

TSIT01 Datasäkerhetsmetoder

Lecture 7: Password handling, Operating system security,
Virtualization

Guilherme B. Xavier

Password handling

- The password system is a widely used line of defense against intruders
- Virtually every system where user login is needed requires the user to supply an ID and a password
- The system then checks its internal database to compare the supplied password to the stored one
- Access is then granted or denied

Password-based authentication

- The password is used to authenticate an user-ID. It grants security in the following ways:
 - The ID determines whether the user is authorized to gain access to the system
 - The ID determines the privileges the user has (Access control)

Password vulnerabilities

- Typically a system that employs password-based authentication has a password file indexed by user ID. Passwords are hashed (not in plain text!)
- Offline dictionary attack
 - Although the password file is typically protected by strong access control, an attacker may get access to it (elevation of privileges, SQL injection, etc...)
 - The attacker then compares the passwords hashes against hashes of commonly used passwords
 - Countermeasures include controls to prevent unauthorized access to the password file, intrusion detection measures to detect a compromise and blocking of current passwords until new ones are issued to the users

Password vulnerabilities

- Specific account attack
 - The attacker targets a specific account and submits password guesses until the correct password is discovered
 - The main countermeasure is an account lockout mechanism after a number of failed login attempts. Typically the lockout is enforced after 3-5 tries

Password vulnerabilities

- Popular password attack
 - A variation of the previous attack. The attacker chooses a popular password and tries it at several user IDs.
 - Here the countermeasures include policies to inhibit the selection of common passwords by the users, as well as scanning the IP addresses of login attempts.

Password vulnerabilities

- Password guessing against single user
 - The attacker attempts to gain knowledge about a specific user and system password policies and uses that knowledge to guess the password.
 - Countermeasures are mainly connected with policies enforcing the creation of secure passwords.
- Workstation hijacking
 - The attacker waits until a logged-in workstation is unattended.
 - The main countermeasure is automatically logging out after a period of inactivity. Intrusion detection schemes may be employed to detect changes in user behaviour.

Password vulnerabilities

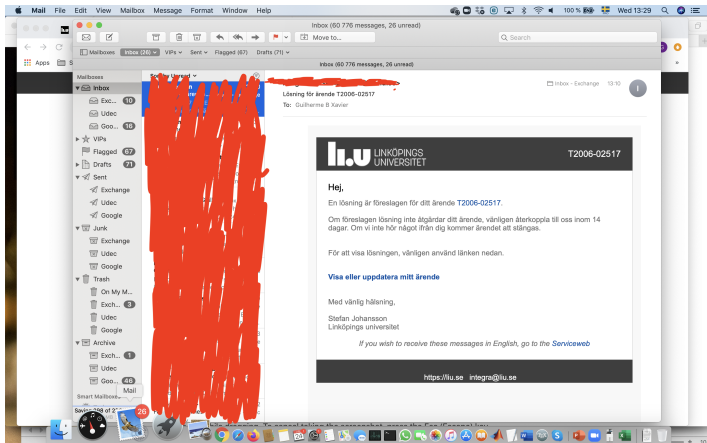
- Exploiting user mistakes
 - If the password is created by the system, the user is more likely to write it down because it may be difficult to remember
 - A user may intentionally share his password, to enable a colleague to do something on his behalf for example
 - Social engineering attacks may trick the user into revealing his password
 - Many computer systems are shipped with preconfigured passwords for system administrators
 - Countermeasures include user training, intrusion detection and simpler passwords combined with two-factor authentication

Password vulnerabilities

- Exploiting multiple password use
 - If the password is shared for a given user against many network devices, then a single compromise opens up many points of entry
 - The main countermeasure is a policy that forbids the reuse of passwords for the same user.

Password vulnerabilities

- Extracting passwords through social engineering



Passwords are nevertheless popular!

- Despite many security vulnerabilities, the use of passwords remain very popular. Reasons include:
 - Techniques that employ client-side hardware (fingerprint scanners) require the implementation of appropriate user authentication software on both the client and server systems (no longer a big problem).
 - Physical tokens such as smart cards are expensive and inconvenient to carry around.
 - Schemes that rely on a single sign-in to multiple services create a single point of security risk
 - Passwords can be changed/forgotten

Storing passwords

- You can't just store password in plain text!
- Solution: Don't store the password p , store its hash $H(p)$
- When a password p' is entered, compute $H(p')$ and compare with $H(p)$.
- For a secure hash function H , $p = p'$ implies $H(p) = H(p')$
- The hash function should be slow
- Dictionary attacks

Hash functions

- Hashing is a one-way cryptographic function that converts any input (like a password) into a fixed-length string of characters (the "hash").
- Hashing is deterministic: the same input always results in the same hash, but it is unfeasible to retrieve the original input from the hash.

SHA-256

- SHA-256 is not recommended because it is fast.
- Produces a fixed 256 bits output
- Common strong algorithms for password storage (slow) are bcrypt, PBKDF2 or SHA512crypt.
- The SHA family is used for digital signatures, data integrity and is central to Bitcoin and other blockchains.

Hashing passwords is not enough

- Hashes can still reveal identical passwords (i.e. Stephen and Sarah)

User	Password Hash
Stephen	39e717cd3f5c4be78d97090c69f4e655
Lisa	f567c40623df407ba980bfad6dff5982
James	711f1f88006a48859616c3a5cbcc0377
Harry	fb74376102a049b9a7c5529784763c53
Sarah	39e717cd3f5c4be78d97090c69f4e655

Another weakness of hashed passwords

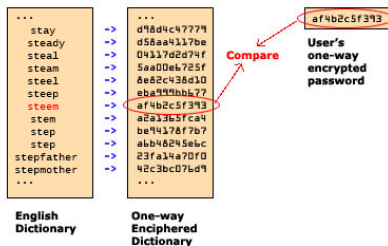
- For a given hash function, one can pre-compute common passwords
- This pre-computed database is known as a rainbow table
- This is a space-time tradeoff. More space = less time

DICTIONARY ATTACK!



Using a dictionary for attack

1. Prepare list of common words (i.e. English dictionary)
2. For each word w , compute $H(w)$
3. Store w and $H(w)$
4. Now compare hashes with known hashes



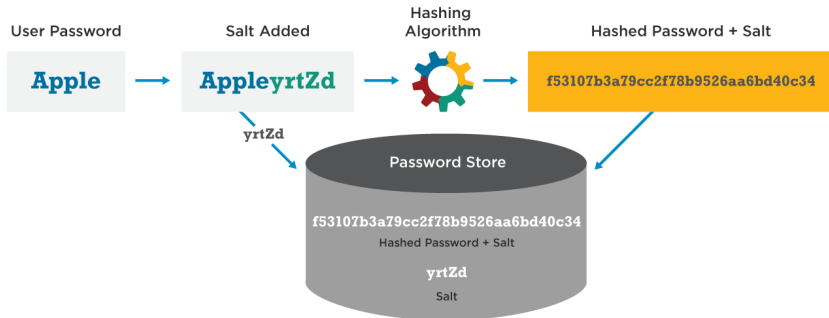
Improving things with salt

- Main problem: Hashes can be pre-calculated across users and systems
- Solution: Add a random, public, salt s before hashing
- Password database contains $H(p||s)$ and s
- When checking an entered password p' , compute $H(p'||s)$ and compare with database
- Salt should be unique for every user

Improving things with salt

- Advantages of adding a salt
 - 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
 - It greatly increases the difficulty of offline dictionary attacks. For a salt b bits long, the number of possible guesses increase by a factor of 2^b .
 - 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

Hashing and salting



Password hashes are now much harder to predict even when using common passwords

Intelligent password cracking

- The password file should always only contain the passwords hashes
- For example, the inputs “password”, “password1” and “Password” after being processed by the SHA1 algorithm, become respectively:
- 5baa61e4c9b93f3f0682250b6cf8331b7ee68fd8
- e38ad214943daad1d64c102faec29de4afe9da3d
- 8be3c943b1609ffffbfc51aad666d0a04adf83c9d

Intelligent password cracking

- Unfortunately things can be optimized for the attackers
- 6.5 is the average number of passwords for a web user (despite maintaining an average of 25 separate accounts)
- 8.2 billion password combinations per second can be tried on a PC with a single AMD Radeon HD7970 GPU (SHA1)
- 3108 Terabytes is the space needed for a table of every possible 10-character password with lowercase letters, along with its corresponding MD5 hash.
- 167 Gigabytes is the space needed for 99.9 of those passwords stored in a rainbow table (some optimizations)

Intelligent password cracking

- The leak of huge password files has helped the hackers learn the patterns of how people choose passwords
- The big break was the leak of 32 million plaintext passwords by an SQL injection attack against online games service RockYou.com
- This coupled with other leaks and new prioritized rules for which variations of a password to try first, made password cracking much more efficient.
- Now no longer the hackers had to guess what people might use, such as a first capital letter, or an extra “!”
- For example, just 6 days after the 2012 leak of 6.5 million LinkedIn password hashes more than 90% of them were cracked.

';--have i been pwned?

Check if you have an account that has been compromised in a data breach

etnoy@broach.se

pwned?

Oh no — pwned!

Pwned on 3 [breached sites](#) and found 1 [paste](#) ([subscribe](#) to search sensitive breaches)



[Notify me when I get pwned](#)



[Donate](#)





Dropbox: In mid-2012, Dropbox suffered a data breach which exposed the stored credentials of tens of millions of their customers. In August 2016, they forced password resets for customers they believed may be at risk. A large volume of data totalling over 68 million records was subsequently traded online and included email addresses and salted hashes of passwords (half of them SHA1, half of them bcrypt).

Compromised data: Email addresses, Passwords



Last.fm: In March 2012, the music website Last.fm was hacked and 43 million user accounts were exposed. Whilst Last.fm knew of an incident back in 2012, the scale of the hack was not known until the data was released publicly in September 2016. The breach included 37 million unique email addresses, usernames and passwords stored as unsalted MD5 hashes.

Compromised data: Email addresses, Passwords, Usernames, Website activity



Plex: In July 2015, the discussion forum for Plex media centre was hacked and over 327k accounts exposed. The IP.Board forum included IP addresses and passwords stored as salted hashes using a weak implementation enabling many to be rapidly cracked.

Compromised data: Email addresses, IP addresses, Passwords, Usernames



 GOT HASHES?

Meta pays the price for storing hundreds of millions of passwords in plaintext

Company failed to follow one of the most sacrosanct rules for password storage.

DAN GOODIN - 27 SEPT 2024 19:53  190

Protecting yourself against password leaks

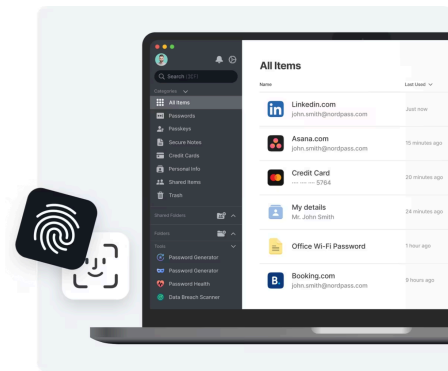


Figure: A password manager such as NordPass generates strong, unique passwords for each site

Protecting yourself against password leaks

TIME TO CRACK: MD5 Hashed Passwords

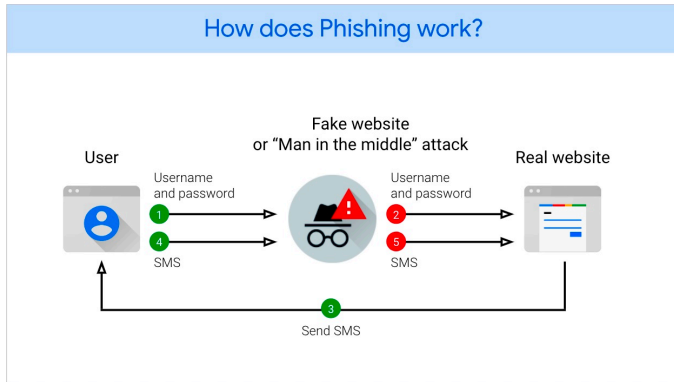


Number of characters	Numbers Only	Lowercase Only	Upper and Lower Case	Number, Upper, Lower	Number, Upper, Lower, Symbols
8	Instantly	Instantly	2 minutes	5 minutes	3 hours
9	Instantly	8 seconds	2 hours	5 hours	12 days
10	Instantly	4 minutes	2 days	14 days	3 years
11	Instantly	2 hours	132 days	3 years	279 years
12	Instantly	2 days	19 years	159 years	26.5 thousand years
13	Instantly	6 weeks	995 years	10 thousand years	3 million years
14	3 minutes	3 years	51 thousand years	608 thousand years	239 million years
15	26 minutes	82 years	2 million years	37 million years	22.7 billion years
16	5 hours	2136 years	140 million years	3 billion years	3 trillion years
17	43 hours	96 thousand years	8 billion years	145 billion years	205 trillion years
18	18 days	2 million years	379 billion years	9 trillion years	20 quadrillion years
19	6 months	38 million years	20 trillion years	557 trillion years	2 quintillion years
20	5 years	977 million years	2 quadrillion years	35 quadrillion years	176 quintillion years
21	49 years	26 billion years	54 quadrillion years	3 quintillion years	17 sextillion years
22	490 years	660 trillion years	3 quintillion years	133 quintillion years	2 septillion years

- Source: <https://specopssoft.com/blog/passphrase-best-practice-guide/>

Passkeys

- It is clear passwords have too many downsides.
- Even two-factor authentication (2FA) does not ensure full protection to phishing attacks

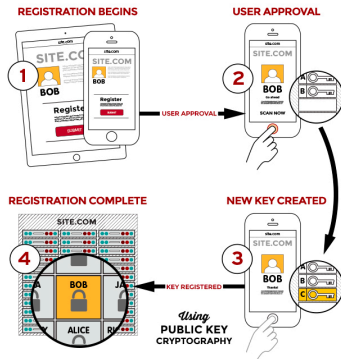


Passkeys

- The FIDO (Fast IDentity online) alliance has developed the framework of passkeys.
- Major industry partners are supporting and have already deployed it (Google, Microsoft, Apple).
- Passkeys provides secure authentication without the need to memorize passwords.
- It relies on using “something that you have” to create key pairs to each specific server.
- The private keys never leave your authentication device.

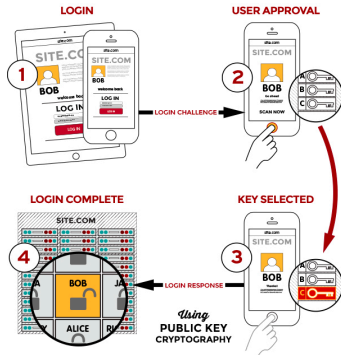
Passkeys

- First a key pair is created on the mobile device and the public key is registered to the server.
- This public key is stored on the server and associated to the user.



Passkeys

- The server sends a challenge (a random piece of data) to the mobile device.
- The device uses the private key to sign the challenge.
- The server uses the public key to verify the signature.



Benefits of passkeys

- **Phishing Resistance:** Since passkeys authenticate based on device-held keys, users can't be tricked into entering them on fake websites.
- **No Passwords to Remember:** Users don't need to create or manage passwords.
- **No Password Reuse:** Every account gets its own unique public-private key pair, eliminating the risk of credential reuse.
- **Secure Storage:** Private keys are securely stored in device-specific secure hardware, like the Secure Enclave on iPhones or Trusted Platform Module (TPM) on PCs.

Operating system security

- Computer client and server systems are central components of the IT infrastructure
- Hardening of operating system security is essential

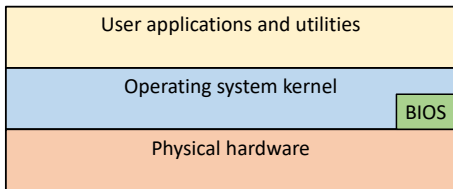


Figure: Operating system software layers

Operating system security

- A small number of hardening measures can prevent 85% of targeted cyberintrusions (source: Australian Signals Directorate):
 - Safe-list approved applications
 - Patch third-party applications
 - Patch operating system vulnerabilities and use the latest versions
 - Restrict administrative privileges

Operating system security

- Building and deploying a system should be a planned process
- NIST SP 800-123 (Guide to general server security) states that this process must:
 - Assess risks and plan the system development.
 - Secure the underlying operating system and then the key applications.
 - Ensure any critical content is secured.
 - Ensure appropriate network protections mechanisms are used.
 - Ensure appropriate processes are used to maintain security.

System security planning

- Careful planning will help ensure the new system is as secure as possible. On the other hand implementing security measures *after* the system is deployed is never good practice.
- What should be considered during the planning process (NIST SP 800-123):
 - The purpose of the system, type of stored information, applications and services provided, and their security requirements
 - Categories of users of the system, their privileges and types of information they can access
 - How users are authenticated
 - How access to the information stored on the system is managed
 - Who will administer the system and how (local or remote access)
 - Additional security measures (i.e. firewalls, anti-virus,...)

Operating system hardening

- The first critical step in securing a system is to secure the base operating system
- A good security foundation needs a properly installed, patched and configured operating system
- The default configuration for many OSs often maximizes ease of use and functionality rather than security.

Operating system hardening

- Basic steps as recommended by NIST SP 800-123:
- Install and patch the operating system
- Harden and configure and OS to adequately address the identified security needs by: removing unnecessary services and applications and configuring users, groups and permissions.
- Install and configure additional security controls, such as anti-virus, firewalls and IDSs if needed
- Test the security of the basic operating system

OS installation: Initial setup and hardening

- A network-connected unpatched system is vulnerable
- Ideally new systems should be constructed on a protected network.
- The boot process must also be secured.
- Care is needed when installing additional device drivers (provided by a third party).
- Use automated patching (unless in mission-critical systems).

OS installation: Initial setup and hardening

- A network-connected unpatched system is vulnerable
- Ideally new systems should be constructed on a protected network.
- The boot process must also be secured.
- Care is needed when installing additional device drivers (provided by a third party).
- Use automated patching (unless in mission-critical systems).
- In that case stage and validate on test systems before deploying in production

Application security

- Once the base OS is installed and appropriately secured, the required services and applications must then be installed and configured.
- Servers should only have the bare minimum needed to run (i.e. no office productivity tools).
- On server systems software that provides remote access or service, including Web ,database and file access servers is of particular concern.
- On client systems, software such as Java, PDF viewers, Flash, web browsers and Microsoft Office are known targets and need to be secured.

Security maintenance

- Once the system is appropriately built, secured and deployed, the process of maintaining security is continuous.
- Monitoring and analyzing logging information; performing regular backups; recovering from security compromises; Regularly testing system security; Patching regularly.

Logging

- Logging is a reactive control that can only inform you about things that have already happened.
- But effective logging can help system admins to more quickly and accurately identify what happened.
- It helps to detect that attacks are happening (this may not be directly visible at all).
- Logging can generate significant volumes of information
- Manual analysis of logs is normally impractical, but IDSs perform automated analysis.

Data backup and archive

- Regular backups of data is a critical control that assists in maintaining the integrity of the system and data.
- There may also be legal or operational requirements for the retention of data.
- Backup is the process of making copies of data over relatively short time periods of a few hours to some weeks.
- Archive is the process of retaining copies of data over extended periods of time (months or years), in order to meet legal and operational requirements to access past data.
- Key decisions include whether the backup copies are kept online or offline and whether copies are stored locally or transported to a remote site.

Linux security

- Linux is free, open-sourced and available in a wide variety of distributions, targeted at every usage scenario imaginable.
- Examples include everything from supercomputers, servers to client and embedded systems.
- Linux employs a discretionary access control security model (privileges can be handed out).

Linux security model

- The basic premise is: People or processes with “root” privileges can do anything: other accounts can do much less.
- From the attacker’s perspective the challenge boils down to gaining root privileges.
- In spite of a very simple security model, Linux can be very secure but needs proper and careful configuration.
- The relatively few number of attacks targeting Linux is not really because it is more inherently secure than other modern OSs. There are much less Linux users, and also they are on average much more “tech-savvy”.

Linux security model

- Each user in Linux belongs to one or more groups.
- Users can read, write and execute objects (files and directories), depending on each objects' permissions.
- Each object has three sets of permissions: owner, group and “others”.
- The Linux kernel enforces these permissions.

Linux security model

- When process/programs are executed, permissions are used twice in the process.
- First, a program's file restrictions restrict those who can execute, access or change it.
- Then when running, a process normally runs as (with the identity of) the user and group of the person or process that executed it.

Linux security model

- Whoever owns an object, can set or change its permissions.
- The system superuser account, called root, has the ability to both take ownership and change the permissions of all objects in the system.
- A process executed by the root user, will also have root permissions.

Limitations to access control in Unix

- Files have only one owner and one group
- Permissions are read, write, and execute
- All other access rights must be mapped to basic file permissions
- Other operations need to be done through `suid` applications (setting the owner's privileges to the application run by a different user)
- Complex security policies are often impractical

Windows

- Windows is the world's most popular OS.
- The vast majority of users has little technical knowledge, opening up more challenges in security.
- Windows based on the Windows 95 code base (95, 98, 98 SE and Me) had no security model.
- All current Windows versions are based on the NT code base.

Hardening Windows

- For backwards compatibility reasons, Windows XP users usually had to log as local Administrators, with all the obvious security issues.
- From Windows Vista, by default user accounts were not administrators. When a user wants to perform a privileged operation, admin credentials need to be provided.
- From Windows Vista onwards, IPv6 is enabled by default (authentication)
- All versions of Windows since XP have included a built-in software firewall. From XP SP2, the firewall is enabled by default.

Formal models and Windows

- Windows provides more fine-tuned access control
- But the same critique holds; there is no check that the levels really form a hierarchy
- Although Windows User Access Control (UAC) attempts to implement Biba

Bluekeep

- Users of Windows Server 2008 and Windows 7 and earlier affected
- Bluekeep was a critical vulnerability in remote desktop services.
- Did not require user interaction and is “wormable”.
- Was exploited mostly for cryptocurrency mining.



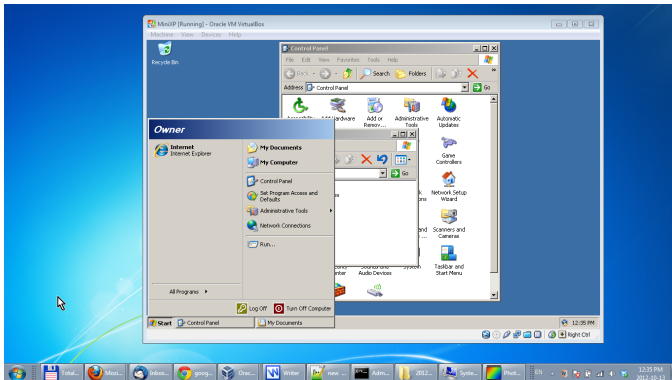
Nearly 1 Million Systems Affected By 'Wormable' BlueKeep Vulnerability (CVE-2019-0708)

May 29, 2019



Virtualization

- In the past decade, virtualization has become very popular
- A *virtual machine* (VM) is a simulated computer, which in turn runs an operating system

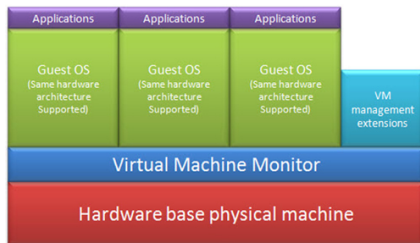


The virtual revolution

- Virtualization took off more than a decade ago
- In 2020: 95% of US companies use server virtualization, with lower percentages for other areas such as storage, application, etc...
(<https://www.spiceworks.com/marketing/reports/state-of-virtualization/>)

Full virtualization

- There are many forms of virtualization, with different architectures
- We will cover *full virtualization*, where the guest OS has no idea it is ran on a VM
- Hypervisor or Virtual Machine Monitor takes care of managing the guests



Virtualization and security

- Virtualization allows a higher degree of isolation compared to chroot
- One compromised guest system typically does not compromise the host
- However, virtual machines can be complex and hard to analyze
- Virtualization can also increase flexibility, making systems easier to administer

Virtualization reduces administrative burden

- A new OS can be installed, configured, secured and tested in a VM
- The administrator then takes a snapshot of this VM
- This snapshot can then be distributed to many hosts
- If a guest system is compromised, it can be frozen (including RAM contents), which makes forensic analysis much easier
- A virtual machine can be migrated between hosts (often while running!)
- This makes it less of a hassle to install patches and reboot the host server

The Virtual Zoo

Virtual machine escape

- The most serious type of attack
- Usually grants privileged access to the host machine
- ... which of course compromises all guests on that host
- Often caused by bugs in the hypervisor
- Example: Cloudburst attack on VMWare in 2009

Virtualization paved the way for cloud computing

- Virtual Private Server (VPS): A virtual machine sold as a service
- VPS allows customers root access within their VPS
- Example: Amazon AWS provides the Elastic Compute Cloud (EC2), where VPS instances can be managed and created
- EC2 can be rented in the following ways:
 - On-demand (hourly rates)
 - Reserved (long-term rental)
 - Spot (bid-based, runs jobs only if the spot price is below the bid price)
- Future trend: Centralization of computing