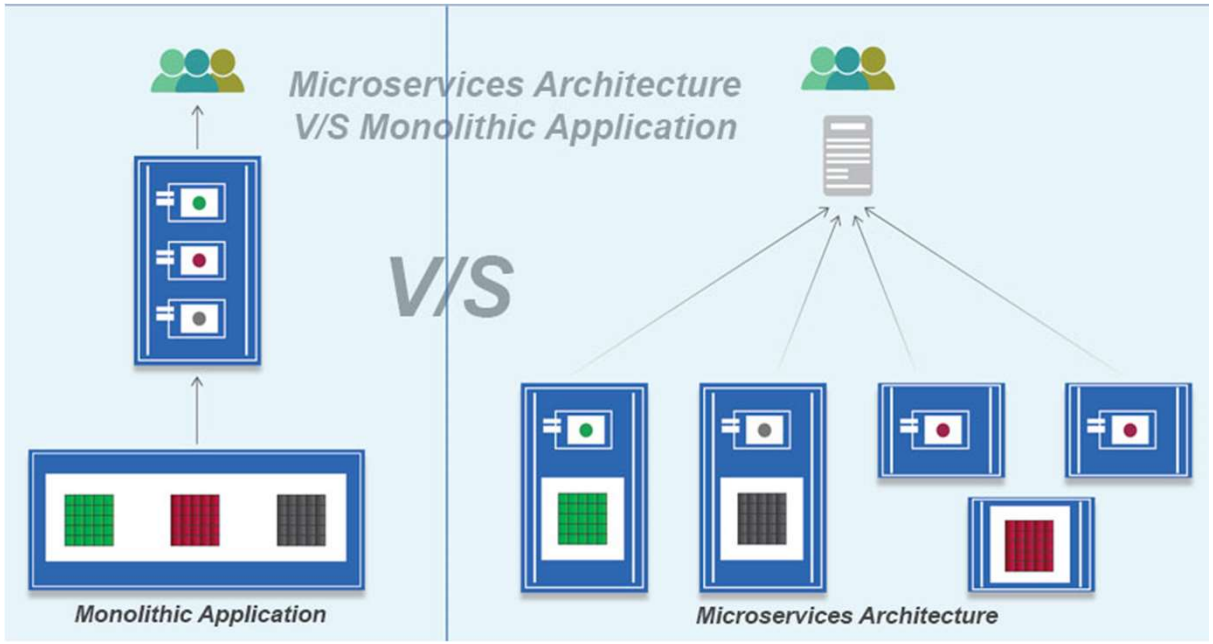
Monolithic and Microservice Application

Monolithic Software Application: all the features of applications are coded into a one code-base.

Limitations: dependency, making changes are difficult.

Micro-service Software Application: all the applications have different code-base and should be deployed separately.

Limitations: low resource utilization



Docker

Docker is an open source containerization platform. Docker enables developers to package applications into containers—standardized executable components that combine application source code with all the **operating system** (OS) libraries and dependencies required to run the code in any environment.

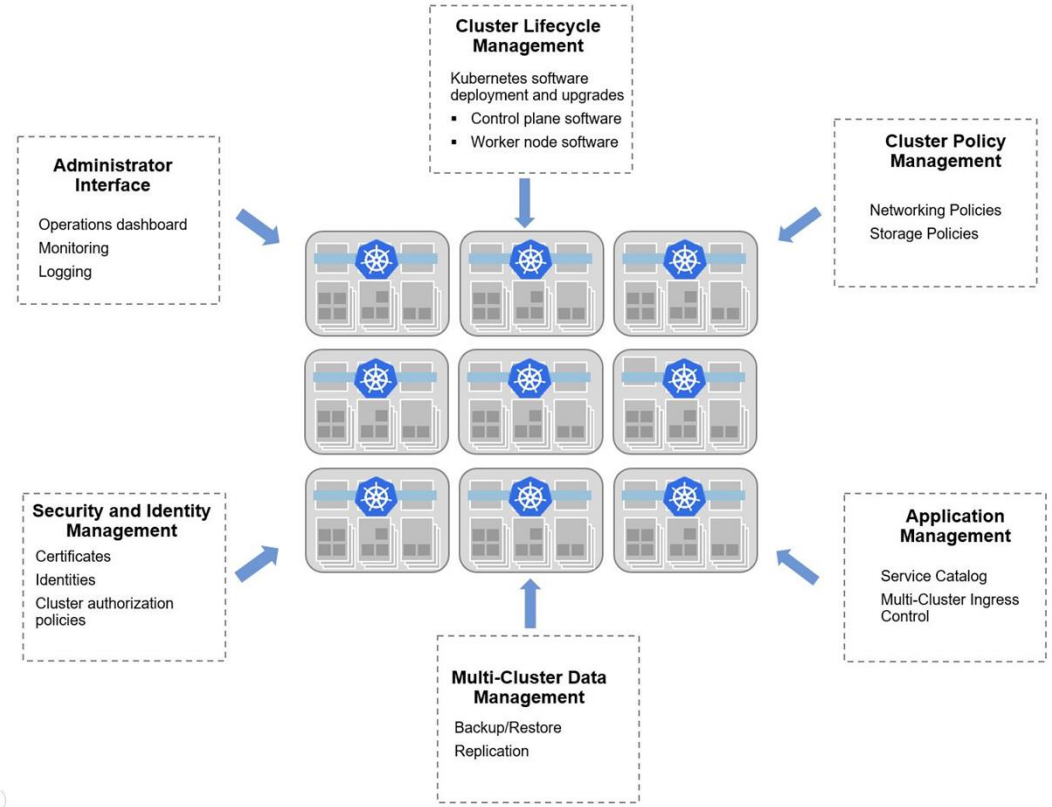
Why Kubernetes

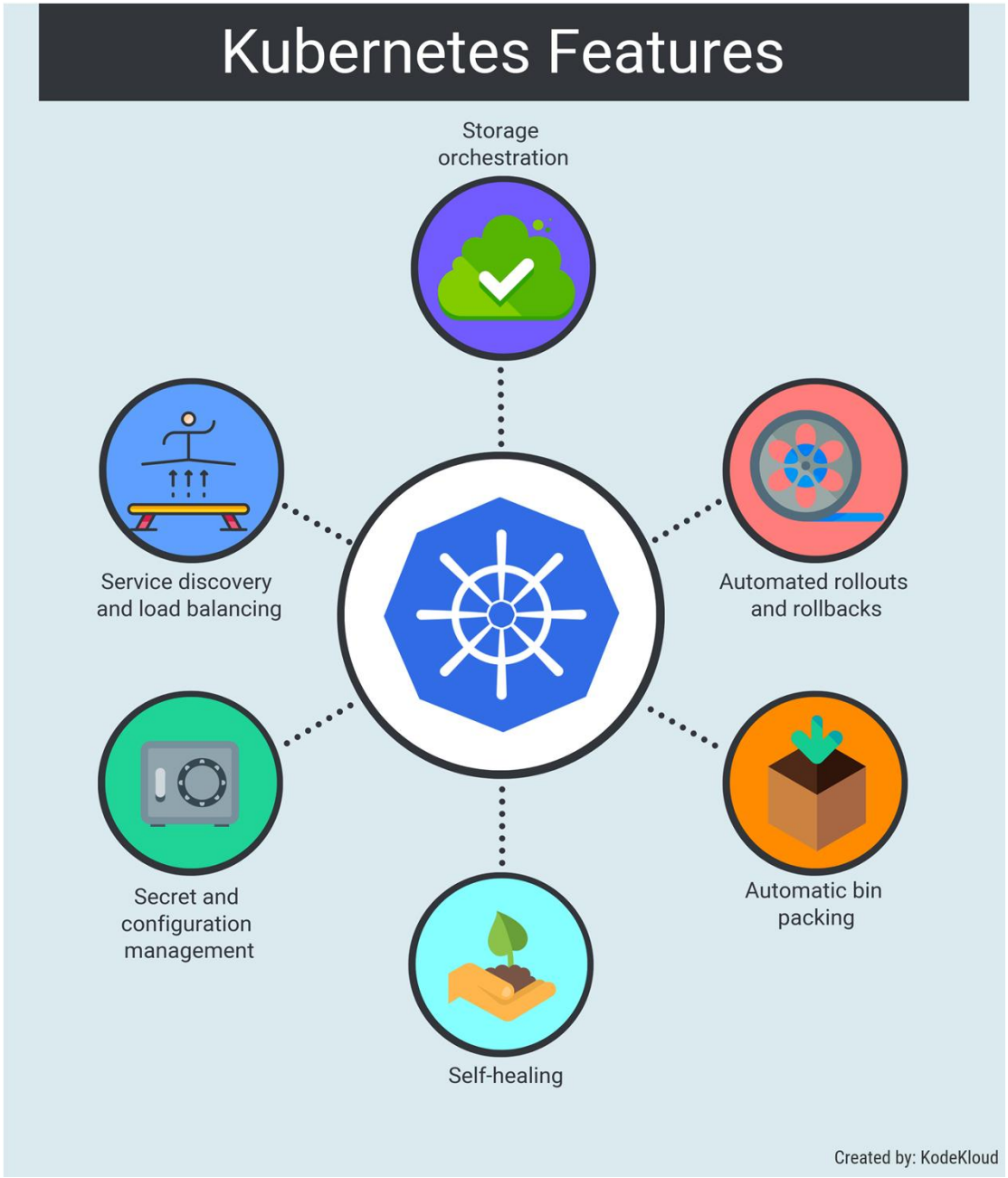
- Each service's monitoring is difficult
- Scaling a particular service based on load is not possible.
- Too much manual intervention is required for managing containers.
- Managing containers on multiple servers become difficult.

- **Kubernetes** is an orchestration tool for containerized applications. Starting with a collection of Docker containers, **Kubernetes** can control resource allocation and traffic management for cloud applications and microservices.

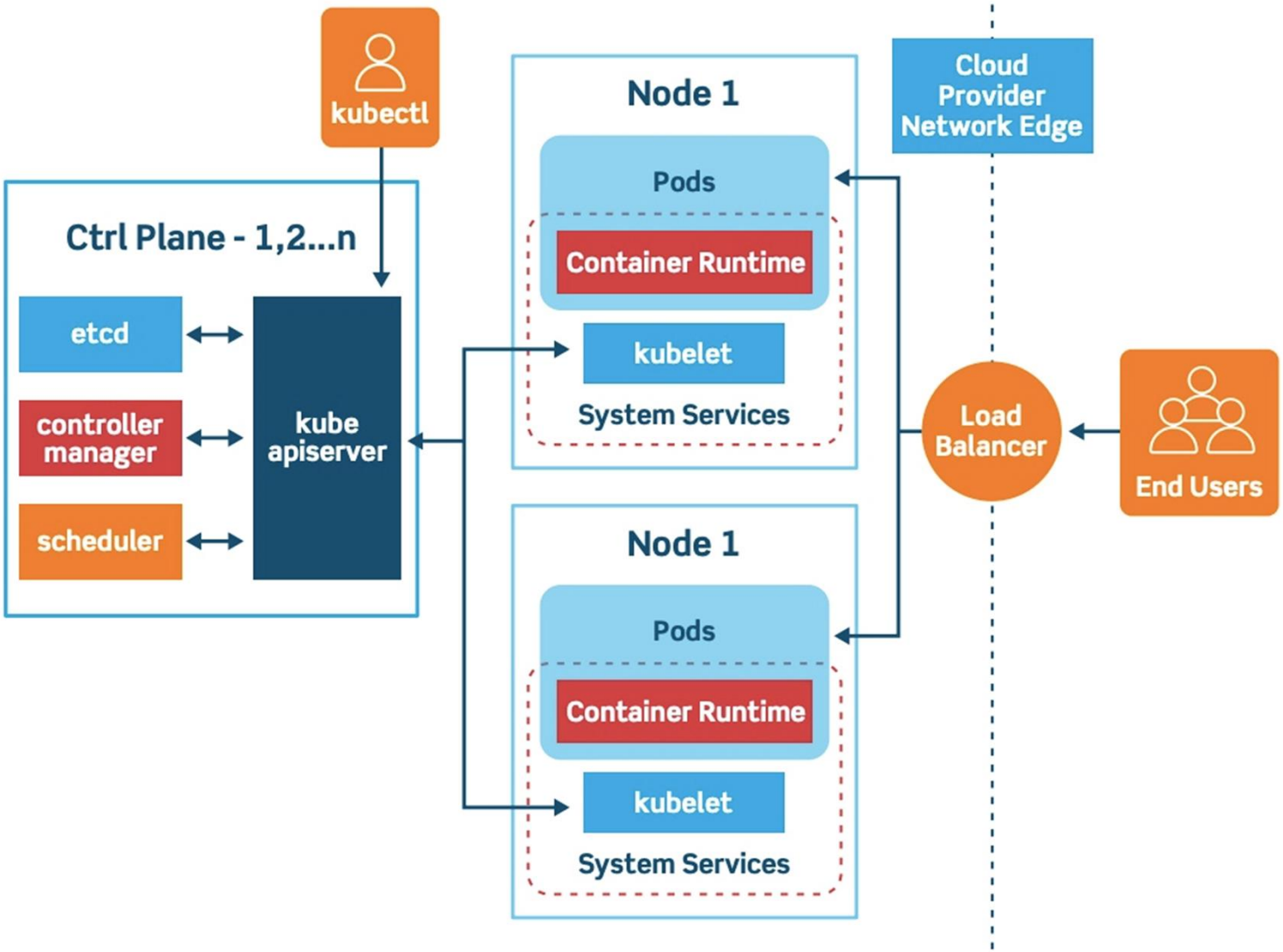
What is Kubernetes

- Kubernetes being a container orchestration tool, is used when our application is distributed in multiple containers. Its job is to monitor, scale, restart containers automatically even if they are spread across multiple nodes.




Kubernetes Features
Storage orchestration
Service discovery and load balancing
Automated rollouts and rollbacks
Secret and configuration management
Automatic bin packing
Self-healing
Created by: KodeKloud

Kubernetes Architecture



Kubernetes Intsllation

1. **Kubeadm**: bare-metal installation
2. **Minikube**: virtualized environment for kubernetes
3. **Kops**: Kubernetes on AWS
4. **Kubernetes on GCP**: Kubernetes running on google cloud platform.

What is EKS?

- Amazon Elastic Kubernetes Service (Amazon **EKS**) is a managed service that you can use to run Kubernetes on AWS without needing to install, operate, and maintain your own Kubernetes control plane or nodes.
- Runs and scales the Kubernetes control plane across multiple AWS Availability Zones to ensure high availability.

Amazon EKS

- IAM -> VPC -> Security Groups

Introduction to Big Data

Module 5

Data vs Information ??

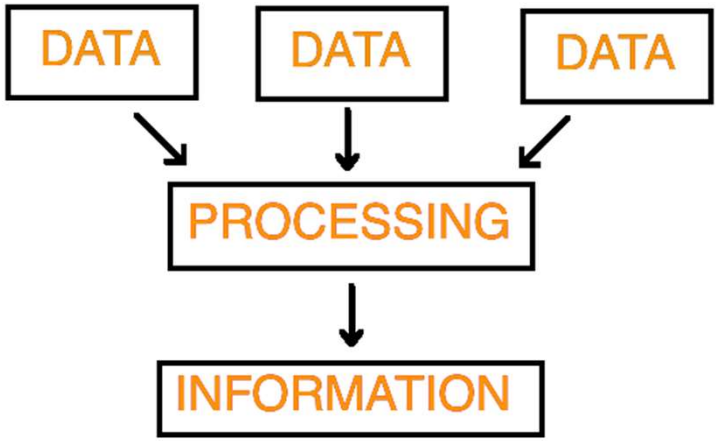
Data vs. Information

Data

- raw facts
- no context
- just numbers and text

Information

- data with context
- processed data
- value-added to data
  - summarized
  - organized
  - analyzed



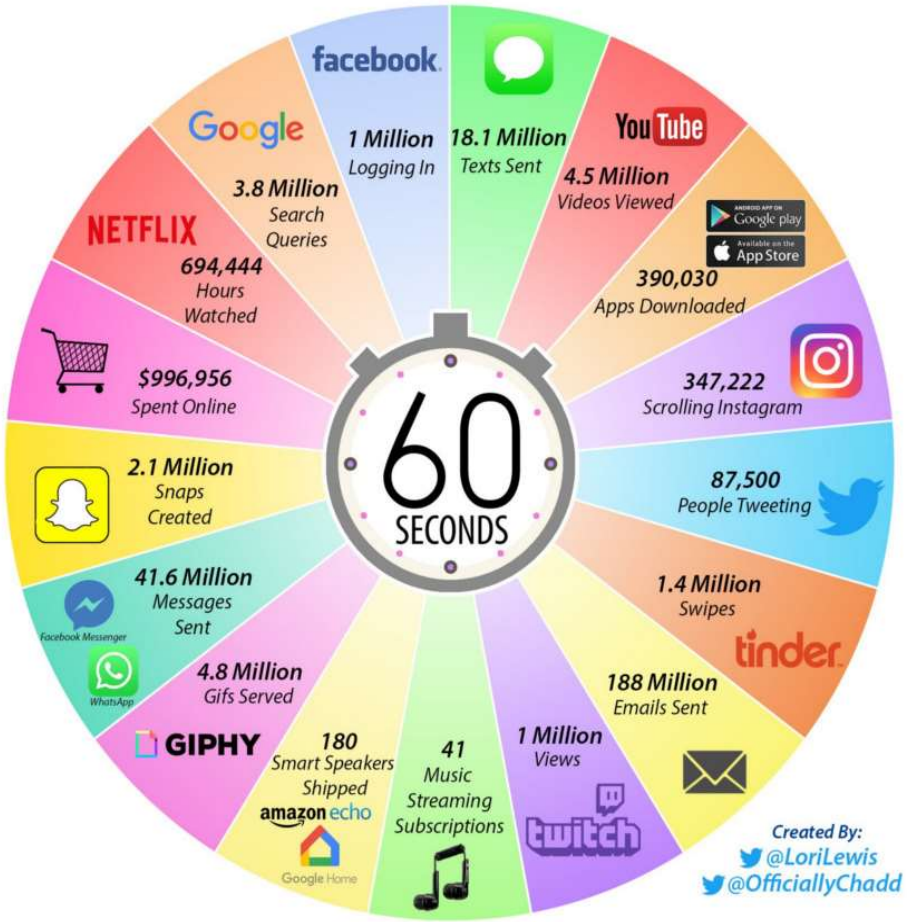
Courtesy: [1], [2]

Examples of Data and Information

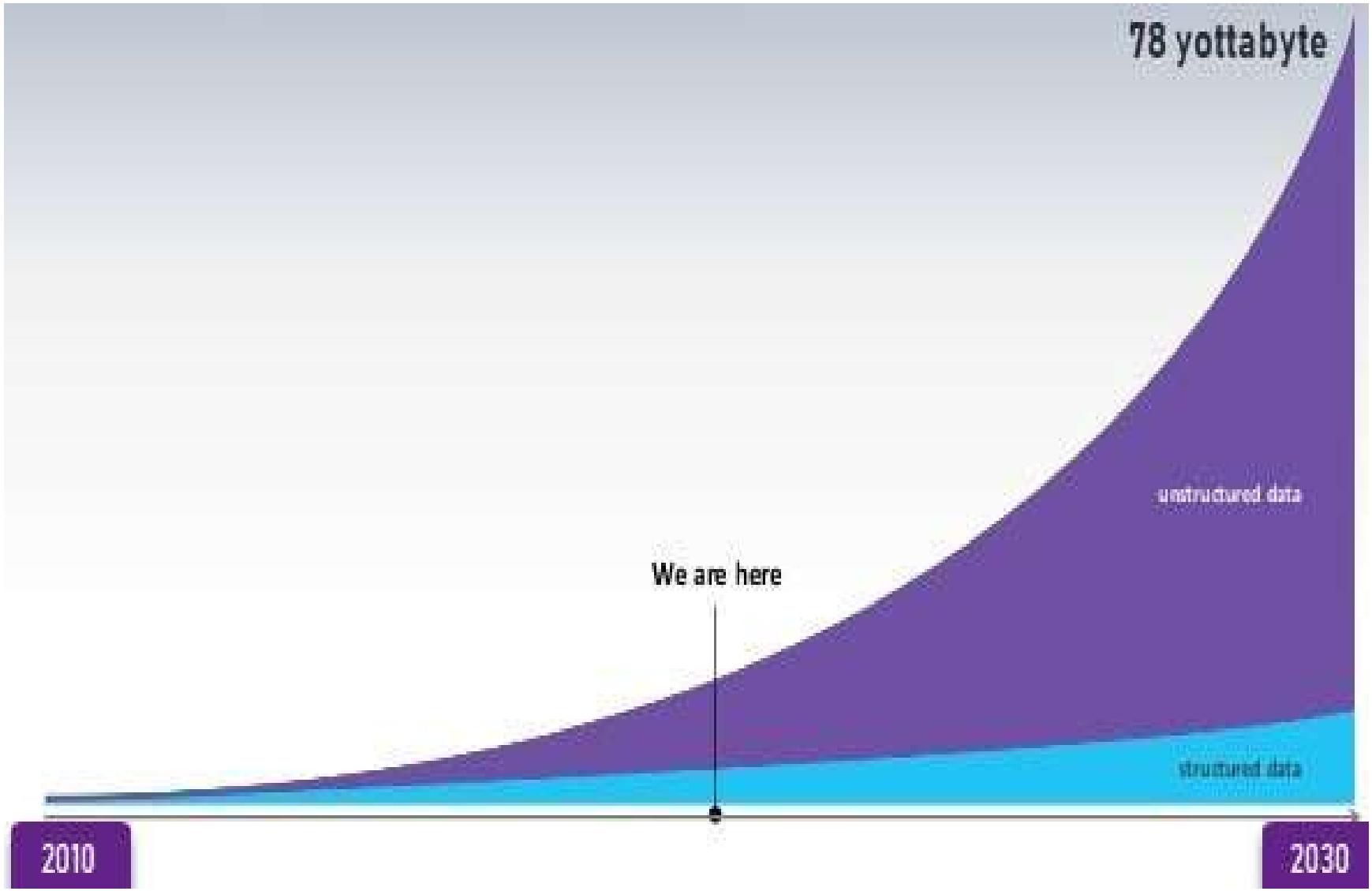
- The history of temperature readings all over the world for the past 100 years is data. If this data is organized and analyzed to find that global temperature is rising, then that is information.
- The number of visitors to a website by country is an example of data. Finding out that traffic from the U.S. is increasing while that from Australia is decreasing is meaningful information.

How much data do we generate?

- **Structured Data**
  - Relational Databases
  - Well defined schema
- **Unstructured Data**
  - Videos, audio, images etc.
- **Semi-structured Data**
  - structured in form but not well defined (no schema)
  - XML files



The Rise of Unstructured Data



How large your data is?

- What is the maximum file size you have dealt so far?
  - Movies/files/streaming video that you have used?
- What is the maximum download speed you get?
  - To retrieve data stored in distant locations?
- How fast your computation is?
  - Time to transfer, process and get result?

Memory unit	Size	Binary size
kilobyte (kB/KB)	10^3	2^{10}
megabyte (MB)	10^6	2^{20}
gigabyte (GB)	10^9	2^{30}
terabyte (TB)	10^{12}	2^{40}
petabyte (PB)	10^{15}	2^{50}
exabyte (EB)	10^{18}	2^{60}
zettabyte (ZB)	10^{21}	2^{70}
yottabyte (YB)	10^{24}	2^{80}

Courtesy: [3]

Sources of Data

- "Every day, we create 2.5 quintillion (10^{18}) bytes of data
  - 90% of the data in the world today has been created in the last two years alone.
  - The data come from several sources
    - *sensors used to gather climate information*
    - *posts to social media sites,*
    - *digital pictures and videos*
    - *purchase transaction records*
    - *cell phone GPS signals*

      etc.

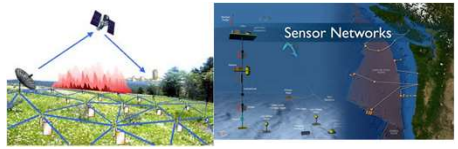
Courtesy: [3]

Examples

Social media and networks
(All of us are generating data)

Scientific instruments
(Collecting all sorts of data)

Mobile devices
(Tracking all objects all the time)



Sensor technology and networks
(Measuring all kinds of data)

Courtesy: [3]

Now Data is Big data!

- No single standard definition!
- 'Big-data' is similar to 'Small-data', but bigger

  …but having data bigger consequently requires different approaches

    - techniques, tools and architectures

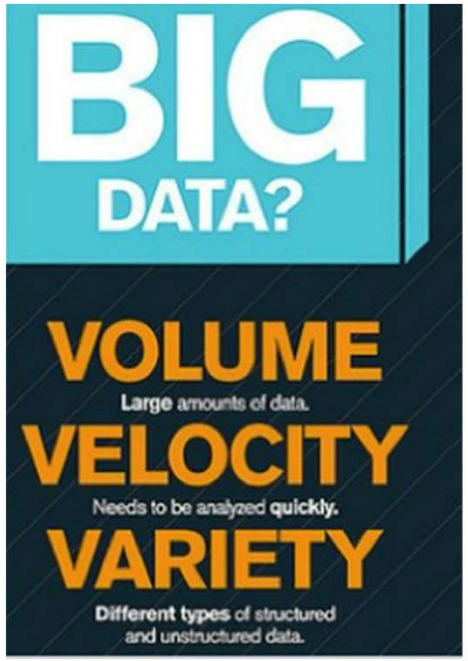
  …to solve: new problems

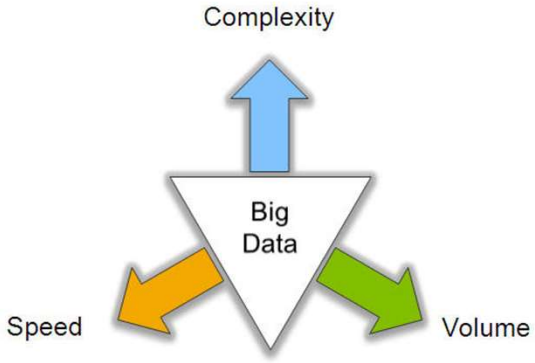
    …and, of course, in a better way

Big data is data whose scale, diversity, and complexity require new architecture, techniques, algorithms, and **analytics** to manage it and extract value and hidden knowledge from it.

Courtesy: [3]

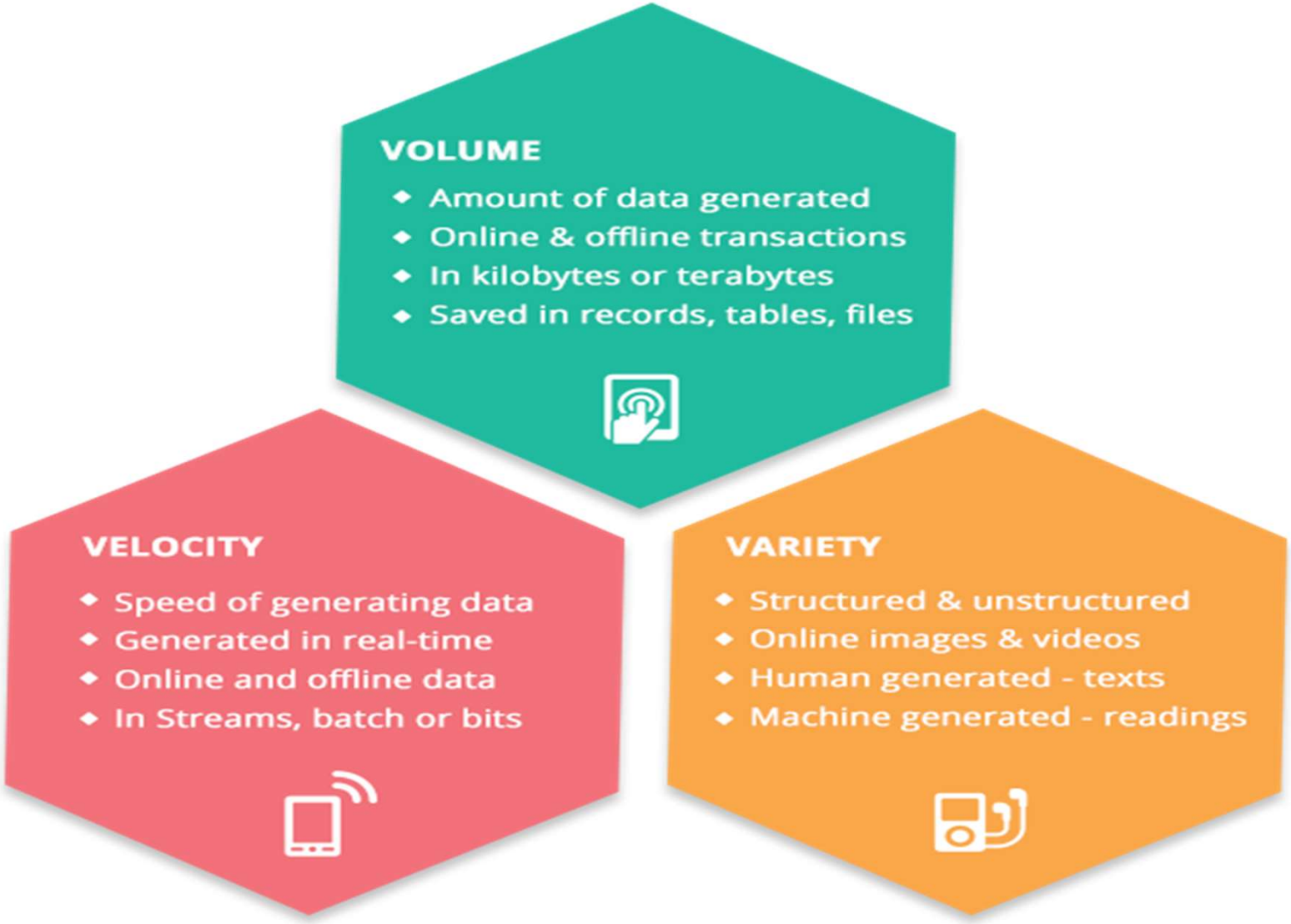
Characteristics of Big data: **V3**





Courtesy: [3]

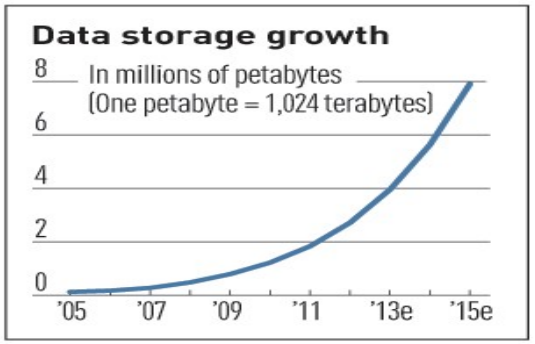
THE 3Vs OF BIG DATA

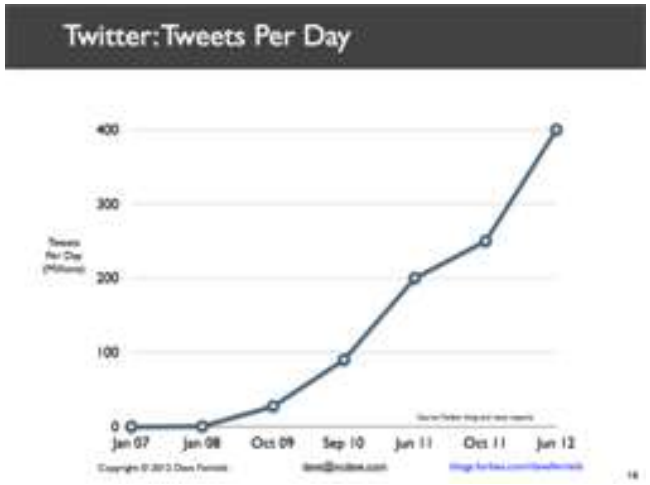


www.whishworks.com

V3 : V for Volume

- Large volume of data, More computation
  - More tools and techniques required
- needs to be processed rapidly
  - More storage capacity



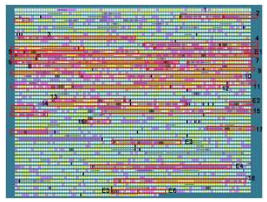


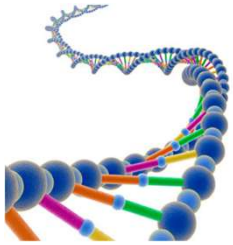
Exponential increase in collected/generated data

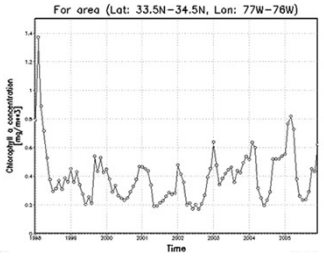
Courtesy: [3]

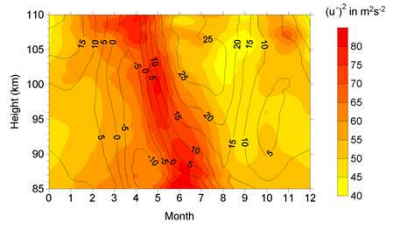
V3: V for Variety

- Various formats, types, and structures
  - Text, numerical, images, audio, video, sequences, time series, social media data, multi-dimensional arrays, etc…
- Static data vs. streaming data
- A single application can be generating/collecting many types of data









To extract knowledge➔ all these types of data need to be linked together

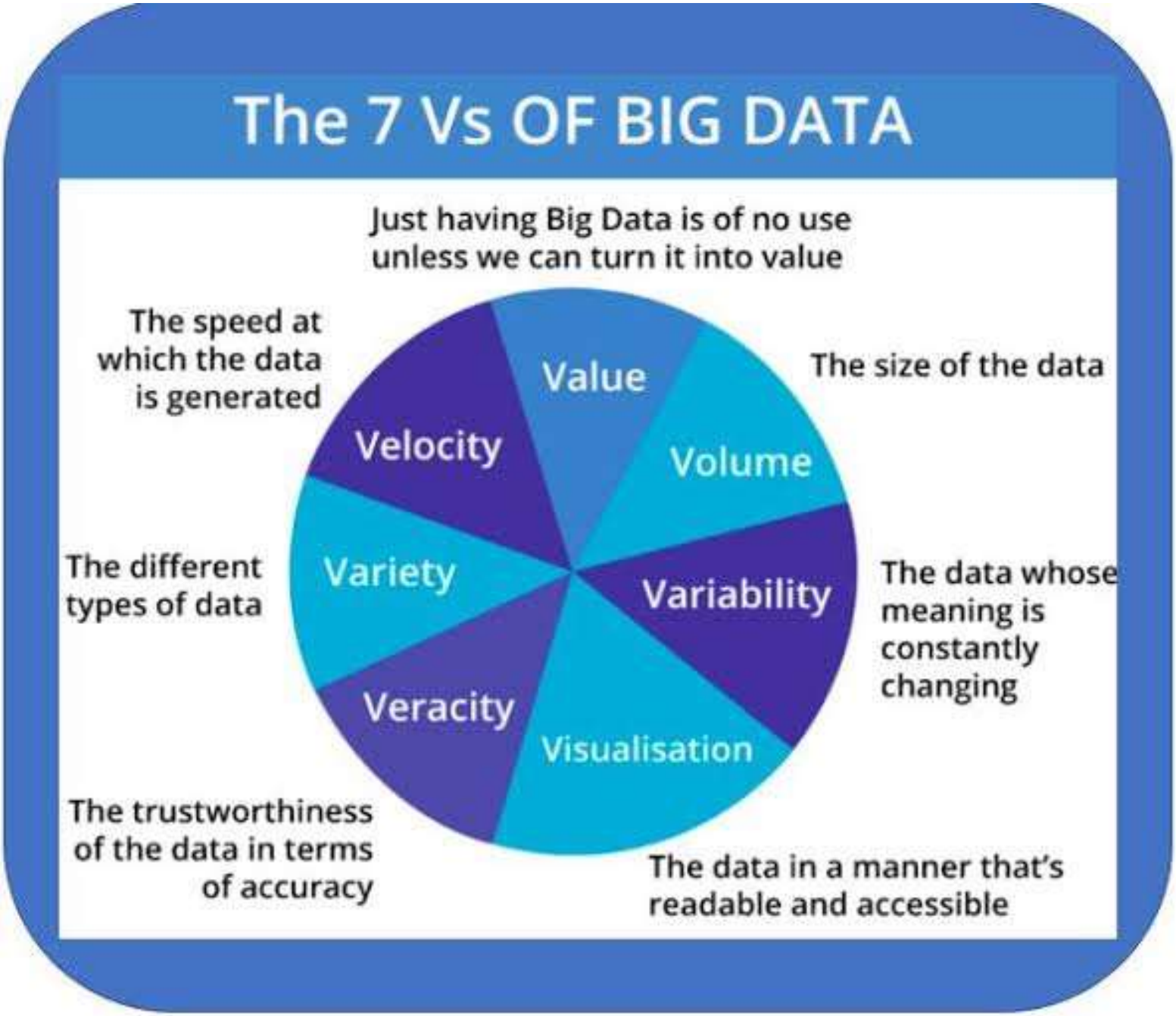
Courtesy: [3]

V3: V for Velocity

- Data is being generated fast and need to be processed fast
  - For time-sensitive processes such as catching fraud, big data must be used as it streams into your enterprise in order to maximize its value
  - Scrutinize 5 million trade events created each day to identify potential fraud
  - Analyze 500 million daily call detail records in real-time to predict customer churn faster
- Sometimes, 2 minutes is too late!
  - The latest we have heard is 10 ns (nano seconds) delay is too much

Courtesy: [3]

The 7 Vs OF BIG DATA



References

[1] https://www.slideshare.net/edjuma/data-vs-information-4

[2] https://www.exetercityfutures.com/qsteps-blog/

[3] Dr. Debasis Samanta, Data Analytics, Lecture #1.

[4] https://www.diffen.com/difference/Data_vs_Information

[5] https://bigdatapath.wordpress.com/2019/11/13/understanding-the-7-vs-of-big-data/

The Rise of NoSQL

- NoSQL = Not only SQL
- A fundamental shift or alternative to storing data which does not conform to the relational format
- Features
  - Schema Agnostic
  - Non relational
  - Highly distributable

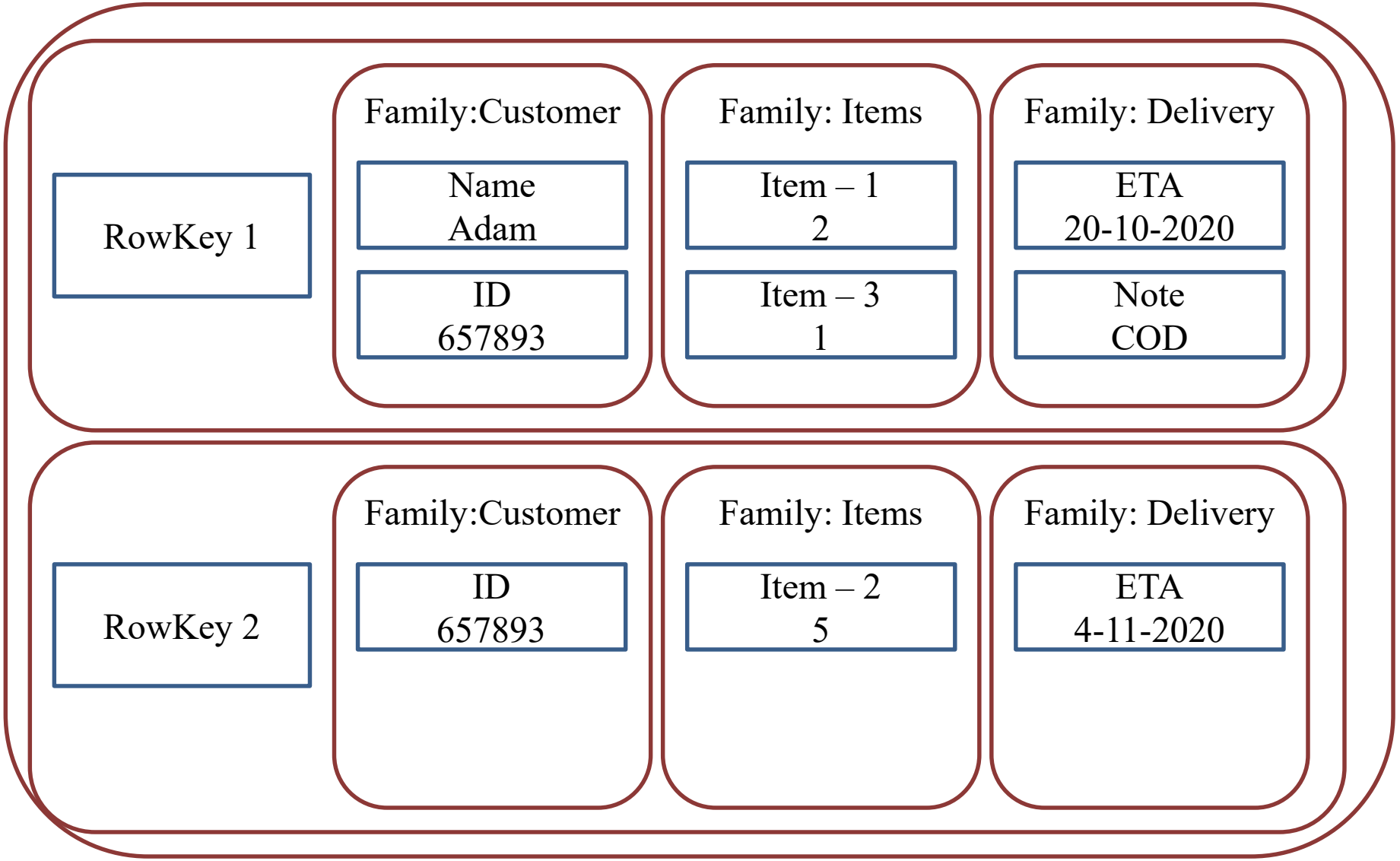
Four Core NoSQL Varieties

- Columnar
- Key-Value
- Triple
- Document

Columnar Databases

- Similar to relational model – concept of rows and columns still exist
- Optimized and designed for faster column access
- Ideal for running aggregate functions or for looking up records that match multiple columns
- A single record may consist of an ID field and multiple column families
- Each one of these column families consists of several fields. One of these column families may have multiple “rows”

Columnar Databases



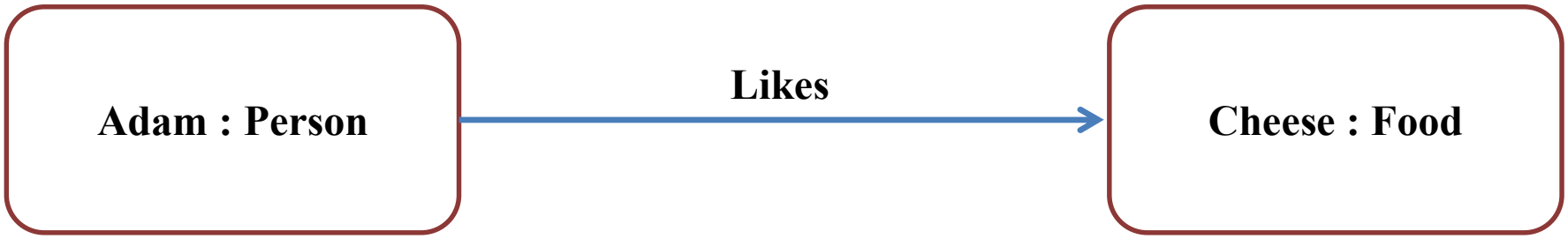
Key – Value Stores

- An ID field — the key in key-value stores — and a set of data
- Some key-value stores support typing (such as integers, strings, and Booleans) and more complex structures for values (such as maps and lists)
- Key-value stores are optimized for speed of ingestion and retrieval

Triple

- Every *fact* or assertion is described as a triple of subject, predicate, and object:
  - A *subject* is the thing you're describing. It may also have a type, which could be a physical object (like a person) or a concept (like a meeting).
  - A *predicate* is the property or relationship belonging to the subject.
  - An *object* is the intrinsic value of a property (such as integer or Boolean, text).

Triple



- Three points of information
  - Adam is a person
  - Cheese is a food item
  - Adam likes cheese

Document Stores

- Hold documents that combine information in a single logical unit
- Retrieving all information from a single document is easier with a database and is more logical for applications
- A document is any unstructured or tree-structured piece of information.
  - It could be a recipe, financial services trade, PowerPoint file, PDF, plain text, or JSON or XML document.

Examples of NoSQL Products

- **Columnar:** DataStax, Apache Cassandra, HBase, Apache Accumulo, Hypertable
- **Key-value:** Basho Riak, Redis, Voldemort, Aerospike, Oracle NoSQL
- **Triple/graph:** Neo4j, Ontotext's GraphDB (formerly OWLIM), MarkLogic, OrientDB, AllegroGraph, YarcData
- **Document:** MongoDB, MarkLogic, CouchDB, FoundationDB, IBM Cloudant, Couchbase

Hadoop Distributed File System

Why Hadoop ?

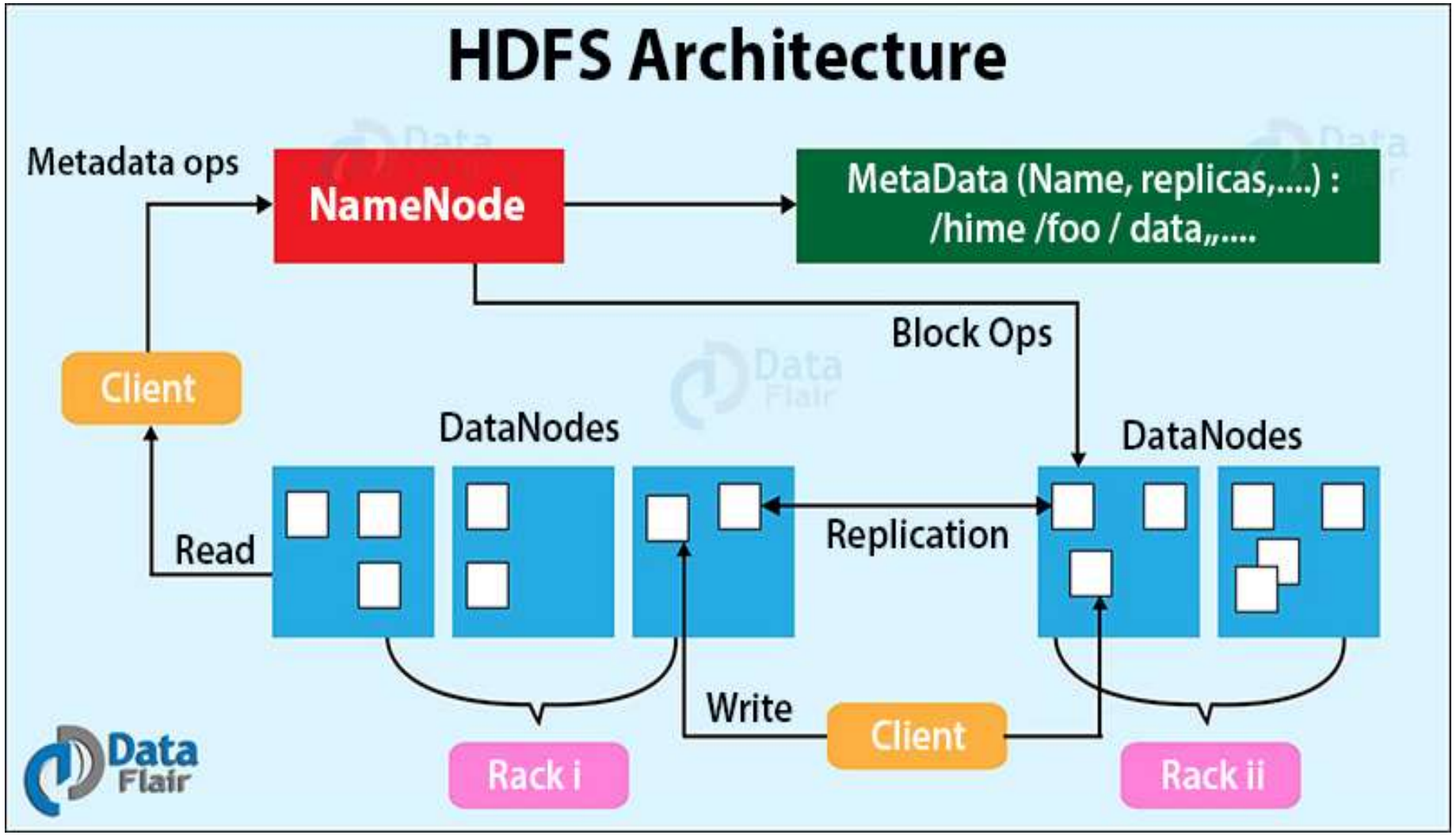
- Need to process huge datasets on large clusters of computers.
- Very expensive to build reliability into each application.
- Nodes fail every day
  - *Failure is expected, rather than exceptional*
  - *The number of nodes in a cluster is not constant*
- Need a common infrastructure
  - *Efficient, reliable, easy to use*
  - *Open Source, Apache Licence*

Introduction

- The Hadoop Distributed File System (HDFS) is a distributed file system designed to run on commodity hardware.
- HDFS is highly fault-tolerant and is designed to be deployed on low-cost hardware.
- HDFS provides high throughput access to application data and is suitable for applications that have large data sets.

Assumptions & Goals

1. Hardware Failure: detection of faults and quick automatic recovery from them is a core architectural goal of HDFS.
2. Streaming Data Access: data is read continuously with a constant bitrate.
3. Large Data Sets
4. Simple Coherency Model: HDFS applications need a write-once-read-many access model for files.
5. "Moving Computation is Cheaper than Moving Data": A computation requested by an application is much more efficient if it is executed near the data it operates on. HDFS provides interfaces for applications to move themselves closer to where the data is located.
6. Portability Across Heterogeneous Hardware and Software Platforms: HDFS has been designed to be easily portable from one platform to another.


HDFS Architecture
Metadata ops
NameNode
MetaData (Name, replicas,....) :
/hime /foo / data„....
Block Ops
Client
DataNodes
DataNodes
Read
Replication
Write
Client
Rack i
Rack ii
Data Flair

HDFS Architecture

- NameNode
- DataNodes
- HDFS Client
- Image Journal
- CheckpointNode
- BackupNode
- Upgrade, File System Snapshots

NameNode – one per cluster

- Maintain The HDFS namespace, a hierarchy of files and directories represented by inodes
- Maintain the mapping of file blocks to DataNodes
  - Read: ask NameNode for the location
  - Write: ask NameNode to nominate DataNodes
- Checkpoint: native files store persistent record of images (no location)
- Stores metadata – filename, location of blocks etc.
- Maintains metadata in memory

Inode: ***a data structure that stores various information about a file in Linux****, such as the access mode (read, write, execute permissions), ownership, file type, file size, group, number of links, etc. Each inode is identified by an integer number. An inode is assigned to a file when it is created.*

DataNodes

- Stores file contents as blocks
- Different blocks of same file are on different data nodes
- Same block is replicated across data nodes
- **Handshake** when connect to the NameNode
  - Verify namespace ID and software version
  - New DN can get one namespace ID when join
- **Register** with NameNode
  - Storage ID is assigned and never changes
  - Storage ID is a unique internal identifier

Contd…

- **Block report:** identify block replicas
  - Block ID, the generation stamp, and the length
  - Send first when register and then send per hour
- **Heartbeats:** message to indicate availability
  - Default interval is three seconds
  - DN is considered “dead” if not received in 10 mins
  - Contains Information for space allocation and load balancing
    - Storage capacity
    - Fraction of storage in use
    - Number of data transfers currently in progress
  - NN replies with instructions to the DN
  - Keep frequent. Scalability

HFDS Client

- A code library exports HDFS interface
- Read a file
  - Ask for a list of DN host replicas of the blocks
  - Contact a DN directly and request transfer
- Write a file
  - Ask NN to choose DNs to host replicas of the first block of the file
  - Organize a pipeline and send the data
  - Iteration
- Delete a file and create/delete directory
- Various APIs
  - Schedule tasks to where the data are located
  - Set replication factor (number of replicas)

Contd…

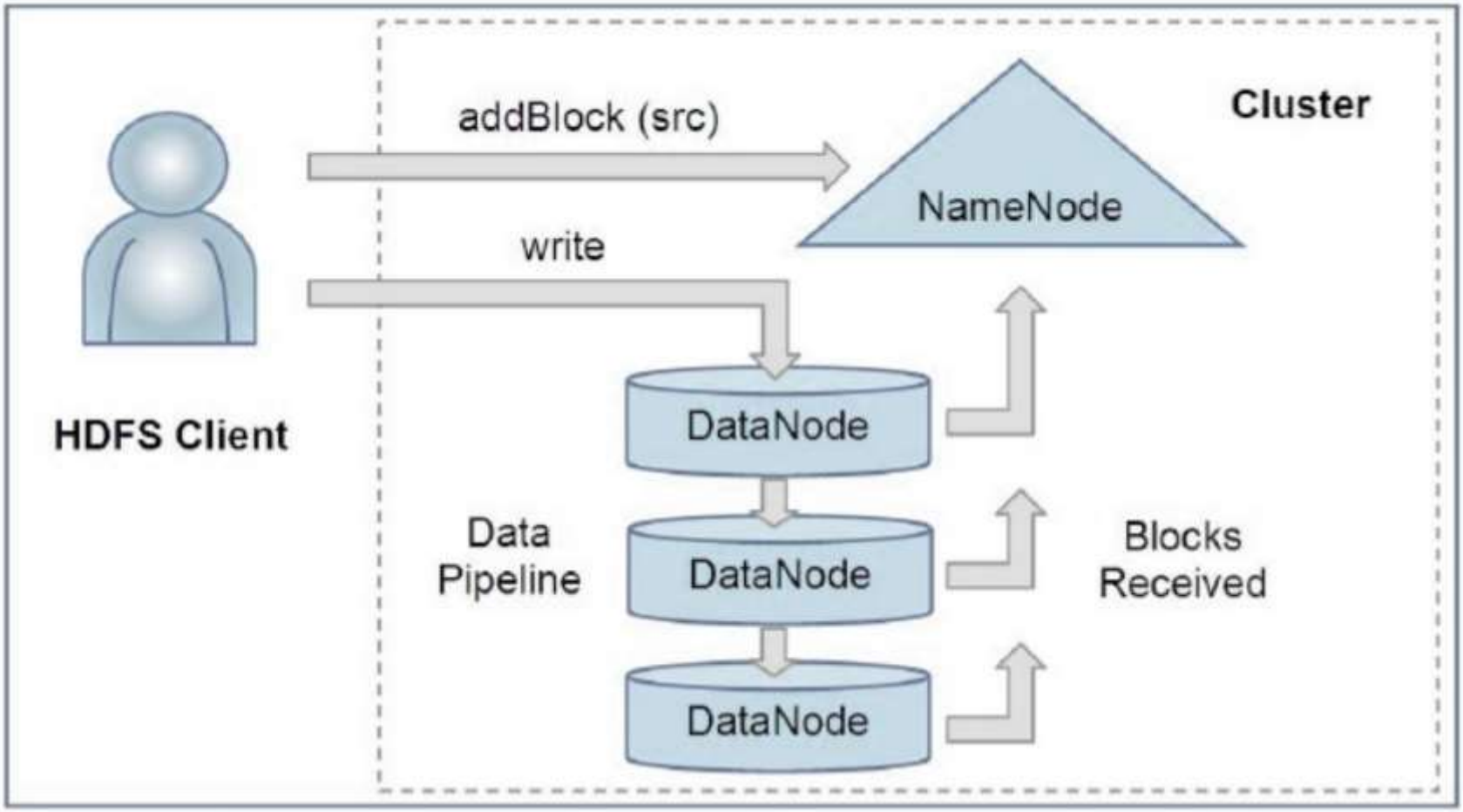

addBlock (src)
Cluster
NameNode
write
HDFS Client
DataNode
Data
Pipeline
DataNode
Blocks
Received
DataNode

Image and Journal

- **Image:** metadata describe organization
  - Persistent record is called checkpoint
  - Checkpoint is never changed, and can be replaced
- **Journal:** log for persistence changes
  - Flushed and synched before change is committed
- Store in multiple places to prevent missing
  - NN shut down if no place is available
- **Bottleneck:** threads wait for flush-and-sync
  - Solution: batch

CheckpointNode

- CheckpointNode is NameNode
- Runs on different host
- Create new checkpoint
  - Download current checkpoint and journal
  - Merge
  - Create new and return to NameNode
  - NameNode truncate the tail of the journal
- **Challenge:** large journal makes restart slow
  - Solution: create a daily checkpoint

BackupNode

- Recent feature
- Similar to CheckpointNode
- Maintain an in memory, up-to-date image
  - Create checkpoint without downloading
- Journal store
- Read-only NameNode
  - All metadata information except block locations
  - No modification

Upgrades, File System and Snapshots

- Minimize damage to data during upgrade
- **NameNode**
  - Merge current checkpoint and journal in memory
  - Create new checkpoint and journal in a new place
  - Instruct DataNodes to create a local snapshot
- **DataNode**
  - Create a copy of storage directory
  - Hard link (mirror copy) existing block files

Some More Points: HDFS Architecture

- HDFS has a master/slave architecture.
- An HDFS cluster consists of a single NameNode, a master server that manages the file system namespace and regulates access to files by clients.
- In addition, there are a number of DataNodes which manage storage attached to the nodes that they run on.
- HDFS exposes a file system namespace and allows user data to be stored in files.
- Internally, a file is split into one or more blocks and these blocks are stored in a set of DataNodes.
- The NameNode executes file system namespace operations like opening, closing, and renaming files and directories.

Contd…

- It also determines the mapping of blocks to DataNodes.
- The DataNodes are responsible for serving read and write requests from the file system's clients.
- The DataNodes also perform block creation, deletion, and replication upon instruction from the NameNode.

Important Points:

- Operates on top of an existing file system.
- Files are stored as blocks
  - *Default size is 64 MB*
- Reliability through replication.
- NameNode stores metadata and manages access.
- No caching due to large file sizes.

Failure and Recovery

- NameNodes keep track of DataNodes through periodic HeartBeat messages
- If no heartbeat is received for a certain duration, DataNode is assumed to be lost
  - NameNode determines which blocks were lost
  - Replicates the same on other DataNodes
- NameNode failure = File system failure
- Two options
  - Persistent backup and checkpointing
  - Secondary/backup NameNode

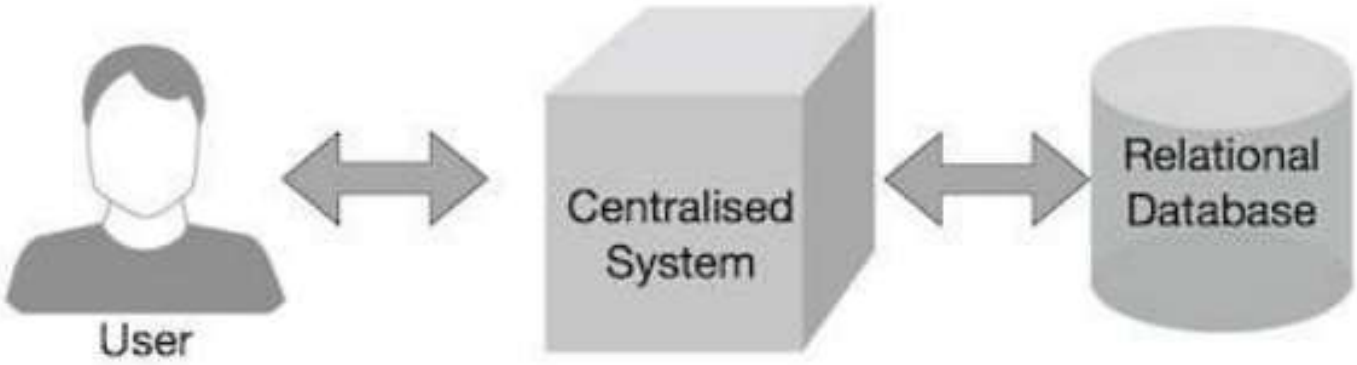
Replication Policy

- **Rule1:** No DataNode contains more than one replica of any block
- **Rule2:** No rack contains more than two replicas of the same block, provided there are sufficient racks on the cluster

Introduction to MapReduce

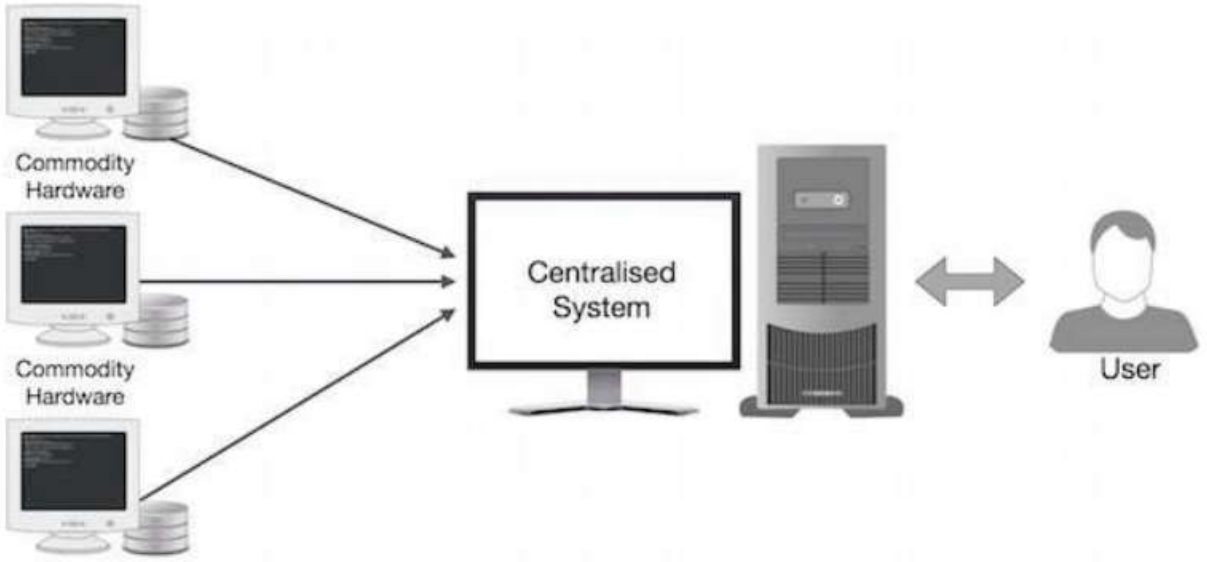
Why MapReduce?

- Traditional Enterprise Systems normally have a centralized server to store and process data.
- Moreover, the centralized system creates too much of a bottleneck while processing multiple files simultaneously.



Contd…

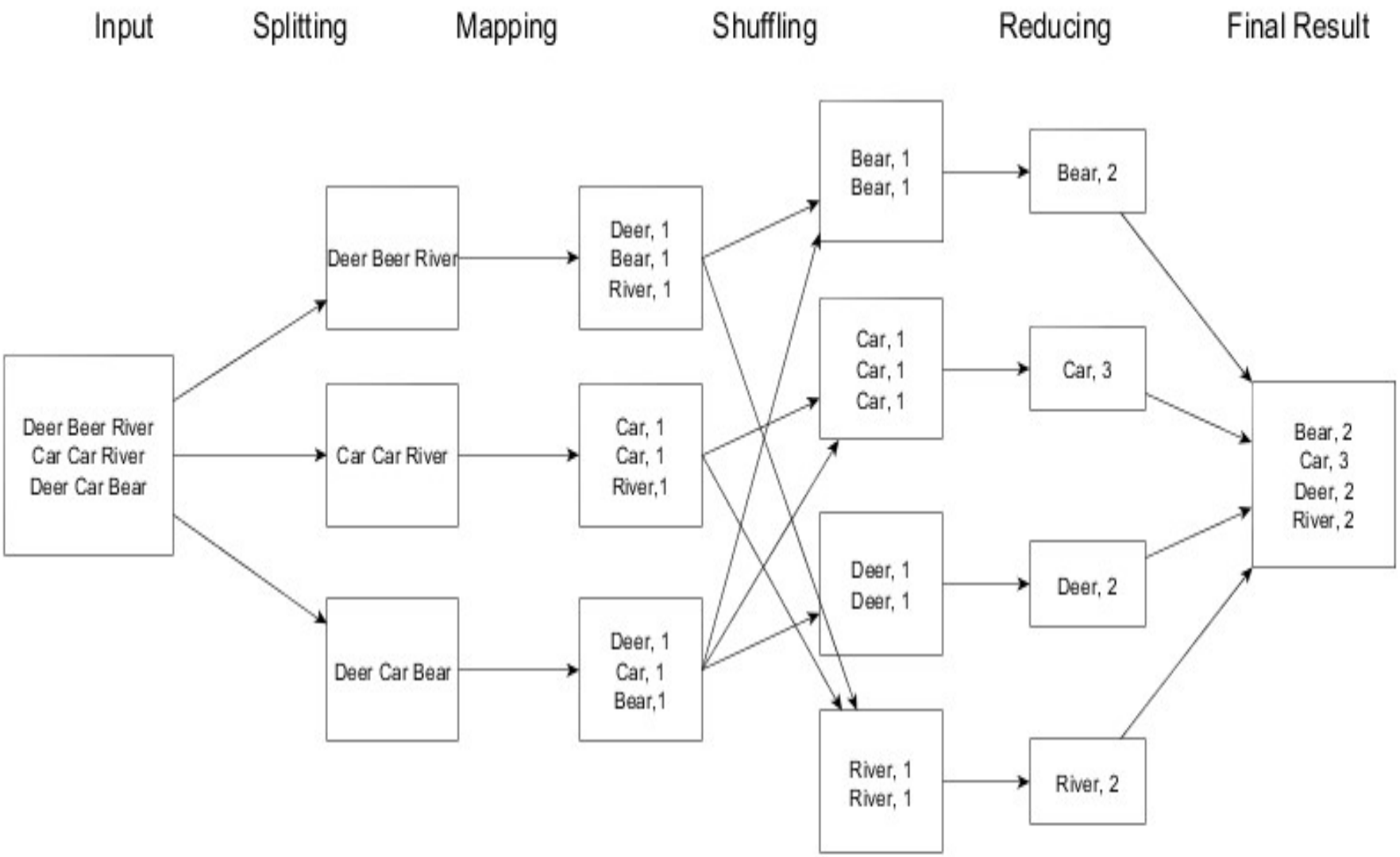
- Google solved this bottleneck issue using an algorithm called MapReduce.
- MapReduce divides a task into small parts and assigns them to many computers.
- Later, the results are collected at one place and integrated to form the result dataset.



How MapReduce Works?

- MapReduce is a software framework for processing large datasets in a distributed fashion over several machines.
  - Map data into multiple <key, value> pairs
  - Reduce over all pairs with the same key.
- Consider a simple example of finding the count of every word present in a document.

Word Count using MapReduce



References

[1] https://cs.uwaterloo.ca/~david/cs848s13/alex-presentation.pdf

[2] https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html

[3]https://hci.stanford.edu/courses/cs448g/a2/files/map_reduce_tutorial.pdf

Cloud Security

Module 6 Part 1

What is cloud security and why it is essential?

Cloud Security

- Cloud Security is security principles/measures applied to protect data, applications and infrastructure associated within the Cloud Computing technology.
- These security measures are configured to protect cloud data, support regulatory compliance and protect customers' privacy as well as setting authentication rules for individual users and devices.
- From authenticating access to filtering traffic, cloud security can be configured to the exact needs of the business.

Authentication

Authentication is the process of *verifying who a user is.*

Operating System usually authenticates user using:

- **What you know?**
  - ✔ Username/Password
- **What you have?**
  - ✔ User Card/key
- **What you possess?**
  - ✔ User Attribute - fingerprint/eye retina

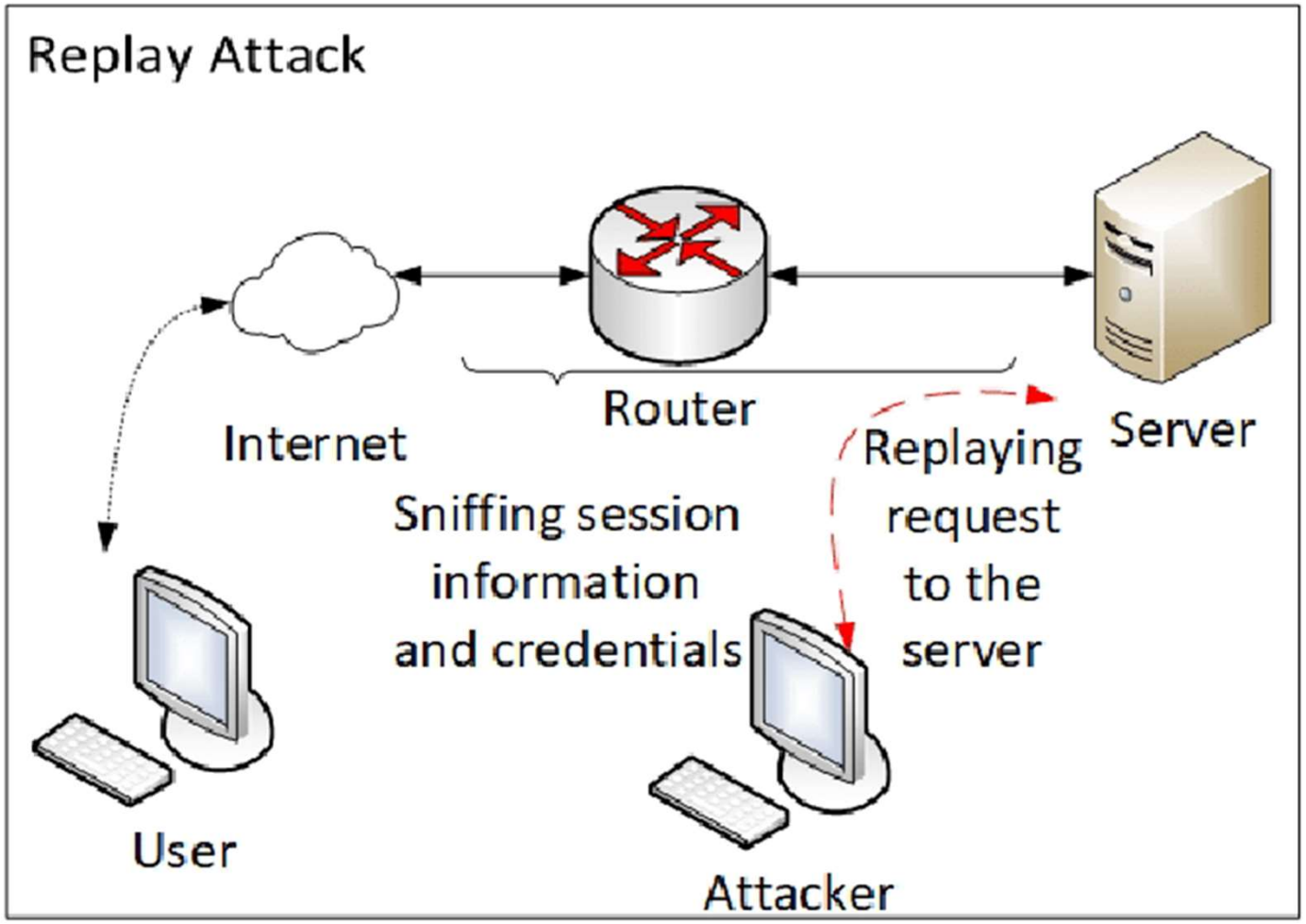
One Time Password (OTP)

a. Random Numbers
b. Secret key
c. Network Password

Advantages of OTP over static password?

Replay Attack

- OTP is not vulnerable to Replay Attack.



Program Threat

Vulnerability vs Threat vs Risk ???

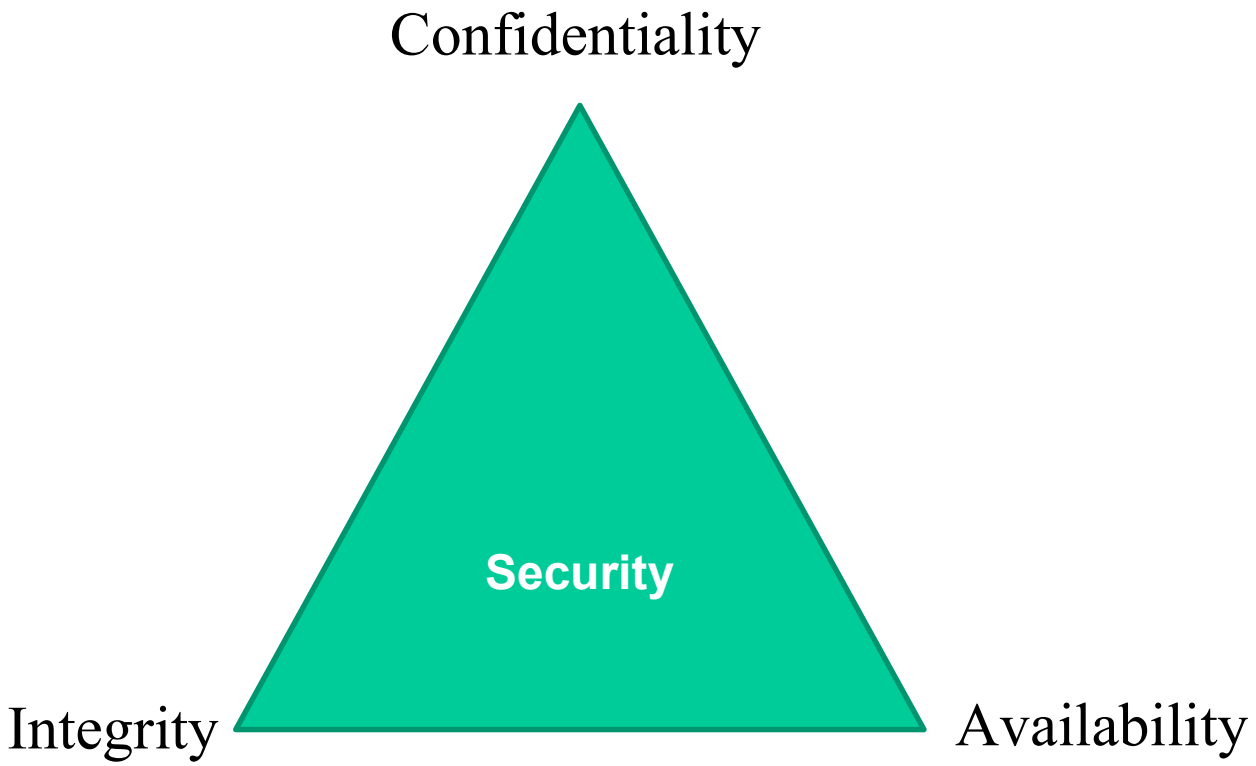
Threat is an action, potential action, or inaction, likely to cause damage, harm or loss.

Program Threat is the process invoked by the program to do malicious tasks.

- Trojan Horse
- Trap Door
- Logic Bomb
- Virus

Understanding Security Risks

CIA Triad



Top Security Risks

- **LOSS OF GOVERNANCE:** in using cloud infrastructures, the client necessarily cedes control to the Cloud Provider (CP) on a number of issues which may affect security. At the same time, SLAs may not offer a commitment to provide such services on the part of the cloud provider, thus leaving a gap in security defenses.

- **LOCK-IN:** there is currently little on offer in the way of tools, procedures or standard data formats or services interfaces that could guarantee data, application and service portability. This can make it difficult for the customer to migrate from one provider to another or migrate data and services back to an in-house IT environment. This introduces a dependency on a particular CP for service provision, especially if data portability, as the most fundamental aspect, is not enabled.

- **ISOLATION FAILURE:** multi-tenancy and shared resources are defining characteristics of cloud computing. This risk category covers the failure of mechanisms separating storage, memory, routing and even reputation between different tenants (e.g., so-called guest-hopping attacks).

- **COMPLIANCE RISKS:** investment in achieving certification (e.g., industry standard or regulatory requirements) may be put at risk by migration to the cloud:

  - if the CP cannot provide evidence of their own compliance with the relevant requirements

  - if the CP does not permit audit by the cloud customer (CC).

In certain cases, it also means that using a public cloud infrastructure implies that certain kinds of compliance cannot be achieved (e.g., PCI DSS).

Contd...

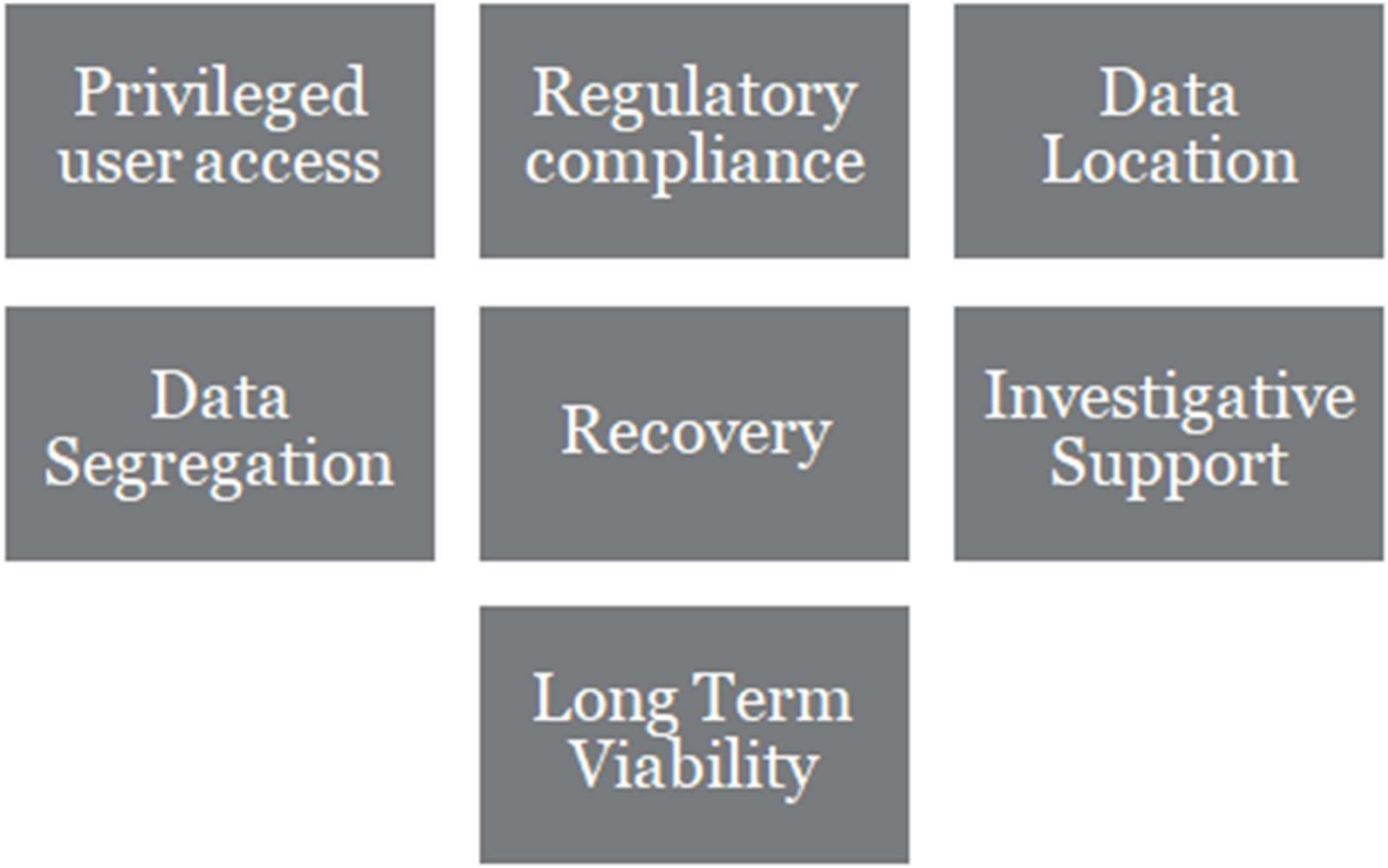
- **MANAGEMENT INTERFACE COMPROMISE:** customer management interfaces of a public cloud provider are accessible through the Internet and mediate access to larger sets of resources (than traditional hosting providers) and therefore pose an increased risk, especially when combined with remote access and web browser vulnerabilities.

- **DATA PROTECTION:** cloud computing poses several data protection risks for cloud customers and providers. In some cases, it may be difficult for the cloud customer (in its role as data controller) to effectively check the data handling practices of the cloud provider and thus to be sure that the data is handled in a lawful way..

- **INSECURE OR INCOMPLETE DATA DELETION:** when a request to delete a cloud resource is made, as with most operating systems, this may not result in true wiping of the data. Adequate or timely data deletion may also be impossible (or undesirable from a customer perspective), either because extra copies of data are stored but are not available, or because the disk to be destroyed also stores data from other clients.

- **MALICIOUS INSIDER:** while usually less likely, the damage which may be caused by malicious insiders is often far greater. Cloud architectures necessitate certain roles which are extremely high-risk. Examples include CP system administrators and managed security service providers.

Gartner's Seven Cloud Computing Security Risks



Securing Multi-Tenant Environment

- Every tenant's workload must be completely isolated from every other tenant's workloads and administrators.
- For organizations to effectively isolate their workloads they need to:
  - Prevent unauthorized communication between one cloud tenant's VMs and virtual networks and any other tenant's resources.
  - Prevent a privileged user from either exposing their workloads to others (accidentally or intentionally) or gaining unauthorized access to another tenant's workloads.
  - Log all virtual administrator activity per tenant to ensure compliance.

Effective Isolation and Workload Security in Multi-tenant Cloud

Goals:

1. **Cloud Protection**

   HyTrust Cloud Control provides advanced privileged user access control, policy enforcement, forensic and automated compliance for private clouds.

2. **Data Encryption**

   HyTrust Data Control provides powerful data-at-rest encryption and integrated key management for workloads running in any cloud environment.

3. **Key Management**

   Encrypting workloads helps enterprises to ensure their data is protected. One of the challenges of workload encryption is scaling the management of encryption keys.

Contd...

- The **HyTrust** Cloud Security Policy Framework makes secure multi tenancy possible by enforcing

  - *access controls and encryption policies for virtual and cloud infrastructure.*

- Effectively segmenting cloud deployments and securely isolating each tenant's critical applications and data.
- detailed logging and analysis of privileged admin account actions.

Vulnerability

- **Vulnerability** is a weakness which can be exploited by an attacker, to perform unauthorized actions within a computer **system**.
- Vulnerability is the intersection of three elements: a system susceptibility or flaw, attacker access to the flaw, and attacker capability to exploit the flaw.
- To exploit a vulnerability, an attacker must have at least one applicable tool or technique that can connect to a system weakness.

Vulnerability Scanning Tools:

- *Nessus*
- *Nikto2*
- *Netsparker, and many more...*

Vulnerabilities In Cloud Computing

Vulnerabilities in Virtual Machines

- Possible covert channels in the collocation of VMs.
- Unrestricted allocation and deallocation of resources with VMs.
- Uncontrolled Migration - VMs can be migrated from one server to another server due to fault tolerance, load balance, or hardware maintenance.
- Uncontrolled snapshots – VMs can be copied in order to provide flexibility, which may lead to data leakage.
- Uncontrolled rollback could lead to reset vulnerabilities - VMs can be backed up to a previous state for restoration, but patches applied after the previous state disappear.
- VMs have IP addresses that are visible to anyone within the cloud - attackers can map where the target VM is located within the cloud (Cloud cartography).

Vulnerabilities In Cloud Computing

Vulnerabilities in Virtual Machine Images

- Uncontrolled placement of VM images in public repositories.
- VM images are not able to be patched since they are dormant artifacts.

Vulnerabilities in Virtual Networks

- The cloud characteristic ubiquitous network access means that cloud services are accessed via network using standard protocols. In most cases, this network is the Internet, which must be considered untrusted. Internet protocol vulnerabilities - such as vulnerabilities that allow man-in-the-middle attacks - are therefore relevant for cloud computing.
- Sharing of virtual bridges by several virtual machines.

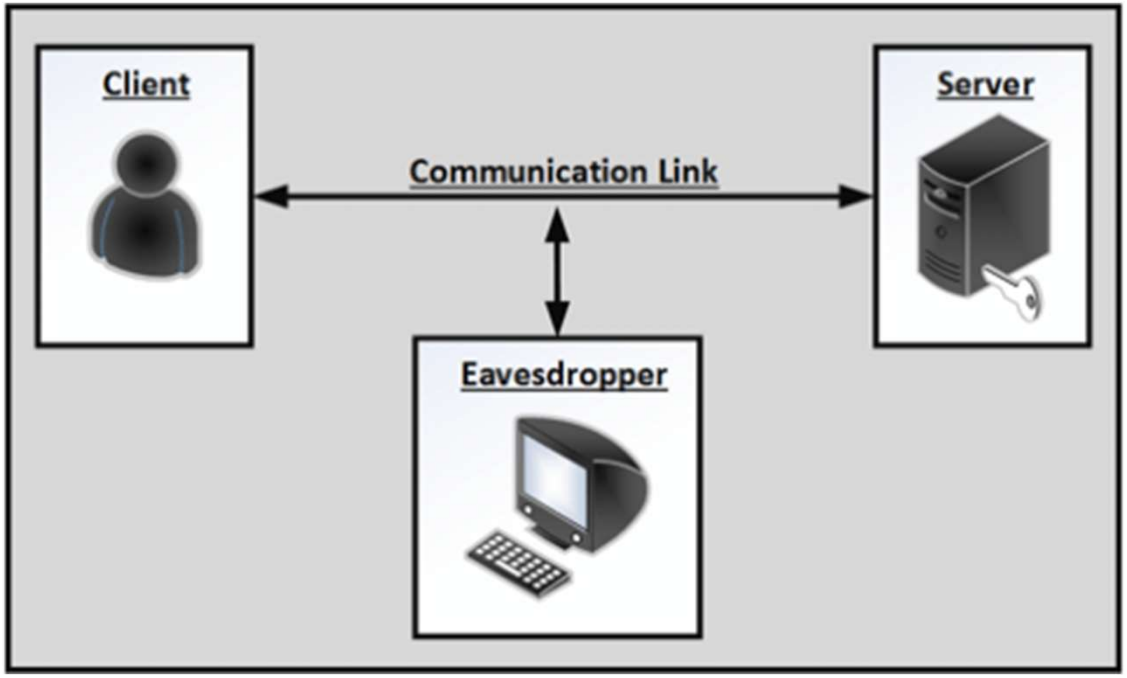
Hypervisor Vulnerabilities

- A typical scenario enabled by exploiting a hypervisor's vulnerability is called *'guest to host escape',*
  allowing the guest to execute code on the host.

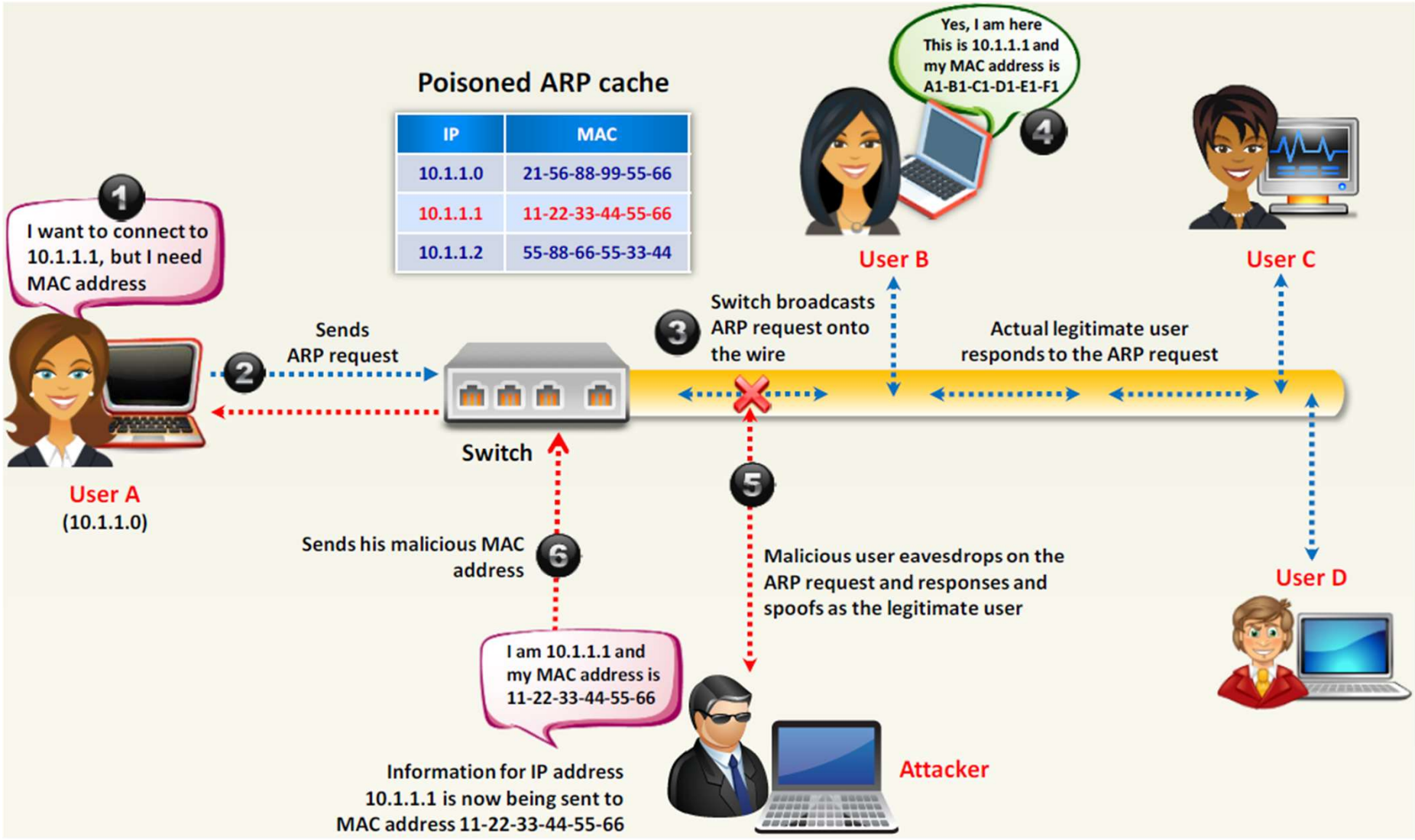
- Another scenario is '*VM hopping*': in which an attacker hacks a VM using some standard method and then – exploiting some hypervisor vulnerability – takes control of other VMs running on the same hypervisor.

Cloud Account/Server Hijacking

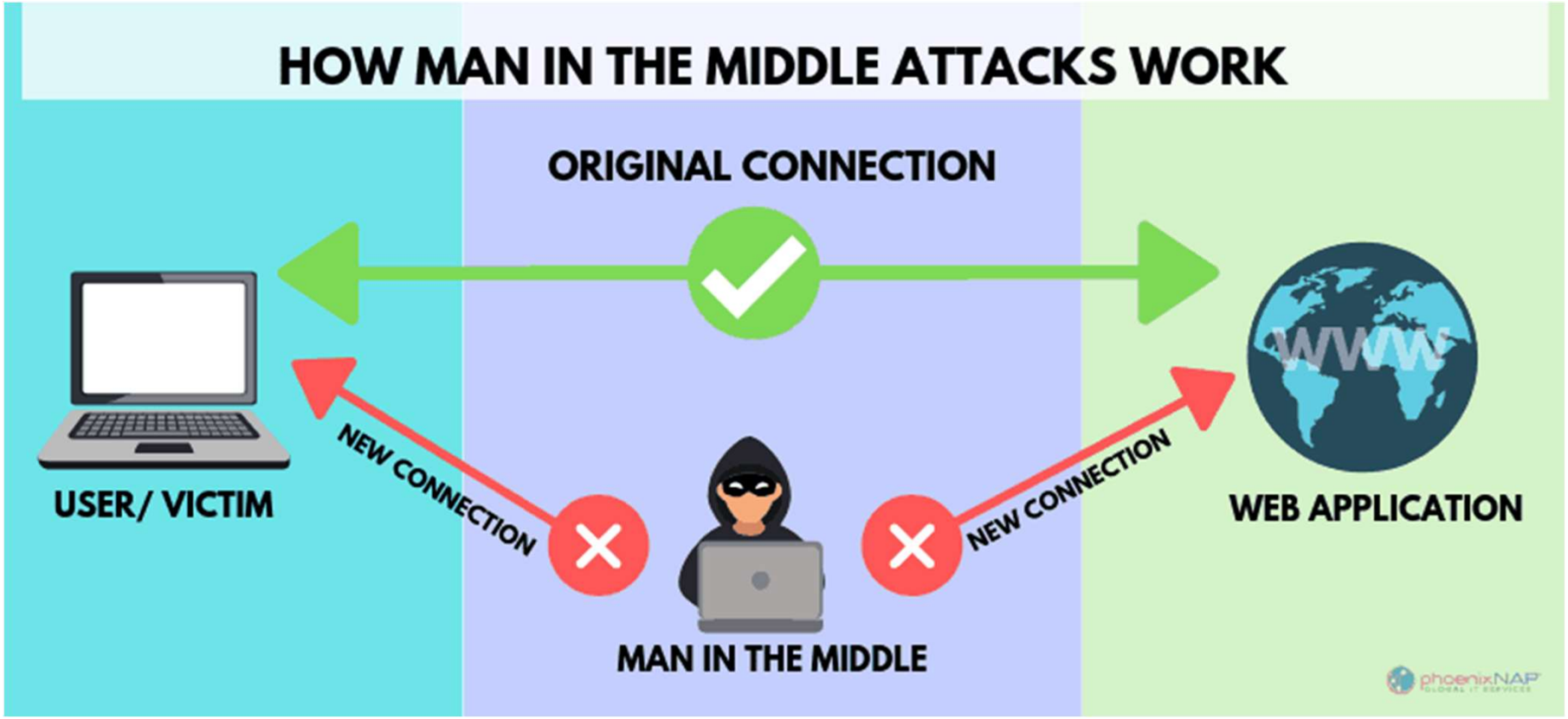
- **Cloud account hijacking** is a common tactic in identity theft schemes in which the attacker uses stolen account information to conduct malicious or unauthorized activity.
- When **cloud account hijacking** occurs, an attacker typically uses a compromised email **account** or other credentials to impersonate the **account** owner.



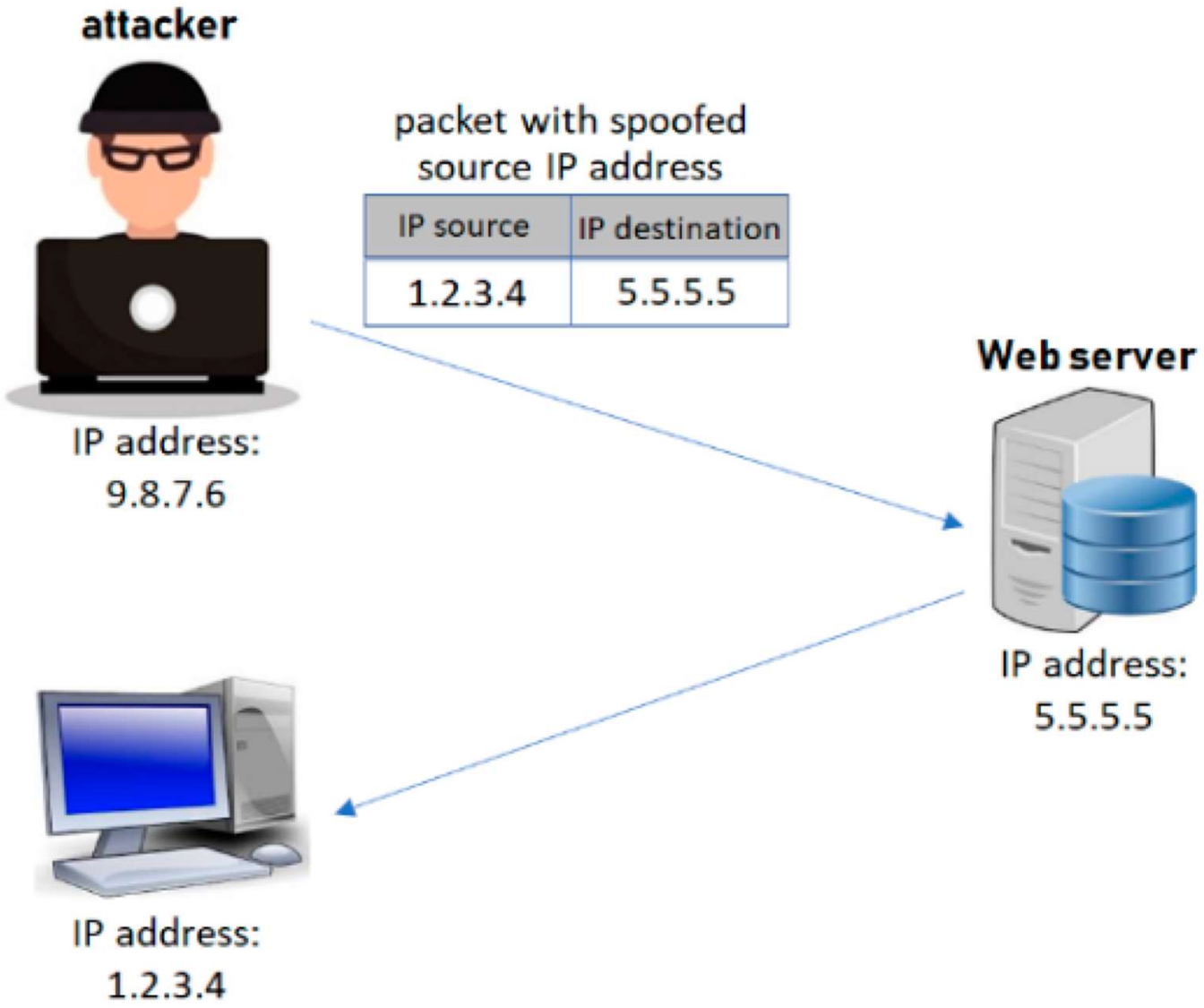
ARP Spoofing



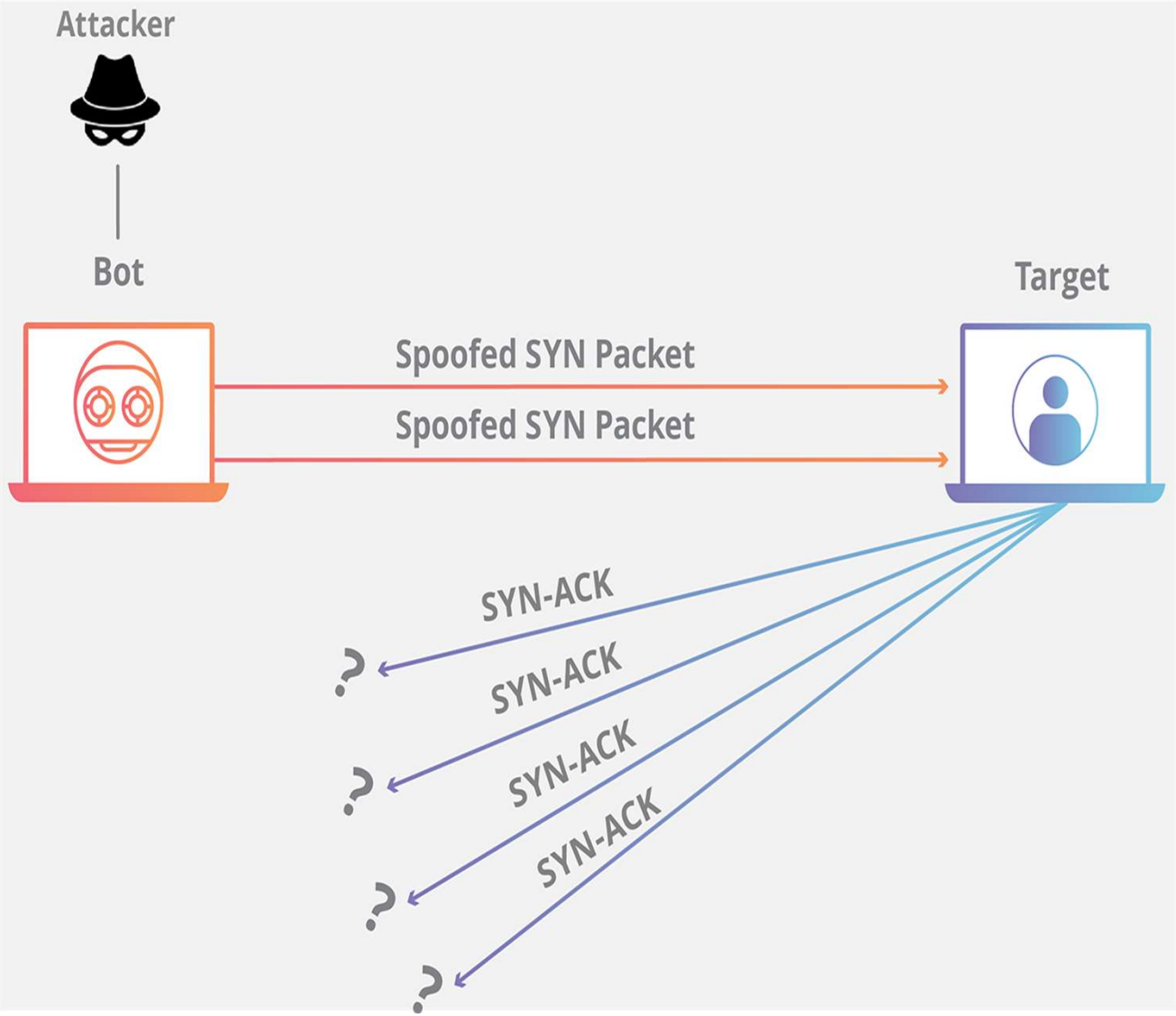
Man-in-the-Middle (MITM) Attack



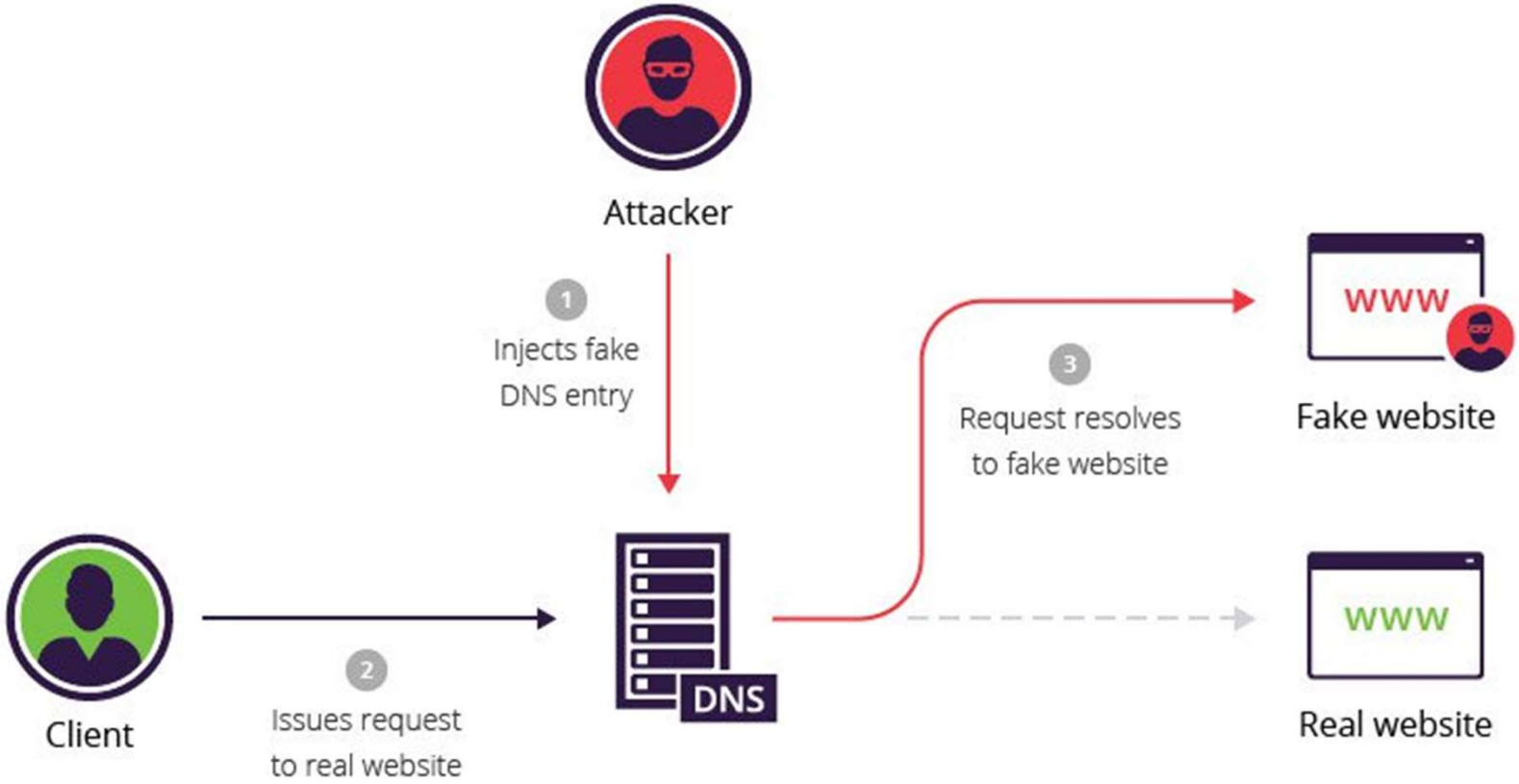
IP Spoofing


attacker
IP address:
9.8.7.6
packet with spoofed
source IP address
IP source
IP destination
1.2.3.4
5.5.5.5
Web server
IP address:
5.5.5.5
IP address:
1.2.3.4

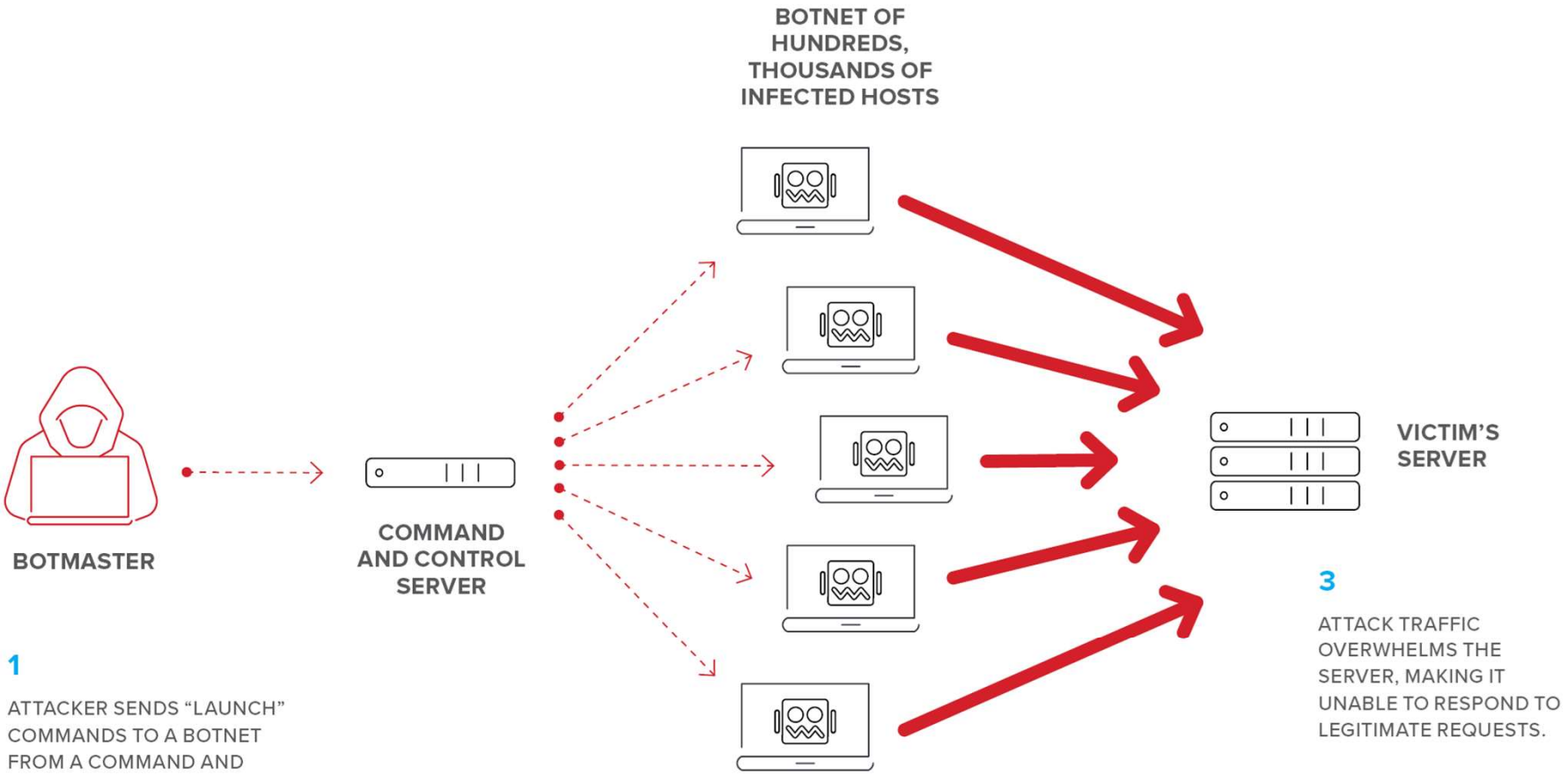
TCP Handshaking in IP Spoofing



DNS Spoofing



DDoS Attack


BOTNET OF HUNDREDS, THOUSANDS OF INFECTED HOSTS
BOTMASTER
COMMAND AND CONTROL SERVER
VICTIM'S SERVER
1
ATTACKER SENDS "LAUNCH" COMMANDS TO A BOTNET FROM A COMMAND AND CONTROL SERVER.
2
BOTS SEND ATTACK TRAFFIC TO VICTIM'S SERVER.
3
ATTACK TRAFFIC OVERWHELMS THE SERVER, MAKING IT UNABLE TO RESPOND TO LEGITIMATE REQUESTS.

Secure Your Cloud

- Ensure local backup
- Avoid storing sensitive information
- Use encryption
- Apply reliable password
- Log everything
- Do not forget the firewall

Objectives

- Encryption techniques
- Data Encryption - Overview
- Symmetric Encryption
- Asymmetric Encryption
- Digital signature
- User Authentication

Encryption Techniques

- **Data Encryption**

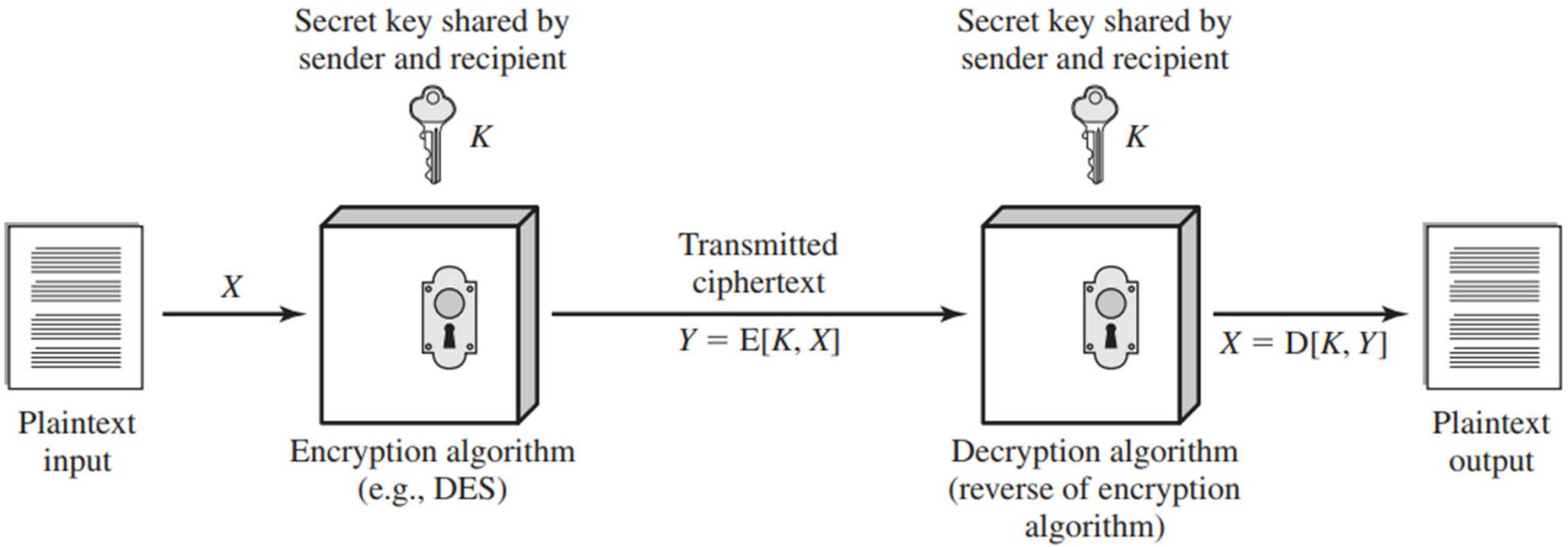
✔ Symmetric Encryption

✔ Asymmetric Encryption

Symmetric Encryption

- Also known as conventional encryption or single-key encryption.
- Used to achieve data confidentiality.

Symmetric Encryption



- **There are two requirements for secure use of symmetric encryption:**

1. We need a strong encryption algorithm.

✔ An opponent who knows the algorithm and has access to one or more ciphertexts would be unable to decipher the ciphertext or figure out the key.

2. Sender and receiver must have obtained copies of the secret key in a secure fashion and must keep the key secure.

- **Two general approaches to attacking a symmetric encryption**

1. *Cryptanalysis:* Exploit knowledge of *algorithm + pairs of plain-text and cipher-text*

2. *Brute-force Attack:* try every possible key on a piece of ciphertext.

- The most commonly used symmetric encryption algorithms are block ciphers.
- Processes the plaintext input in fixed-size blocks and produces a block of ciphertext of equal size for each plaintext block.
- ***Examples:*** Data Encryption Standard (DES), triple DES, and the Advanced Encryption Standard (AES)

	DES	Triple DES	AES
Plaintext block size (bits)	64	64	128
Ciphertext block size (bits)	64	64	128
Key size (bits)	56	112 or 168	128, 192, or 256

DES = Data Encryption Standard
AES = Advanced Encryption Standard

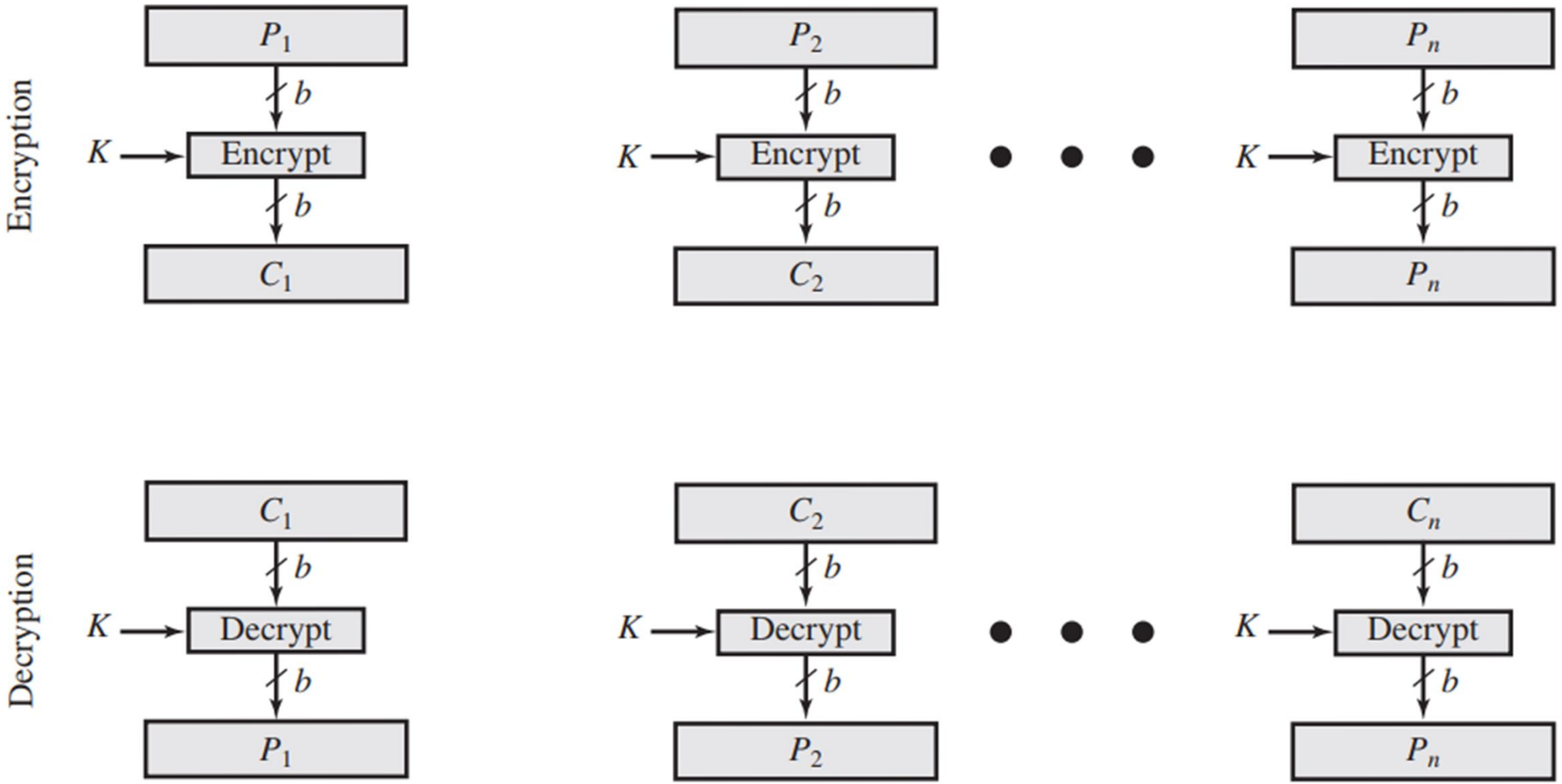
Comparison of Three Popular Symmetric Encryption Algorithms

Average Time Required for Exhaustive Key

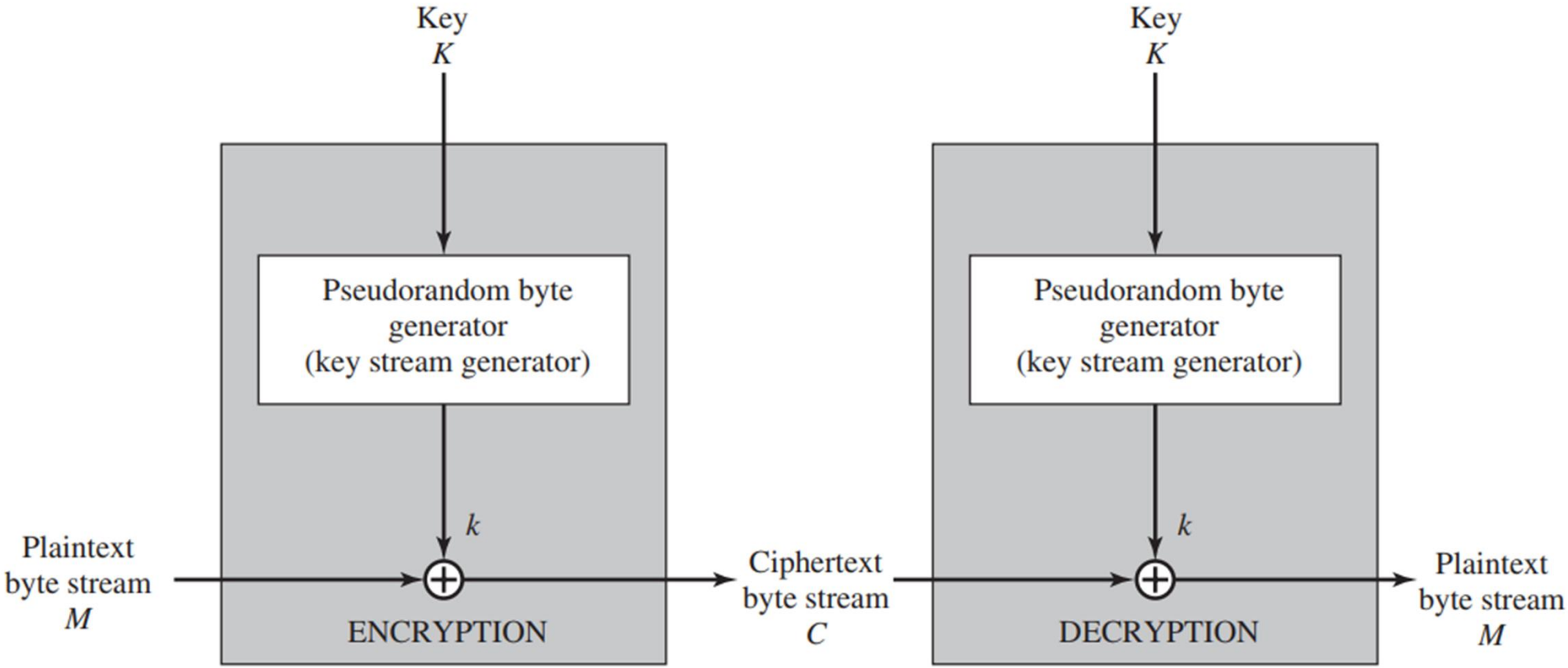
Key size (bits)	Cipher	Number of Alternative Keys	Time Required at 10^9 decryptions/μs	Time Required at 10^{13} decryptions/μs
56	DES	$2^{56} \approx 7.2 \times 10^{16}$	$2^{55}\,\mu s = 1.125$ years	1 hour
128	AES	$2^{128} \approx 3.4 \times 10^{38}$	$2^{127}\,\mu s = 5.3 \times 10^{21}$ years	5.3×10^{17} years
168	Triple DES	$2^{168} \approx 3.7 \times 10^{50}$	$2^{167}\,\mu s = 5.8 \times 10^{33}$ years	5.8×10^{29} years
192	AES	$2^{192} \approx 6.3 \times 10^{57}$	$2^{191}\,\mu s = 9.8 \times 10^{40}$ years	9.8×10^{36} years
256	AES	$2^{256} \approx 1.2 \times 10^{77}$	$2^{255}\,\mu s = 1.8 \times 10^{60}$ years	1.8×10^{56} years

- As key size increases, it reduces the vulnerability of brute-force attack.

Block Cipher Encryption (Electronic Codebook Mode)



Stream Encryption



✔ For lengthy messages, the ECB mode may not be secure. A cryptanalyst may be able to exploit regularities in the plaintext to ease the task of decryption.

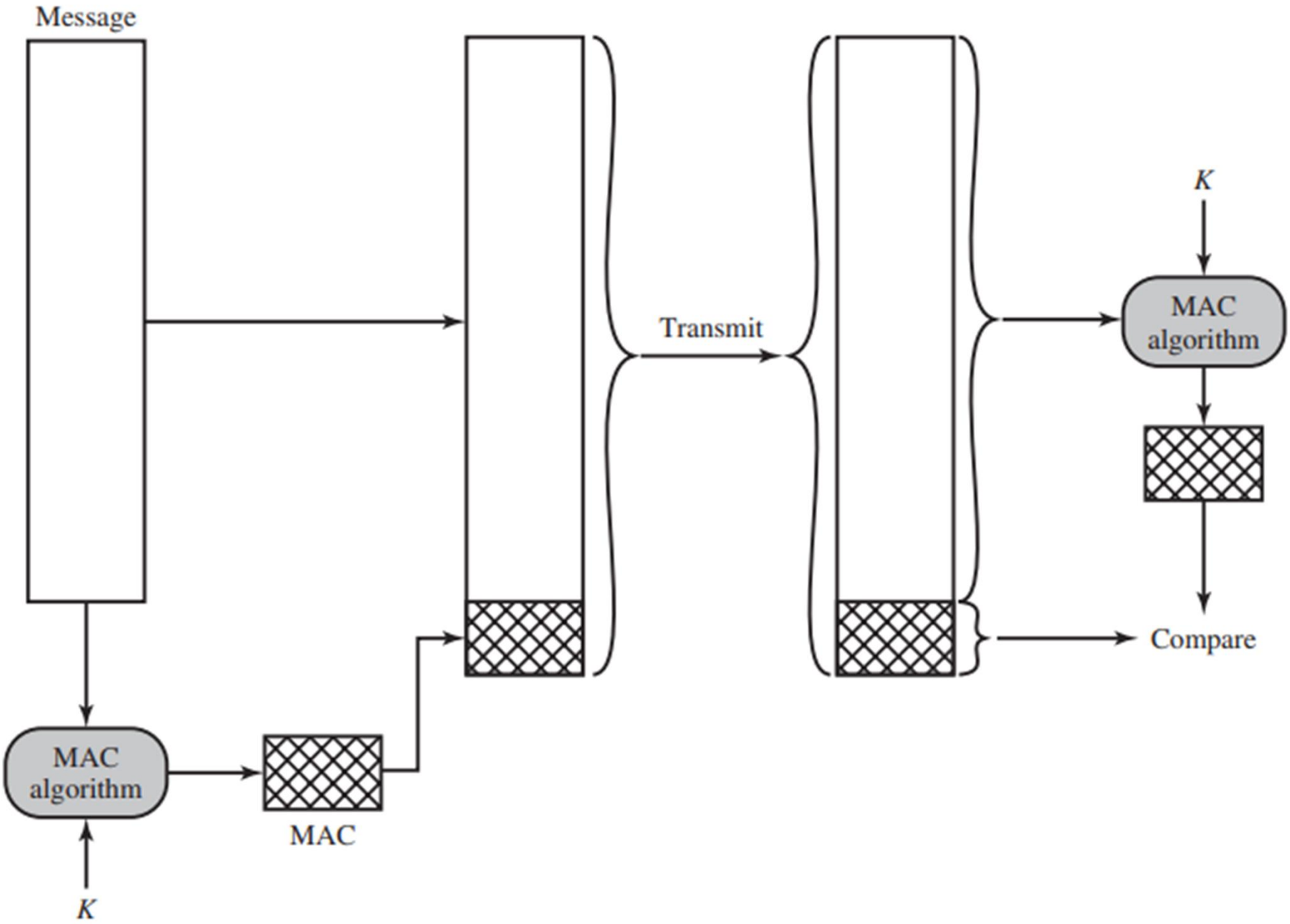
Message Authentication and Hash Function

- Encryption protects against passive attack (eavesdropping).
- A different requirement is to protect against active attack (falsification of data and transactions). Protection against such attacks is known as message or data authentication.
- Message or data authentication is a procedure that allows communicating parties to verify that received or stored messages are authentic.
- The two important aspects are to verify that the contents of the message have not been altered and that the source is authentic.

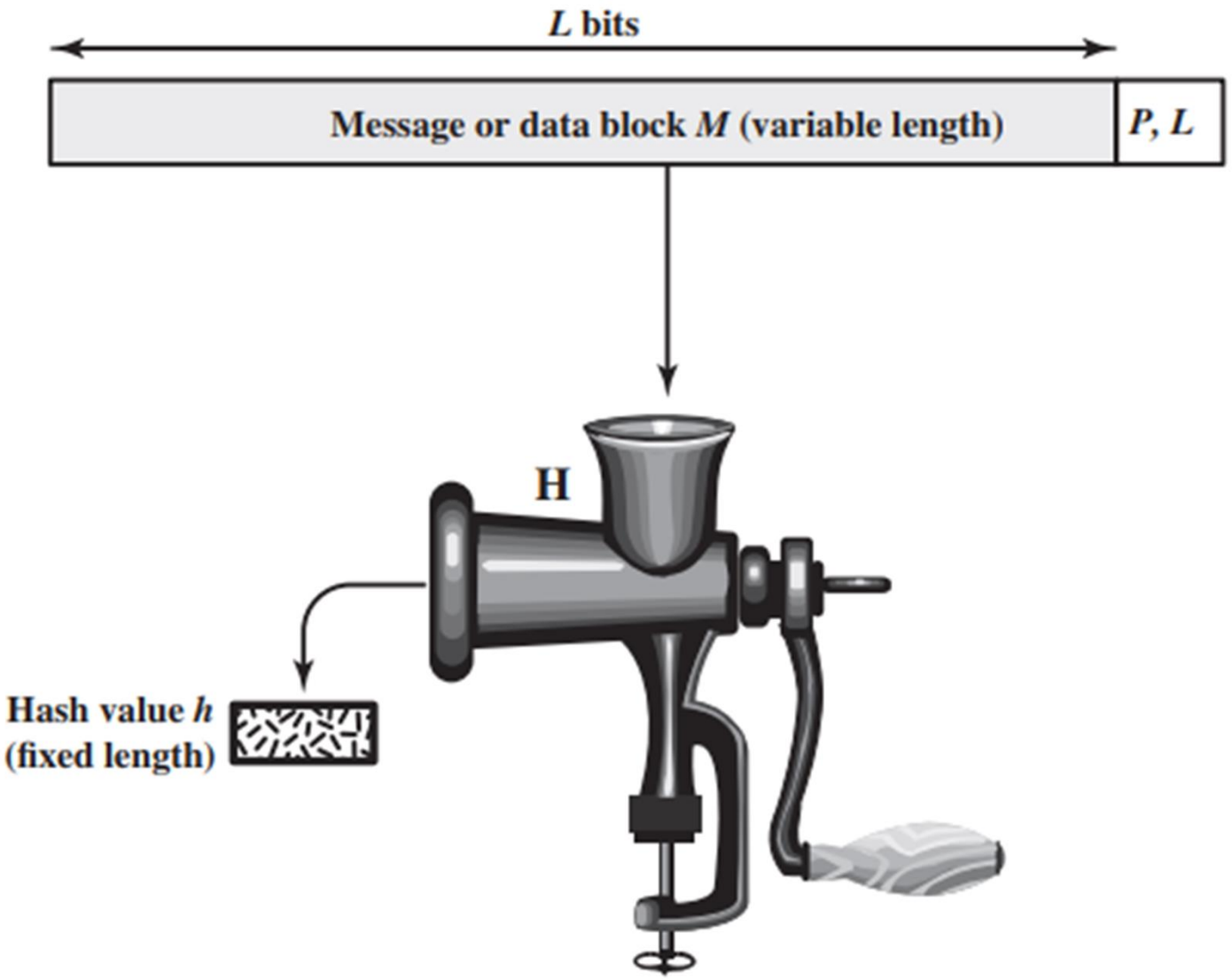
➢ **Authentication Using Symmetric Encryption**

symmetric encryption alone is not a suitable tool for data authentication.

Message Authentication without Message Encryption

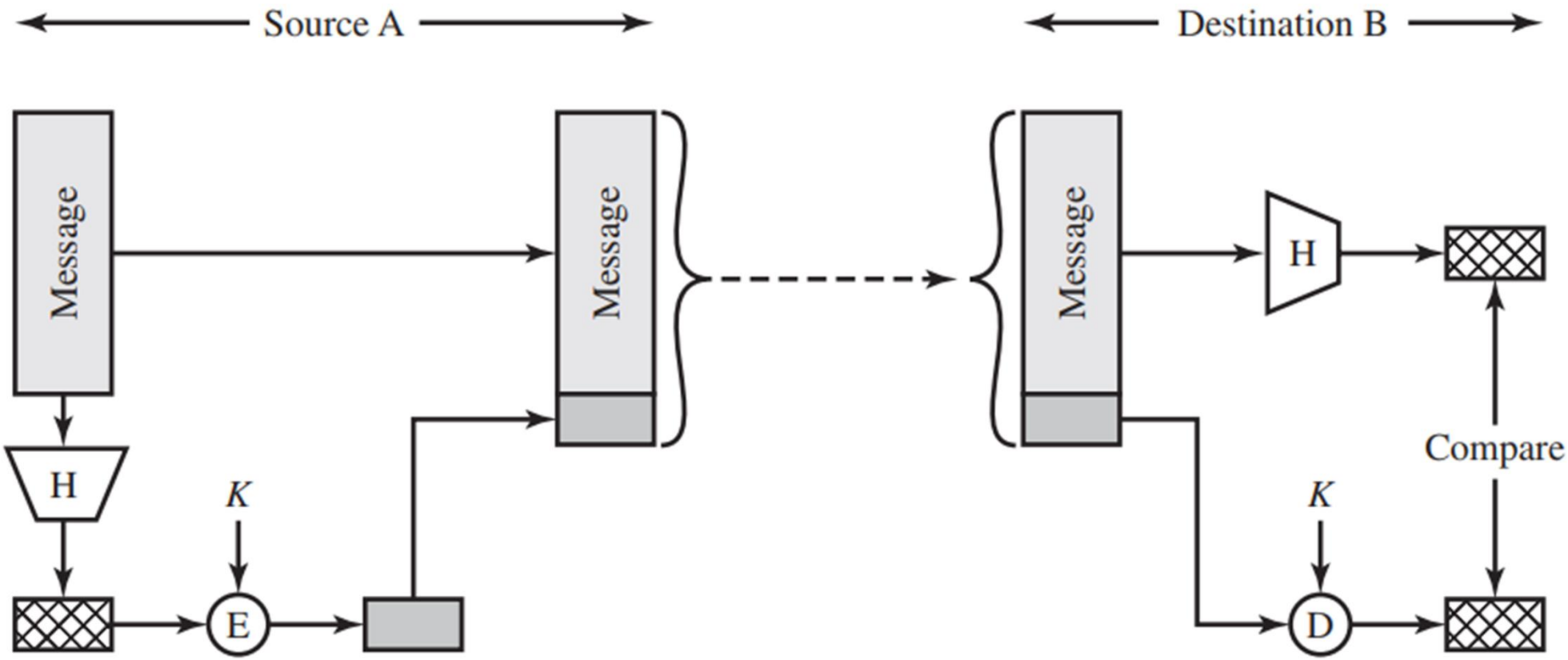


One Way Hash Function



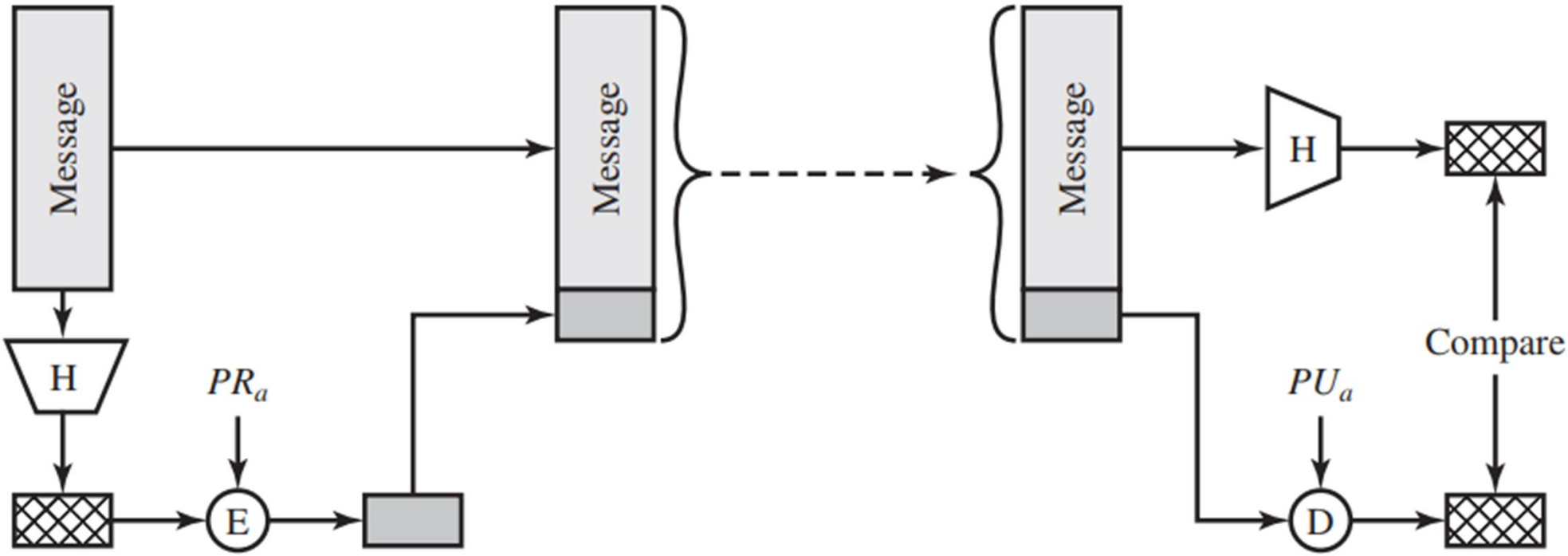
P, L = padding plus length field

Message Authentication using One-way Hash Function



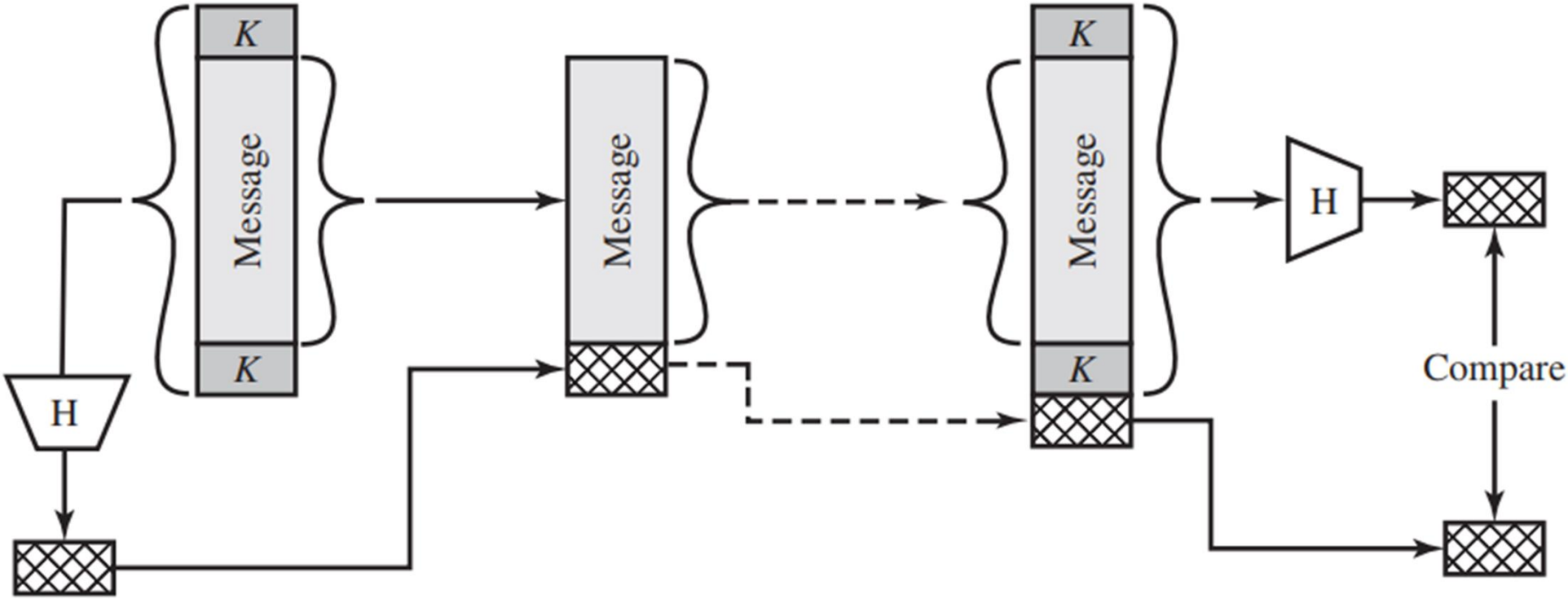
(a) Using symmetric encryption

Continued…



(b) Using public-key encryption

Continued…



(c) Using secret value

Secure Hash Function

- The one-way hash function, or secure hash function, is important not only in message authentication but in digital signatures.

Hash Function Requirements:

The purpose of a hash function is to produce a *"fingerprint"* of a file, message, or other block of data. To be useful for message authentication, a hash function *H* must have the following properties:

1. H can be applied to a block of data of any size.
2. H produces a fixed-length output.
3. H(x) is relatively easy to compute for any given x, making both hardware and software implementations practical.
4. For any given code h, it is computationally infeasible to find x such that H(x) = h. A hash function with this property is referred to as **one-way** or **preimage resistant**.

Continued…

5. For any given block x, it is computationally infeasible to find $y \neq x$ with $H(y) = H(x)$. A hash function with this property is referred to as **second preimage resistant**. This is sometimes referred to as **weak collision resistant**.

6. It is computationally infeasible to find any pair (x, y) such that $H(x) = H(y)$. A hash function with this property is referred to as **collision resistant**. This is sometimes referred to as **strong collision resistant**.

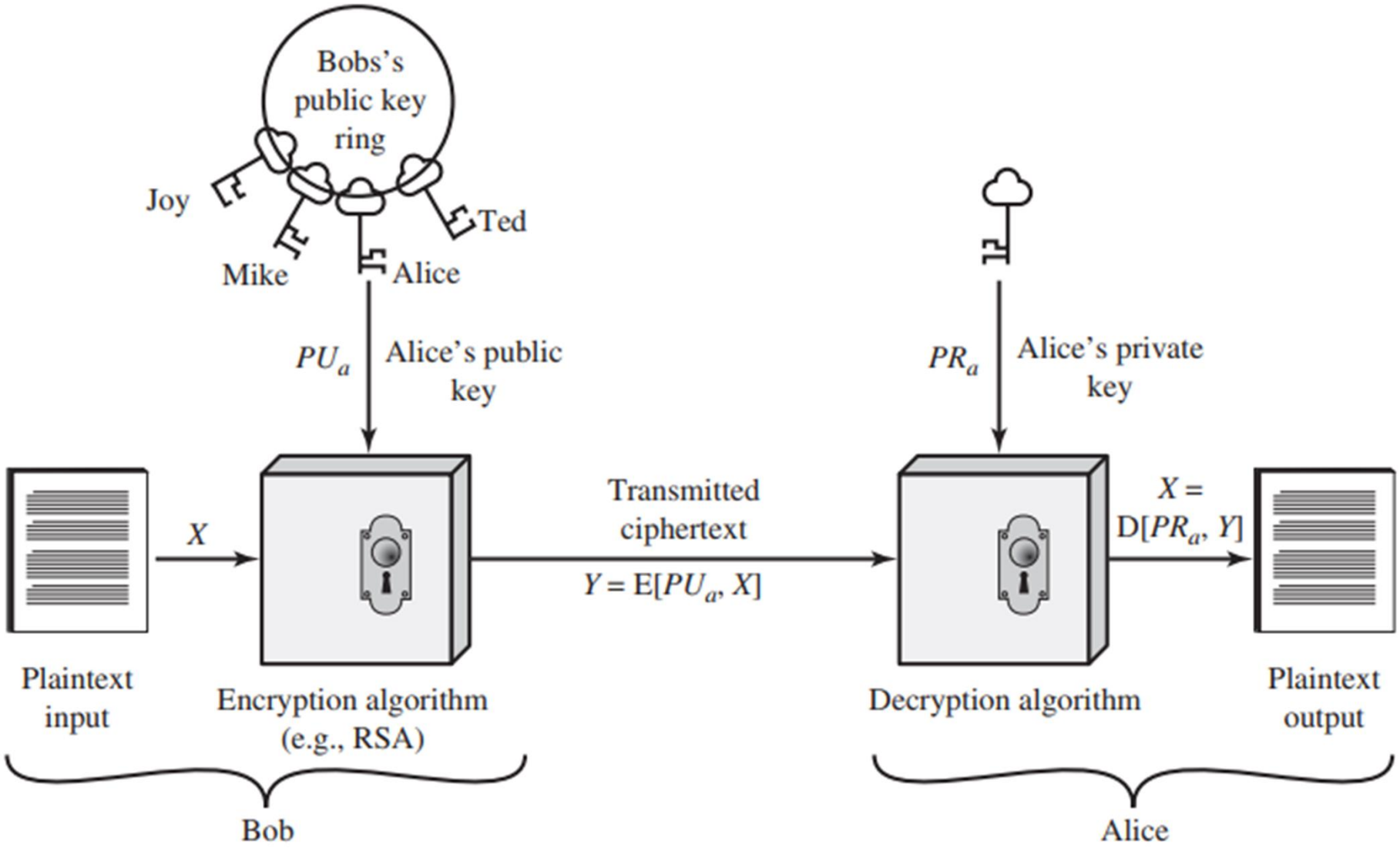
Hash Function Applications:

- Password
- Intrusion Detection

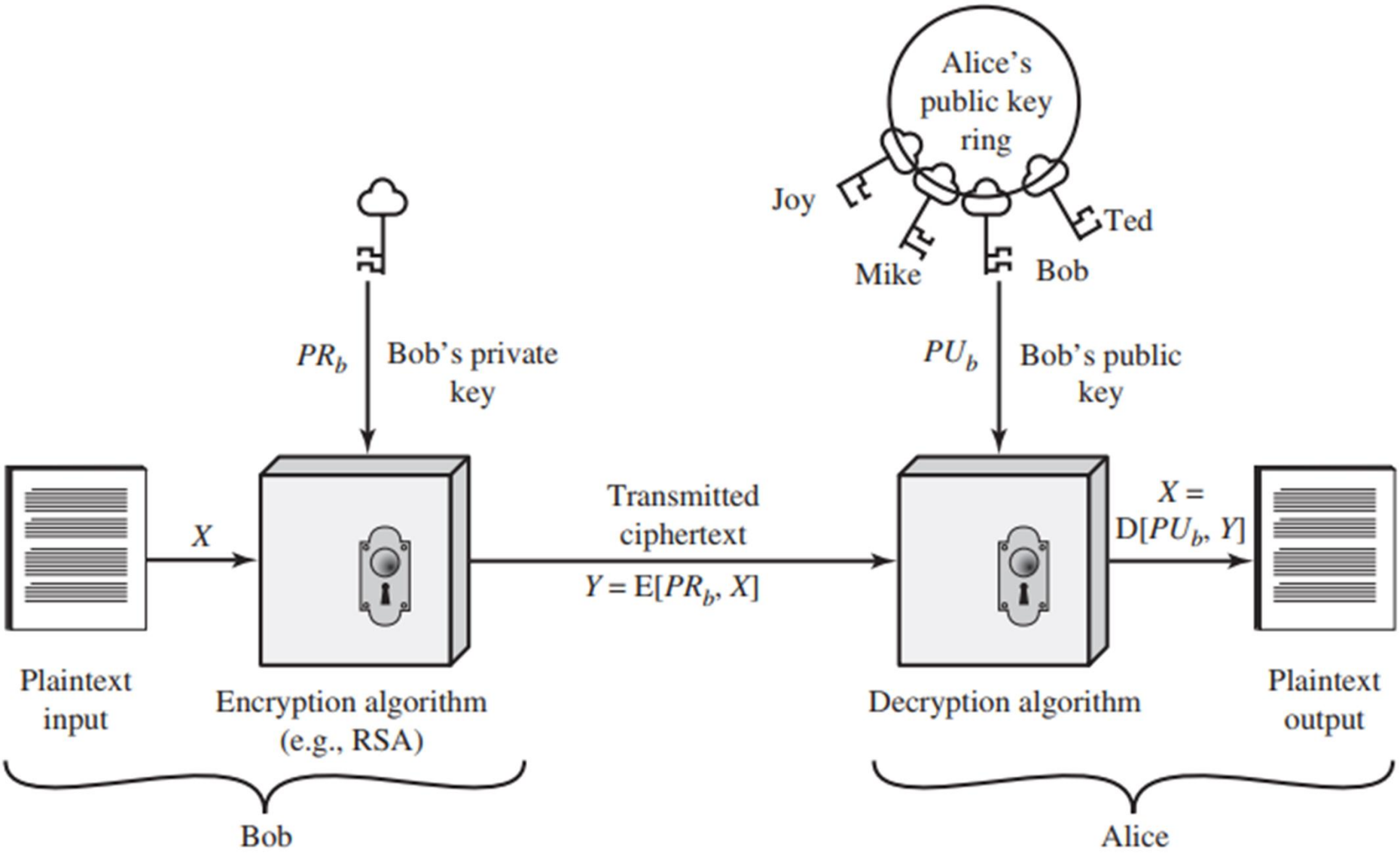
Public Key Encryption

- First publicly proposed by Diffie and Hellman in 1976.
- Public-key cryptography is asymmetric, involving the use of two separate keys, namely, public and private.
- Each user generates a pair of keys to be used for the encryption and decryption of messages.
- Each user places one of the two keys in a public register or other accessible file. This is the public key. The companion key is kept private.

Encryption with Public Key



Encryption with Private Key



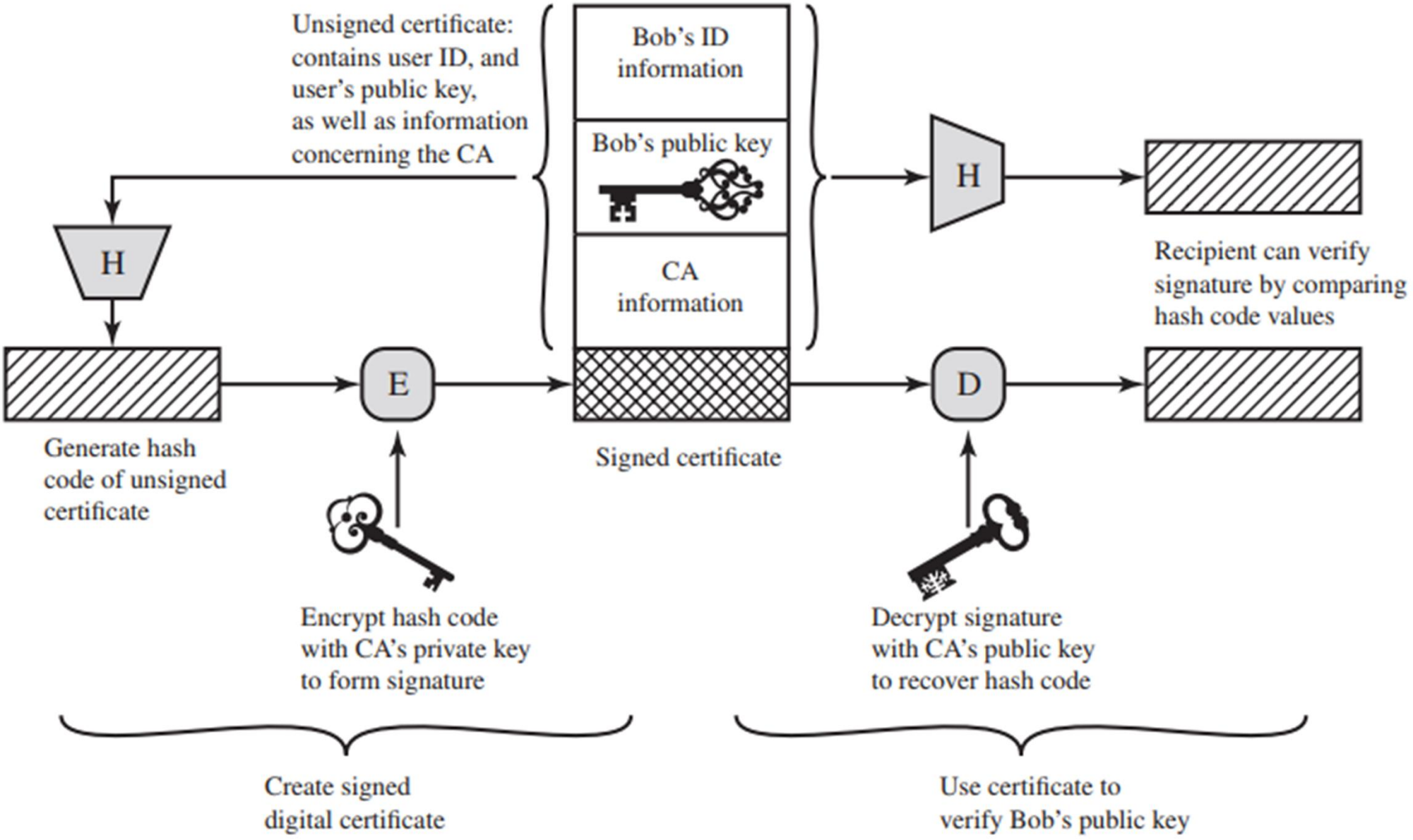
Applications of Public Key Cryptosystem

- Digital Signature
- Symmetric key distribution,
- Encryption of secret keys.

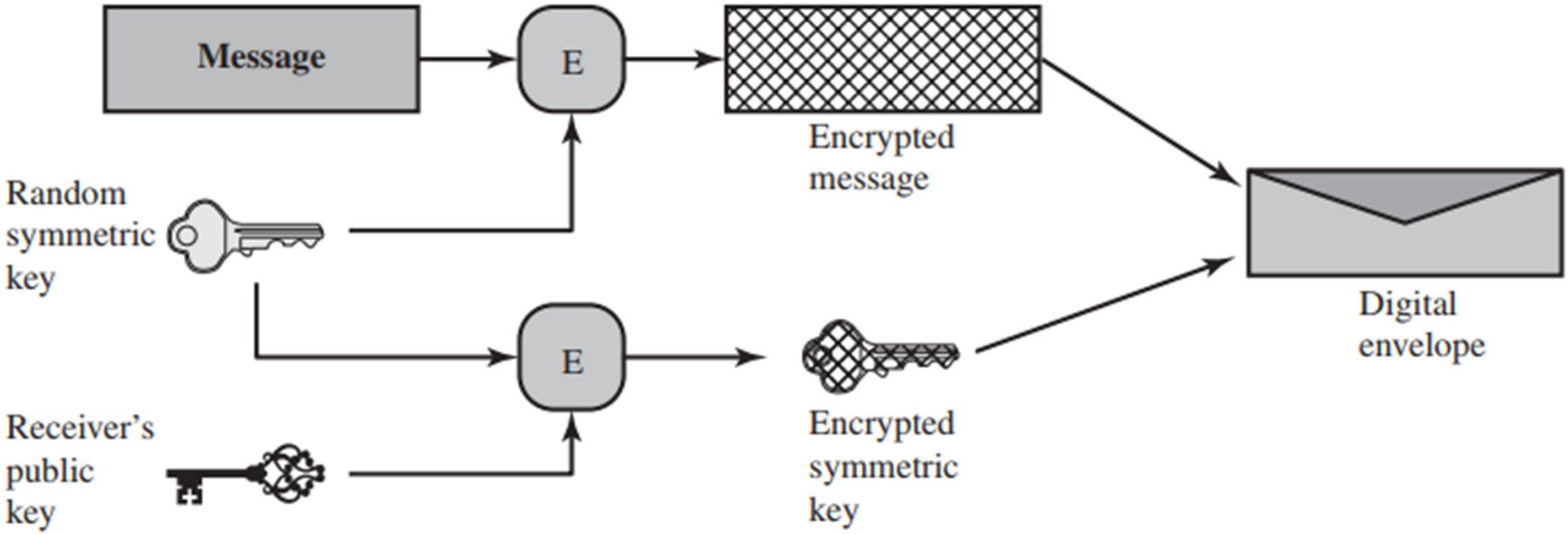
Requirements of Public Key Cryptosystem

1. It is computationally easy for a party B to generate a pair (public key PU_b, private key PR_b).
2. It is computationally easy for a sender A, knowing the public key and the message to be encrypted, M, to generate the corresponding ciphertext: $C = E(PU_b, M)$
3. It is computationally easy for the receiver B to decrypt the resulting ciphertext using the private key to recover the original message: $M = D(PR_b, C) = D[PR_b, E(PU_b, M)]$
4. It is computationally infeasible for an opponent, knowing the public key, PU_b, to determine the private key, PR_b

Digital Signature

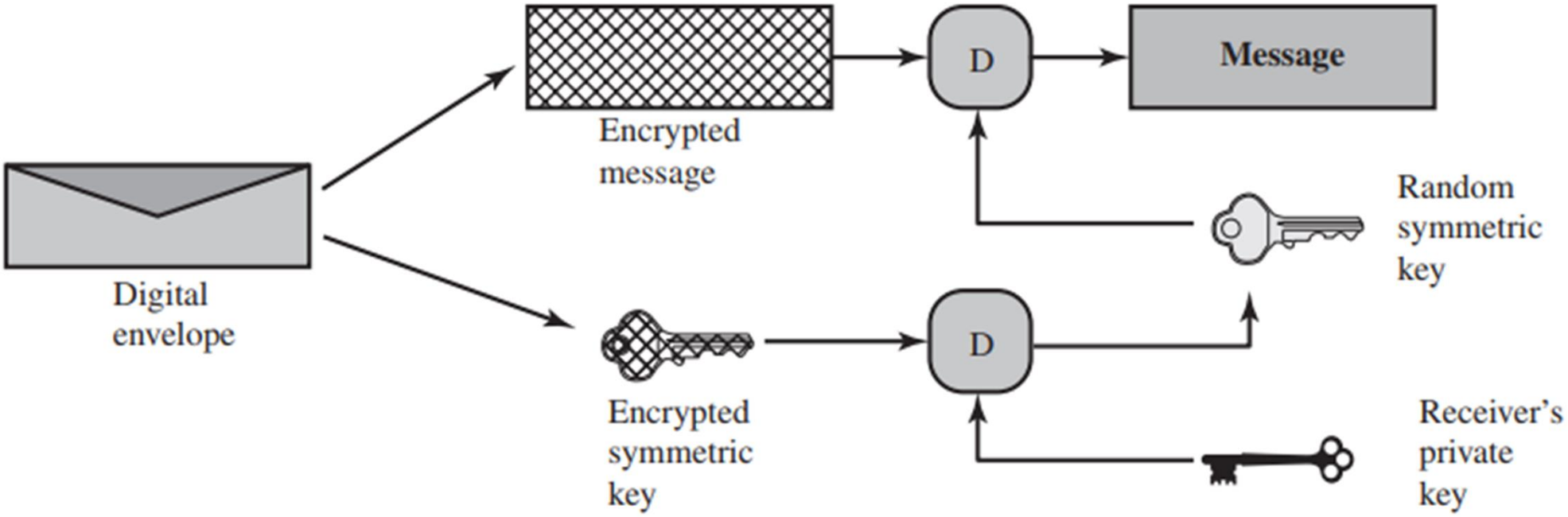


Digital Envelopes



(a) Creation of a digital envelope

Digital Envelopes Continued…



(b) Opening a digital envelope

User Authentication

- An authentication process consists of two steps:

1. **Identification step:** Presenting an identifier to the security system. (Identifiers should be assigned carefully, because authenticated identities are the basis for other security services, such as access control service.)
2. **Verification step:** Presenting or generating authentication information that corroborates the binding between the entity and the identifier.

1. **Something the individual knows:** Examples includes a password, a personal identification number (PIN), or answers to a prearranged set of questions.
2. **Something the individual possesses:** Examples include electronic keycards, smart cards, and physical keys. This type of authenticator is referred to as a *token*.
3. **Something the individual is (static biometrics):** Examples include recognition by fingerprint, retina, and face.
4. **Something the individual does (dynamic biometrics):** Examples include recognition by voice pattern, handwriting characteristics, and typing rhythm.

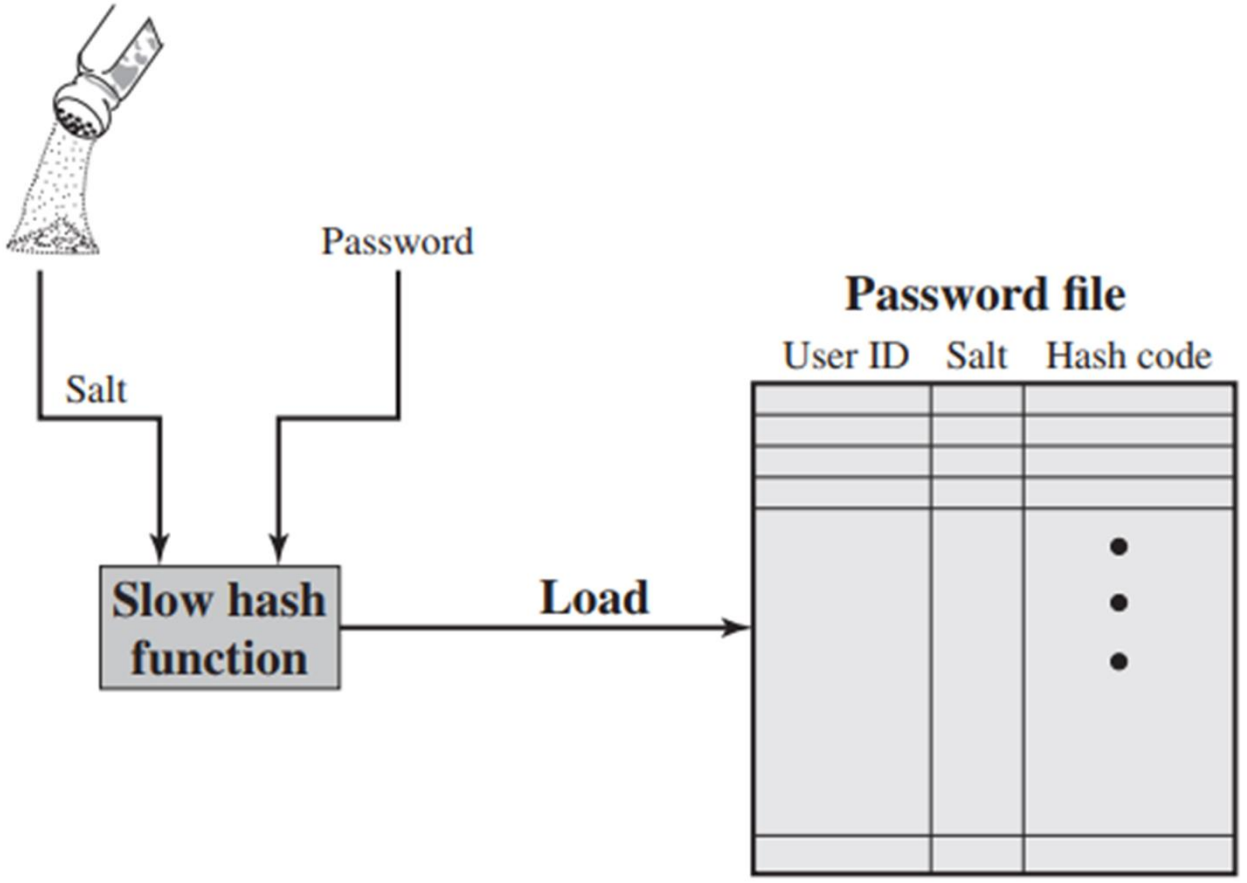
Password Based Authentication

- A widely used line of defense against intruders.

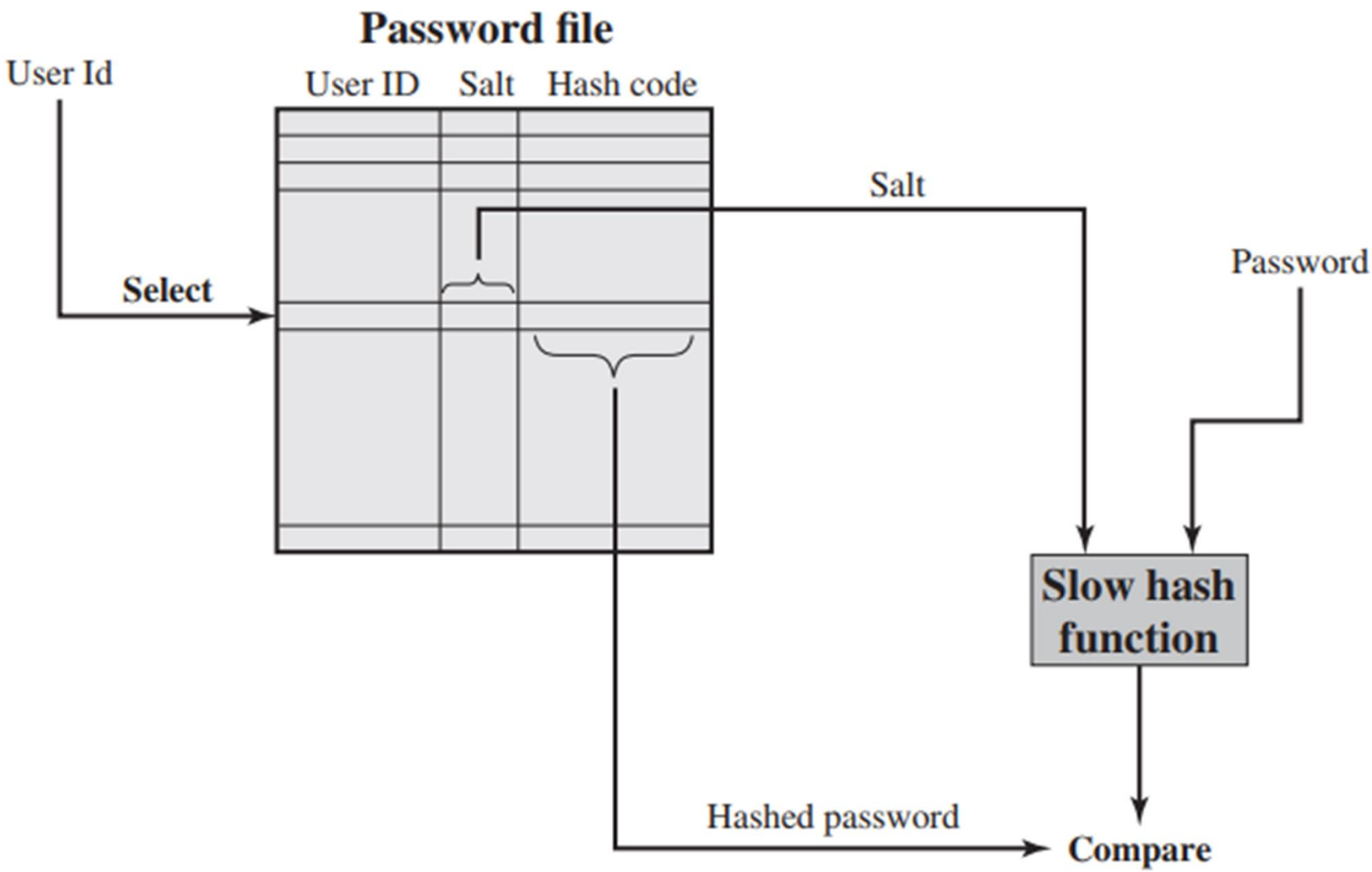
The Vulnerability of Passwords

1. Offline dictionary attack
2. Specific account attack
3. Popular password attack
4. Password guessing against single user
5. Workstation hijacking
6. Exploiting user mistakes
7. Exploiting multiple password use
8. Electronic monitoring

- A widely used password security technique is the use of hashed passwords and a salt value.
- To load a new password into the system, the user selects or is assigned a password. This password is combined with a fixed-length salt value.



(a) Loading a new password



(b) Verifying a password

The salt serves three purposes:

- It prevents duplicate passwords from being visible in the password file. Even if two users choose the same password, those passwords will be assigned different salt values. Hence, the hashed passwords of the two users will differ.
- It greatly increases the difficulty of offline dictionary attacks. For a salt of length b bits, the number of possible passwords is increased by a factor of 2^b,increasing the difficulty of guessing a password in a dictionary attack.
- It becomes nearly impossible to find out whether a person with passwords on two or more systems has used the same password on all of them.

NOTE:

In password file access control, the hashed passwords are kept in a separate file from the user IDs, referred to as a **shadow password file**.