Risk Assessment

Objective: Identify and mitigate security risks on a LAMP server to ensure GDPR compliance and protect sensitive data stored on it.

Scope *:* Ubuntu 22.04.2 Server system with a basic configuration for future services. The server is currently not ready for production. It has a good basic configuration, but lacks protection against common and automated attacks, exposing it to potential compromises that could affect business continuity.

This document will be subject to review every three months or following the implementation of the services on the system.

Probability and Impact Assessment Criteria:

Chance
1 (Rare): Never happened, extremely unlikely to happen in the next year (<5% chance).
2 (Unlikely): Could happen under exceptional circumstances (5-20% annual probability).
3 (Possible): It could happen (20-50% annual probability).
4 (Probable): Will almost certainly happen (50-80% annual probability), or has happened in the past.
5 (Very Likely): It will almost certainly happen either more often (80-100% annual probability), or it is a common occurrence.

Impact:
1 (Insignificant): No detectable impact.
2 (Low): Minimal disruption, low financial costs, no reputational damage, compliance maintained.
3 (Medium): Local/temporary disruption, potentially significant costs, minor reputational damage, potential minor compliance breach.
4 (High): Extensive/significant disruption, high costs, significant reputational damage, serious compliance breach with potential penalties.
5 (Catastrophic): Total business disruption, loss of critical data, very high costs, irreparable reputational damage, severe legal penalties, potential bankruptcy.

Risk assessment criteria:

Risk is calculated with a formula Probability x Impact

Low (1-7):	Minimal risk, to be mitigated within three months.
Medium(8-12):	Moderate risk, to be mitigated within a month.
High(13-19):	High risk, to be mitigated within a week.
Critic (20-25):	Critical risk, to be mitigated immediately.

Hence the following Heat Map:

Risk table

Risk ID	Asset/Vulnerability	Threat Actor & Attack Vector	Probability (1-5)	Impact (1-5)	Risk Score	Risk Level	Recommended action
RISK-001	Firewall Configuration	Anyone can perform a port scan and attack any exposed service without limitations.	5	5	25	CRITIC	Configure the firewall to allow only the actions necessary for the required services to function
RISK-002	Backups	No scheduled backups, no recovery plan	4	5	20	CRITIC	Implement a robust backup system and test for integrity.
RISK-003	Protection against common attacks on web services	SQL Injection, Cross-Site Scripting (XSS), and other attacks	4	4	16	HIGH	Install a WAF and set more restrictive input configurations
RISK-004	Password Setup	Anyone with access can see the data inside the database in plain text or access web services.	3	5	15	HIGH	Add different, secure passwords for accessing each service

RISK-005	SSH Configuration	Standard configurations increase an attacker's chance of success	3	5	15	HIGH	Disable root access from SSH. Set rules for allowed addresses, use an elliptical key, and change the default port.
RISK-006	File integrity	An attacker's changes intended to hide them may go undetected.	3	5	15	HIGH	Set up a hash checking system to detect unauthorized changes
RISK-007	Information gathering	Incomplete logs may not detect successful or attempted attacks.	3	4	12	MEDIUM	Configure kernel auditing and logging
RISK-008	Installed Services	Possible exploitation of an obsolete or unused service but installed as a dependency or by mistake	3	4	12	MEDIUM	Conduct a census of essential services and dependencies, uninstalling anything that is not strictly necessary to reduce the attack surface
RISK-009	Login Configurations	No limits on brute-force or password complexity	3	4	12	MEDIUM	Add rules to limit the number of password attempts, add a strong password rule, and implement an expiration to force password renewal.
RISK-010	DoS Protection	An external attacker could try to fill the /var and /tmp folders or flood the web service with requests causing the entire system to fail.	4	3	12	MEDIUM	Limit the number of packages running on a single host or concurrently, and consider installing specific software for web applications. Move the /var and /temp folders to a dedicated partition.
RISK-011	Installing packages	A trusted package may have been compromised or have bugs	4	3	12	MEDIUM	Install a packet control system
RISK-012	GRUB boot loader	An attacker could log in as root from boot without needing to enter any password if the host is compromised.	2	5	10	MEDIUM	Set a boot password to prevent single user mode from being used
RISK-013	Core dump	The core dump is a file that may contain sensitive data that can be viewed in clear text.	3	3	9	MEDIUM	Disable core dump and evaluate an external logging system on a dedicated server
RISK-014	Listening doors	Possible attacks on known ports via bots or targeted attacks	3	3	9	MEDIUM	Make sure that only the necessary ports are open, and that they are well protected.
RISK-015	Active processes	A listening process could be exploited maliciously	3	3	9	MEDIUM	Check the active processes on the server, remove or limit obsolete or unnecessary listening processes (such as snap)
RISK-016	Accessible pages	Test html or php files can reveal confidential information that can be exploited for other attacks	3	3	9	MEDIUM	Move or delete any page that is accessible from the web and is not in the public domain
RISK-017	Logs management	No backup in case an attacker deletes logs of their malicious actions	3	3	9	MEDIUM	Setting up a real-time logging system on an external server

RISK-018	Users	A large number of users with broad permissions exposes the machine to more risks	2	4	8	MEDIUM	Review the user list and disable unnecessary ones. Manage permissions for each user, limiting possible actions to those necessary for the proper functioning of the services.
RISK-019	Compiler permissions	A user with limited permissions could exploit compilers to elevate privileges	2	4	8	MEDIUM	Restrict access to compilers
RISK-020	Network protocols	Every enabled but unused protocol is a potential attack vector	2	4	8	MEDIUM	Disable unnecessary protocols
RISK-021	New file permissions	The umask is too restrictive, allowing new files to be created and accessed too easily.	2	3	6	BASS	Restrict permissions on new files
RISK-022	External device drivers	An attacker could mount a malicious external USB on the system	1	5	5	BASS	Disable USB drivers and similar
RISK-023	Physical access	Unauthorized physical access to the server	1	5	5	BASS	Lock the door, set passwords for the host system and boot
RISK-024	Cleaning old files	Old, unused files and packages may hide vulnerabilities or contain sensitive data.	2	2	4	BASS	Automate the removal of obsolete files

---

Risk Treatment Plan

Risk-001

Risk Level: Critic
Treatment: Mitigate

A server without a firewall configuration is constantly exposed to automated attacks, which according to the latest estimates hit each internet-facing server an average of every 39 seconds, according to 2025 telemetry from security vendors. The average time between exposure and the first exploitation attempt has decreased to 14 hours in the last 12 months.
The firewall is already present on the system; the default policy configuration suggests blocking all incoming traffic and allowing outgoing traffic. Before doing this, you need to add an SSH rule to prevent lockouts. My suggestion is to temporarily allow incoming connections on port 22 (given the current and standard SSH configurations) and on a new port that will be used for SSH connections (for example, 2222).

_____
These commands allow incoming connections to port 2222 with TCP protocol and to port 22 read by current SSH configurations:
```
sudo ufw allow 2222/tcp comment 'SSH new port'
sudo ufw allow ssh
```

_____
These are the default firewall settings that deny all incoming connections:
```
sudo ufw default deny incoming
sudo ufw default allow outgoing
```

_____
Allows connection to the installed Apache server by opening ports 80 (http) and 443 (https):
```
sudo ufw allow 'Apache Full'
```

_____
Verifying your changed settings before applying them is a good security practice:
```
sudo ufw show added
```

_____
Apply the new firewall rules:
```
sudo ufw enable
```

_____
Verify that they are active and once:
```
sudo ufw status numbered
```

_____
Once you have verified that both ports 22 and 2222 are open then you can change the SSH configuration (RISK-005) and restart the connection to connect on the new port.
```
sudo systemctl reload ssh
```

_____
After verifying that the connection was established on port 2222 instead of the standard 22, remove the rule that allowed listening on the old port:
```
sudo ufw delete allow 22/tcp
```

_____
_____

Risk-002

Risk Level: Critic
Treatment: Mitigate

A server without a robust backup system violates the GDPR and is a serious security breach. Implementing a backup system is therefore imperative to ensure server uptime and regulatory compliance.
The proposed backup solution is three-tiered, compliant with the 3-2-1 principle, meaning three copies of the data on two different devices with one off-site copy. The strategy also includes automated daily backups using dedicated scripts to ensure transactional consistency, routine inclusion, and optimized storage management through compression.
The mitigation plan for this risk includes installing several tools to simplify backup management, ensure its integrity, and automate it. Specifically:

`rsync` is essential for efficient remote backup synchronization, ensuring synchronization and incremental backup to save resources, and should be configured with an SSH connection.

`mysql-client` which contains the tools needed to administer a SQL server.

`cron` handles the automation of backup operations, minimizing human error and speeding up the process.

`logrotate` for efficient log management

`gzip` and `tar` for compressing logs and backups

`mailutilis` enables sending email notifications for monitoring

They can all be installed via `apt install`

The first step is to create the backup directories and subdirectories with the command
`sudo mkdir -p /backup/{database,web,system,logs}/{daily,weekly,monthly}`

Followed by applying more restrictive permissions
`sudo chmod -R 755 /backup && sudo chown -R root:root /backup`

Note : If you opt for an even more secure solution, you could implement specific user groups for managing backups and monitoring via service scripts.

Next, create the SSH encryption key in a folder outside the backup folders. The following command will create a backup-key storage folder that can be used by automation scripts running with elevated privileges.

`sudo mkdir -p /etc/backup-keys && sudo openssl rand -base64 32 > /etc/backup-keys/backup.key && sudo chmod 400 /etc/backup-keys/backup.key && sudo chown root:root /etc/backup-keys/backup.key`

Note: For greater security, consider managing keys via a dedicated server or splitting keys using a system that uses secret sharing techniques.

_____

After verifying the previous steps, you can move on to the actual backup management, which I recommend implementing via scripts with 755 permissions, incorporating error handling logic, detailed logging, and integrity verification via 256 hash. In particular, I suggest:

- Script that creates the backup with a logging function to ensure regulatory compliance, a pre-check of the encryption key, use of mysql-client to create the backup files, SHA-256 hash encryption to verify that the backups are not corrupted, and automatic retention to comply with the principle of data minimization.
- Separate script to backup Apache configurations, web document root, system configurations, clean temporary files and checksum for each of them.
- Script for remote backup using rsync over SSH, after configuring the remote server.
- Integrity and recovery test script to be able to check the checksum and try to decrypt backups in a test folder and delete it afterwards
- Script for monitoring and tracking with control over the backup creation date, automatic email alerts in case of failure or problems and disk space occupied by backups.
- Maintenance and cleanup scripts to automate data retention in compliance with GDPR, including sophisticated logic that takes into account different retention policies for different data types and creates log files to record the deletion of old backups.

_____

The final step to complete the automation of your backup system involves configuring cron via `sudo crontab -e` with time-distributed scheduling, such as database backup at 2:00 AM, filesystem backup at 3:00 AM, remote sync at 4:00 AM, and cleanup at 5:30 AM to ensure that the previous night's backups are completed before the cleanup operation.

Risk-003

Risk Level: High
Treatment: Mitigate

Web service attacks represent a major threat to internet-facing LAMP servers. According to OWASP 2023 data, injection (including SQL injection) remains the top critical vulnerability, followed by cross-site scripting (XSS) and security misconfigurations.
I recommend implementing a WAF, or web application firewall, such as modsecurity, which is open source and offers a variety of custom configuration options, as well as being the most popular for Apache servers.
Install the module with
`sudo apt install libapache2-mod-security2`

_____
Enable the newly installed module in your Apache instance
sudo a2enmod security2

_____
The module is downloaded with recommended configurations that can be used as a basis for the main WAF guidelines. I recommend using the template as a basis and applying modifications to it.
sudo cp /etc/modsecurity/modsecurity.conf-recommended /etc/modsecurity/modsecurity.conf

____
Once you have copied the basic configuration file, open it with sudo nano /etc/modsecurity/modsecurity.conf and make the following changes:

SecRuleEngine On: Activates the rules engine
SecAuditEngine On: Enables audit logging to track all events
SecRequestBodyLimit 10485760: Limit request size to 10MB to prevent DoS
SecResponseBodyLimit 524288: Limit responses to 512KB to optimize performance.

_____
The next step is to implement custom rules for the correct functioning of the WAF, to do this you must first create the folder that will contain them
sudo mkdir -p /etc/modsecurity/custom-rules

_____
And include the folder and the basic configuration in the configuration file with
sudo nano /etc/apache2/mods-enabled/security2.conf

_____
Now we'll install ModEvasive to protect the server specifically against DoS and brute force attacks by limiting the number of requests per IP in a given period of time. This module must also be enabled on Apache.
sudo apt install -y libapache2-mod-evasive
sudo a2enmod evasive
And then configured with the following changes
sudo nano /etc/apache2/mods-enabled/evasive.conf

DOSPageCount/DOSPageInterval: Limit requests per single page
DOSSiteCount/DOSSiteInterval: Site-wide request limit
DOSBlockingPeriod: Duration of temporary IP blocking
DOSWhitelist: IPs excluded from scanning (local networks)

_____
Continuing the hardening of the Apache service the next step is to configure the security headers
sudo nano /etc/apache2/conf-available/security-headers.con f

Edit the rules file for:

- XSS Prevention
- Clickjacking Prevention
- MIME sniffing prevention
- Basic Content Security Policy
- HSTS for HTTPS
- Hide server information

And apply them with
`sudo a2enconf security-headers`
`sudo a2enmod headers`

_____
The same module can also be used to block lateral movement by modifying the settings in
sudo nano /etc/apache2/conf-available/anti-traversal.conf

With rules for:
- Directory traversal protection
- Block access to sensitive files
- Block access to specific directories

which must then be activated with
`sudo a2enconf anti-traversal`

_____
Additional custom rules against XSS and SQL Injection can be created and added to the folder
`/etc/modsecurity/custom-rules`

Note: After evaluating the implementation and management time of exclusions, it has been decided not to apply the CRS (Core Rule Set) for now, however, future implementation of this open ruleset managed by global cybersecurity experts is strongly recommended.

_____
The implementation of this solution also includes comprehensive logging to monitor in real time any configuration errors or blocked attacks and manageable via logrotate in `/var/log/apache2/modsec_audit.log`

_____

Risk-004
Risk Level: High
Treatment: Mitigate

The lack of specific passwords for each service increases the likelihood of unauthorized access and easily exposes all services to a potential attacker who has gained access to the server .
Setting passwords to protect MariaDB and Apache is therefore strongly recommended.

To set a password for the MariaDB service, run the following command and follow the steps on the terminal to create the password, taking care to disable remote root login

```
sudo mysql_secure_installation
```

_____

Set a password for Apache as well using the Apache Server Utilis tool.

```
sudo apt install apache2-utils
```

Create the directory that will contain the authentication file

```
sudo mkdir -p /etc/apache2/auth
```

Create the password file (.htpassword) for accessing the control panel

```
sudo htpasswd -c /etc/apache2/auth/.htpasswd admin
```

Enter a strong password when prompted and then apply the configurations to the Apache server by first creating a configuration to protect the affected directories with

```
sudo nano /etc/apache2/conf-available/auth-directories.conf
```

Enable modules and configuration

```
sudo a2enmod auth_basic
sudo a2enmod authz_user
sudo a2enconf auth-directories
```

_____

In general, it is recommended to use passwords of at least 16 characters, with 4 character classes (a-z, A-Z, 0-9, symbols), different for each service and avoiding using words found in the dictionary.

Risk-005

Risk Level: High
Treatment: Mitigate

Standard SSH configurations significantly increase an attacker's chance of success. The server is exposed on port 22 with default configurations that facilitate brute force attacks and system compromise.

It is good practice to back up critical configurations in case of errors so that they can be restored promptly.

```
sudo cp /etc/ssh/sshd_config /etc/ssh/sshd_config.backup.$(date +%Y%m%d_%H%M%S)
```

_____

Create an elliptical authentication key pair (Ed25519) for greater security than traditional ones but without significantly impacting performance

```
sudo mkdir -p /etc/ssh/hardened_keys
sudo ssh-keygen -t ed25519 -f /etc/ssh/hardened_keys/ssh_host_ed25519_key -N ""
sudo chmod 600 /etc/ssh/hardened_keys/ssh_host_ed25519_key
```

```
sudo chmod 644 /etc/ssh/hardened_keys/ssh_host_ed25519_key.pub
```

_____

Creating a warning banner to inform that the system is restricted to authorized users, provides a legal warning and may have value in any legal proceedings

```
sudo nano /etc/ssh/ssh_banner.txt
```

_____

Open the SSH configuration file

```
sudo nano /etc/ssh/sshd_config
```

And make the following changes:

- Change door from 22 to 2222
- Disable root login
- Access restrictions for authorized users only
- Specify the path for the elliptical keys
- Set general hardening solutions such as a limit on authentication attempts, a limit on the number of concurrent SSH sessions allowed, and a time limit between attempts to avoid brute force attempts.
- Disable unnecessary features
- Evaluate IP restriction
- Enable verbose logging
- Disable blank passwords
- Adding paths to banners with legal value

_____

Check the correctness of the configuration syntax

```
sudo sshd -t
```

_____

If you haven't already, add a rule to your firewall to allow the new port

```
sudo ufw allow 2222/tcp comment 'SSH hardened port '
```

_____

Restart the SSH service while keeping the session active

```
sudo systemctl reload ssh
sudo systemctl status ssh
```

_____

After verifying that port 2222 is working, remove the rule for port 22 that is no longer in use.

```
sudo ufw delete allow 22/tcp
```

_____

For additional protection, I recommend installing fail2ban to automatically block requests from IPs that repeatedly fail login attempts.

```
sudo apt install fail2ban
```

Change the settings to secure SSH on the new port, including setting a path for saving logs.

```
sudo cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local
sudo nano /etc/fail2ban/jail.local
```

_____

Restart fail2ban after changing configurations and verify.

```
sudo systemctl enable fail2ban
sudo systemctl start fail2ban
sudo fail2ban-client status sshd
```

Risk-006

Risk Level: High

Treatment: Mitigate

An attacker's changes to system files can remain hidden without an integrity checking system. Implementing a hash monitoring system will allow for immediate detection of unauthorized changes to critical system files.

A workaround is to create two scripts using native Linux tools like sha256sum to create hashes of critical files and check them.

Create the work directory

```
sudo mkdir -p /var/security/integrity/{baseline,current,reports}
sudo chmod 750 /var/security/integrity
sudo chown root:root /var/security/integrity
```

_____

Create the baseline script, it serves as a check base with the initial hashes

```
sudo nano /usr/local/bin/ create-integrity-baseline.sh
```

_____

Add the necessary permissions to the script and run it, this will create a file with the SHA265 hashes of all critical files on the system

```
sudo chmod +x /usr/local/bin/create-integrity-baseline.sh
sudo /usr/local/bin/create-integrity-baseline.sh
```

_____

Create a second script for checking, the script will generate new hashes based on the current files and compare them with the previously created ones, saving the whole process in a log file

```
sudo nano /usr/local/bin/ check-integrity.sh
sudo chmod +x /usr/local/bin/check-integrity.sh
sudo /usr/local/bin/check-integrity.sh
```

_____

Set up the check script with crontab (previously installed) for automation.

```
sudo crontab -e
```

Note: This is a temporary solution, I strongly recommend switching to a configuration that uses dedicated tools like AIDE for real-time monitoring, advanced configurations and exclusions to avoid false positives (data from log folders for example), multiple hash algorithms, greater speed and scalability.

Risk-007

Risk Level: Medium
Treatment: Mitigate
The absence of a kernel auditing system could prevent the detection of failed or successful attack attempts, unauthorized system modifications, privilege escalations, unauthorized access, and regulatory violations.

Install auditd, start it, and enable it to listen
```
apt install auditd audispd-plugins
systemctl start auditd
systemctl enable auditd
```

_____
The core functionality of auditd is its rules, which define the events to log. Custom rules should be placed in files with the .rules extension in the /etc/audit/rules.d/ directory. A set of rules, inspired by the Center for Internet Security (CIS) security standards, includes: setting immutability; monitoring of auditd configuration files; access to files critical to identity and security; login and logout events; commands that modify user and group information; file permission alterations; and unauthorized file access attempts .
To load the newly created rules
```
augenrules --load
```

_____
Check current status
```
sudo systemctl status auditd
sudo auditctl -s
```

Note: auditd creates log files in the var/log/audit/ folder that can be managed with logrotate. It is also recommended to add a log control script linked to an alerting system with mailutilis and automation with crontab.

Risk-008

Risk Level: Medium
Treatment: Mitigate
The server has numerous services and packages installed (packages.txt in the attachments) that may not be necessary for the required functions. The presence of unused software represents an expanded attack surface, where an attacker could exploit vulnerabilities in obsolete or unmaintained services. The package inventory reveals the presence of hundreds of packages, many of which may not be essential to the operation

of a LAMP server. Reducing the attack surface is a fundamental principle of cybersecurity: every installed software component represents a potential vulnerability vector.

Deep Dependency Analysis with
```
apt-cache depends apache2 mariadb-server php php-mysql | grep "Depends:" | awk '{print $2}' | sort | uniq > /tmp/lamp_dependencies.txt
```

_____
Once the packages essential to the server's operation have been detected, move on to identifying obsolete packages by comparing the lamp_dependencies.txt file with packages.txt.
Also use the apt command to locate obsolete packages.
```
sudo apt autoremove --dry-run
```

And for a check on manually installed packages
```
apt-mark showmanual | sort > /tmp/manual_packages.txt
```

_____
Disable non-essential services such as snap, to deactivate the service (substitute the name of the service you want to remove in the command)
```
sudo systemctl disable snapd.service
sudo systemctl stop snapd.service
sudo systemctl mask snapd.service
```

_____
Consider creating a script to check for orphaned packages and dependencies, with timestamped logs and automation via crontab. Installing a specialized tool for identifying orphaned packages, such as deborphan, is also recommended.

Risk-009

Risk Level: Medium
Treatment: Mitigate
Setting passwords isn't enough to ensure security; passwords must be sufficiently complex and periodically renewed. I recommend installing the PAM module to create a robust password policy that not only focuses on initial secrecy but also takes into account its long-term validity.

Install the necessary module
```
apt install libpam-pwquality
```

_____
Change the settings to set: minimum password length, presence of at least one number; one lowercase letter; one uppercase letter; one symbol, set number of characters that must be different from the old password to discourage the use of incremental passwords
```
nano /etc/security/pwquality.conf
```

_____
Set password expiration times and apply the rules retroactively to existing users. To do this, open the expiration configuration file and set a maximum and minimum password expiration time, as well as a number of days' warning before password expiration.

```
nano /etc/login.defs
```

Apply changes retroactively to existing users with a for loop (vary the parameters depending on the chosen configuration)

```
for user in $(awk -F: '$3 >= 1000 {print $1}' /etc/passwd); do
echo "Applying expiration policy to user: $user"
chage -M 90 -m 7 -W 14 "$user"
done
```

Risk-010

Risk Level: Medium

Treatment: Mitigate

This represents a system vulnerability where an external attacker could compromise service availability through: Disk space saturation: Filling the /var and /tmp directories with large temporary files or logs, Request flooding: Flooding the web server with simultaneous requests to exhaust system resources, Resource exhaustion: Causing a denial of service that would render the system unusable. For protection against DoS attacks on the web server, see the mitigations in Risk-003.

Configure additional system-level protections using iptables to restrict kernel-level connections for in-depth security. As always, it's good practice to back up your current configurations in case of errors.

```
sudo iptables-save > /backup/iptables-backup-$(date +%Y%m%d).rules
```

Add rule to limit simultaneous HTTP and HTTPS connections to 25 per IP

```
sudo iptables -A INPUT -p tcp --dport 80 -m connlimit --connlimit-above 25 -j REJECT --reject-with tcp-reset
sudo iptables -A INPUT -p tcp --dport 443 -m connlimit --connlimit-above 25 -j REJECT --reject-with tcp-reset
```

First rule marks new connections, second rule blocks IPs that make more than 20 connections in 60 seconds

```
sudo iptables -A INPUT -p tcp --dport 80 -m state --state NEW -m recent --set --name HTTP_FLOOD
sudo iptables -A INPUT -p tcp --dport 80 -m state --state NEW -m recent --update --seconds 60 --hitcount 20
--name HTTP_FLOOD -j DROP

sudo iptables -A INPUT -p tcp --dport 433 -m state --state NEW -m recent --set --name HTTP_FLOOD
sudo iptables -A INPUT -p tcp --dport 433 -m state --state NEW -m recent --update --seconds 60 --hitcount 20
--name HTTP_FLOOD -j DROP
```

Save the new rules

```
sudo iptables-save
```

_____

I recommend separating the /var and /tmp directories onto dedicated partitions to prevent flooding from causing a denial of service for the entire system. After calculating the space occupied by the folders and the total available space, proceed with creating the partitions (with loop files if no additional disks are present).

```
sudo dd if=/dev/zero of=/var-partition.img bs=1M count=5120
```
5GB for /var

```
sudo dd if=/dev/zero of=/tmp-partition.img bs=1M count=2048
```
2GB for /tmp

Format new partitions with ext4

```
sudo mkfs.ext4 /var-partition.img
sudo mkfs.ext4 /tmp-partition.img
```

Add partitions to the File System Table in /etc/fstab after backup

```
sudo cp /etc/fstab /etc/fstab.backup.$(date +%Y%m%d)
echo '/var-partition.img /var ext4 loop,defaults 0 2' | sudo tee -a /etc/fstab
echo '/tmp-partition.img /tmp ext4 loop,defaults,noexec,nosuid,nodev 0 2' | sudo tee -a /etc/fstab
```

_____

Advanced anti-DoS configurations at kernel level, creating a configuration file for network parameters containing: protection against SYN flood, connection timeout reduction, protection against ICMP flood, rate limiting for connections

```
sudo nano /etc/sysctl.d/99-anti-dos.conf
```

And apply it

```
sudo sysctl -p /etc/sysctl.d/99-anti-dos.conf
```

_____

Add a specific logging system that tracks DoS attempts by integrating logs from iptables and ModEvasive (Risk-003) and integrate Fail2Ban with specific filters for HTTP DoS attacks (restart fail2ban to apply the changes)

_____

Consider creating a script for real-time monitoring, with timestamps, active and blocked connections, latest events, and disk and partition space monitoring to ensure business continuity.

Risk-011

Risk Level: Medium
Treatment: Mitigate

Every time new software is installed on a system, it potentially exposes it to vulnerabilities that could be exploited by attackers. These include compromised packages at source (supply chain attacks are increasingly common) or vulnerabilities unknown at installation time. I recommend implementing a package integrity check system and automated security update management.

Install additional monitoring tools to those already present
```
sudo apt install -y debsums apt-listchanges unattended-upgrades needrestart
```

_____

It is good practice to keep control structures in separate folders (as done for other risks) with restrictive permissions
```
sudo mkdir -p /var/security/packages/{reports,baselines,logs}
sudo chmod 750 /var/security/packages && sudo chown root:root /var/security/packages
```

_____

Make a backup of your current configurations
```
sudo cp -r /etc/apt /etc/apt.backup.$(date +%Y%m%d_%H%M%S)
```

_____

Create two scripts: one for integrity checking using debsums to check the installed version, digital signature verification with apt-keylist to ensure all repositories from which packages are downloaded are trusted and have valid signing keys, and security upgrades for already installed packages. A second script will act as a continuous surveillance system to actively monitor emerging threats with: Ubuntu system monitoring, CVE checking of critical packages, security auditing with the integrated Lynis, rootkit and malware checking with rkhunter, clamav, and chrootkit, and will return a quantifiable risk score.
```
sudo nano /usr/local/bin/check-package-integrity.sh
sudo chmod +x /usr/local/bin/check-package-integrity.sh

sudo nano /usr/local/bin/vulnerability-monitor.sh
sudo chmod +x /usr/local/bin/vulnerability-monitor.sh
```

_____

Configure automatic security updates and enable periodic updates by editing the configuration files with the desired rules
```
sudo nano /etc/apt/apt.conf.d/50unattended-upgrades
sudo nano /etc/apt/apt.conf.d/20auto-upgrades
```

_____

Automating with crontab to run scripts

Risk-012

Risk Level: Medium

Treatment: Acceptance and transfer

An attacker who managed to compromise the host system hosting the server could gain access through boot, bypassing the password login. Given the low probability of this happening, I recommend accepting the risk while waiting to transfer it by migrating to a professional cloud provider. This involves transferring not only the specific risk but the entire physical security layer to industry specialists, ensuring that the chosen provider complies with current regulations and holds up-to-date certifications (ISO27001).

Risk-013

Risk Level: Medium
Treatment: Mitigate

Core dumps are memory files automatically generated by the system when an application terminates abnormally. These files may contain cleartext passwords and credentials, sensitive data processed by the application, information about the internal structure of the system, and temporary encryption keys. In the context of a LAMP server, uncontrolled core dumps pose a serious risk to the protection of personal data. My suggestion is to completely disable core dump creation and storage, set up an external server for debug log collection, and implement safer alternatives for debugging.

Create a backup of your current configurations

```
sudo cp /etc/security/limits.conf /etc/security/limits.conf.backup.$(date +%Y%m%d_%H%M%S)
sudo cp /etc/systemd/coredump.conf /etc/systemd/coredump.conf.backup.$(date +%Y%m%d_%H%M%S)
```

_____

Edit the limits.conf file to disable core dumps for all users. This file controls resource limits for users. Setting the core dump limit to 0 for both the "soft" and "hard" limits prevents core dumps from being generated for all users on the system.

```
echo "* hard core 0" | sudo tee -a /etc/security/limits.conf
echo "* soft core 0" | sudo tee -a /etc/security/limits.conf
```

_____

Change the systemd configuration for core dumps. The Storage=none configuration prevents systemd from saving core dumps, while other parameters set to 0 ensure that no dumps are created or retained.

```
sudo nano /etc/systemd/coredump.conf
```

Apply changes by reloading systemd

```
sudo systemctl daemon-reload
```

_____

Create a kernel security sysctl configuration file specific to core dumps that prevents creation of core dumps for elevated processes, redirects any core dump attempts to /bin/false which does nothing, and disables appending the PID to the core dump name.

```
sudo nano /etc/sysctl.d/99-disable-coredump.conf
```

Apply changes immediately

```
sudo sysctl -p /etc/sysctl.d/99-disable-coredump.conf
```

_____

System configuration to ensure that each new user session automatically has core dumps disabled, providing an additional layer of protection

```
echo "ulimit -c 0" | sudo tee -a /etc/profile
echo "ulimit -c 0" | sudo tee -a /etc/bash.bashrc
echo "ulimit -c 0" | sudo tee -a /root/.bashrc
```

_____

Implement an external logging system using rsyslog-gnutls to replace the debugging functionality lost when disabling core dumps.

Note: External logging is essential because disabling core dumps loses valuable debugging information. A centralized logging system allows you to track errors without exposing sensitive data.

_____

Install additional debugging tools that don't produce core dumps but rather well-structured and secure logs, such as gdb and strace. These tools operate in real time and can be configured to automatically filter out potentially sensitive information.

_____

Consider creating a script to monitor deployed solutions and schedule its automation with crontab.

Risk-014

Risk Level: Medium
Treatment: Mitigate

Every service that listens on a network port creates an attack surface. Attackers use automated scanners that test thousands of servers every day, looking for open ports on vulnerable services.
Mitigations for risks 001, 005, and 010 partially address this vulnerability. I recommend implementing an automated IPv4 and IPv6 port control system that detects the opening of non-essential ports, complemented by an alerting system.

Create a script that uses netstat and ss to check ports and compares it with the attached ports.txt file to verify that only those ports needed for the services to function are open. It also includes an email alert system for new listening ports (in case the system is compromised and ports are opened to advance the attack), a change log if any (complementary to the system implemented in Risk-007), and automation with crontab.

Note: Consider implementing integrity checks for the script itself and, ideally, sending alerts through channels independent of the monitored server.

Risk-015

Risk Level: Medium

Treatment: Mitigate

Active process management is a critical element of LAMP server security. Every running process represents a potential attack surface, especially those maintaining network connections or accessing sensitive resources. A compromised process can serve as an entry point for lateral movement within the system.

Backup current configurations for emergency rollback

```
sudo cp /etc/systemd/system.conf /etc/systemd/system.conf.backup.$(date +%Y%m%d_%H%M%S)
```

_____

Create a dedicated folder structure to store process analytics.

```
sudo mkdir -p /var/security/processes/{baseline,current,reports}
sudo chmod 750 /var/security/processes
sudo chown root:root /var/security/processes
```

_____

Generate a complete baseline of running processes. The ps aux command displays all processes with detailed information (user, PID, CPU/memory usage, command). The --sort=-%cpu option sorts by decreasing CPU usage, while --sort=-%mem sorts by decreasing memory usage.

```
ps aux --sort=-%cpu > /var/security/processes/baseline/processes_cpu_sorted.txt
ps aux --sort=-%mem > /var/security/processes/baseline/processes_mem_sorted.txt
```

_____

Analyzing active systemd services: systemctl list-units lists systemd units. The --type=service filter shows only services, while --state=active and --state=failed filter for active and failed services, respectively.

```
systemctl list-units --type=service --state=active > /var/security/processes/baseline/active_services.txt
systemctl list-units --type=service --state=failed > /var/security/processes/baseline/failed_services.txt
```

_____

Mapping Process Network Connections

```
netstat -tulpn > /var/security/processes/baseline/network_connections.txt
ss -tulpn > /var/security/processes/baseline/socket_statistics.txt
```

_____

Create a script that uses the previously created files to scan for active, network-using, and non-essential processes.

```
sudo nano /usr/local/bin/analyze-active-processes.sh
sudo chmod +x /usr/local/bin/analyze-active-processes.sh
```

_____

Process analysis at startup. This command lists all systemd services configured to start automatically. It's important to ensure only the necessary services are enabled.

```
systemctl list-unit-files --type=service --state=enabled > /var/security/processes/current/enabled_services.txt
```

_____

Disable unnecessary services for a LAMP server. These commands disable services typically not needed on a production LAMP server. Bluetooth and CUPS are for desktop, avahi is for network discovery, whoopsie and kerneloops are for error reporting.

```
sudo systemctl disable bluetooth.service
sudo systemctl disable cups.service
sudo systemctl disable avahi-daemon.service
sudo systemctl disable whoopsie.service
sudo systemctl disable kerneloops.service
```

_____

Create systemd configuration to limit resources with global limits to prevent fork bombs and excessive resource consumption, and check and timeout for unresponsive services.

```
sudo nano /etc/systemd/system.conf.d/99-resource-limits.conf
```

Apply changes to systemd

```
sudo systemctl daemon-reload
```

_____

Create scripts for continuous process monitoring with: a function to detect new processes, email alerts, monitoring resource-intensive processes, and automating with crontab. I recommend using the comm command to compare sorted lists and awk to filter by usage thresholds.

```
sudo nano /usr/local/bin/monitor-processes.sh
```

Risk-016

Risk Level: Medium

Treatment: Mitigate

The presence of test or history pages represents an information disclosure vulnerability, where HTML or PHP files can reveal confidential information that can be used for subsequent attacks. Furthermore, when Apache displays empty directories or detailed error pages, it provides attackers with valuable information about the system structure.

Check for test HTML or PHP files and remove them from web-exposed directories. The find command recursively searches the /var/www directory (including all subdirectories) for files with common web extensions. The -type f option searches only files (not directories), while \(-name "*.html" -o -name "*.htm"... \) specifies search patterns with logical OR. The -exec ls -la {} \; command runs ls -la on each found file to display permissions, owner, and timestamp.

```
sudo find /var/www -type f \( -name "*.html" -o -name "*.htm" -o -name "*.php" -o -name "*.js" -o -name "*.css" \) -exec ls -la {} \; 2>/dev/null
```

_____

Modify Apache's configuration to hide system information. This includes disabling directory listing, which prevents users from seeing the contents of empty directories. I also recommend creating a generic index page that overrides any default Apache messages for added security.

```
sudo nano /var/www/html/index.html
sudo nano /etc/apache2/conf-available/web-files-protection.conf
sudo a2enconf web-files-protection
sudo apache2ctl configtest
sudo systemctl reload apache2
```

Risk-017

Risk Level: Medium
Treatment: Mitigate

When an attacker compromises a system, their first priority is often to erase all traces of their actions. Without an external logging system, all logs reside on the same compromised server and can be easily deleted.

I recommend installing rsyslog-gnutls (if not already implemented) and rsyslog-relp because they represent the standard for logging in a Linux environment. The rsyslog-gnutls component adds TLS encryption, which transforms potentially interceptable communication to a dedicated remote logging server into a secure channel. It is configured to use Reliable Event Logging Protocol (RELP) to ensure every log message reaches its destination.

Ensure that automatic mechanisms for log classification and differentiated retention are active and compliant with current regulations.

It's also a good idea to integrate a local buffering system in case the connection is temporarily interrupted. This ensures that no data is lost even during service interruptions. Make sure this system guarantees accurate timestamps.

Create an automatic monitoring script to verify that the various logs have been collected and sent successfully.

Risk-018

Risk Level: Medium
Treatment: Mitigate

Incorrect user and group management is one of the major vulnerabilities on a system, often exploited to elevate privileges. Therefore, it is necessary to pay close attention to the permissions of individual users, user groups, and service accounts.

Given the attachment etcpassw.txt some unused services have been detected that can be removed, after verification, the following command performs a check

```
systemctl status games 2>/dev/null || echo "games service not found"
systemctl status news 2>/dev/null || echo "news service not found"
systemctl status uucp 2>/dev/null || echo "uucp service not found"
systemctl status irc 2>/dev/null || echo "irc service not found"
```

If these services are not found, their accounts may be removed.

```
sudo userdel games
sudo userdel news
sudo userdel uucp
sudo userdel irc
```

_____

Other potentially obsolete accounts are: Landscape service, USB multiplexing service

```
sudo userdel landscape
sudo userdel usbmux
```

____

Disable unused service accounts, such as the one for printers

```
sudo usermod -s /usr/sbin/nologin -L lp
```

Risk-019

Risk Level: Medium

Treatment: Mitigate

There are some compilers on the server (compiler.txt attachment) that could potentially be exploited by an attacker who managed to gain access with a user with limited permissions to elevate privileges.

I recommend limiting compiler permissions to the Administrator user group and adding a policy to prevent accidental compiler installation, and consider installing and configuring AppArmor for additional protection.

Risk-020

Risk Level: Medium

Treatment: Mitigate

Every enabled but unused network protocol represents a potential attack vector. Disable non-essential protocols.

The analysis (attached to protocols.txt) reveals several non-essential network protocols; the safest option is to disable them.

Create a backup of your current configurations (with timestamp)

```
sudo cp -r /etc/modprobe.d /etc/modprobe.d.backup.$(date +%Y%m%d_%H%M%S)
```

_____
Create the configuration file to disable the protocols and add to the blacklist: dccp, sctp, rds, tipc, atm, can, x25, netrom, rose, ax25, ieee802154
sudo nano /etc/modprobe.d/blacklist-unused-protocols.conf

_____
Consider implementing a firewall configuration to block protocols at the network layer for additional protection.

Risk-021

Risk Level: Bass
Treatment: Mitigate
The umask is a value that determines the permissions assigned to a new file. Excessively broad permissions could be used in some very specific exploits to allow privilege elevation.
Mitigation involves setting more restrictive permissions for newly created files

Change the umask configuration from the current 002 to 027 keeping rwx for the owner but removing write permission for the group and all permissions from other users.
sudo nano /etc/login.defs
sudo nano /etc/profile
sudo nano /etc/bash.bashrc

_____
Set an even more restrictive umask for the root user, such as 077
sudo nano /root/.bashrc

_____
Configure a less restrictive umask (UMask=0022) for Apache which requires permissions to properly serve web pages
sudo mkdir -p /etc/systemd/system/apache2.service.d/
sudo nano /etc/systemd/system/apache2.service.d/umask.conf

And a more restrictive one for MariaDB (UMask=0077)
sudo mkdir -p /etc/systemd/system/mariadb.service.d/
sudo nano /etc/systemd/system/mariadb.service.d/umask.conf

Reload systemd to apply the changes
sudo systemctl daemon-reload

Risk-022

Risk Level: Bass
Treatment: Mitigate

The usb-storage driver allows the system to automatically recognize and mount USB storage devices. Disabling it eliminates the possibility of attacks using malicious USB drives.

Create a file that prevents the kernel from loading the USB driver (usb-storage blacklist)
`sudo nano /etc/modprobe.d/blacklist-usb-storage.conf`

_____
Remove the active form
`sudo modprobe -r usb-storage`

_____
Update the intramfs to make the blacklist active at system startup
`sudo update-initramfs -u`

_____
Verify that the driver is no longer loaded
`lsmod | grep usb_storage`

Risk-023

Risk Level: Bass
Treatment: Accept
The risk of physical access to the system can never be completely eliminated. It's acceptable for the current situation, but consider transferring the risk to a third party by migrating the server to a dedicated cloud service.

Temporary mitigations that can be implemented are
- Set a GRUB boot password
- Encryption of sensitive partitions
- Lock the door to the room containing the server
- Implement a monitoring system with alerts in case of anomalies

---

Risk-024

Risk Level: Bass
Treatment: Mitigate
The presence of obsolete files on the system constitutes a vulnerability because they may contain outdated packages with known vulnerabilities, files with sensitive data, cause space management issues, and, in the long run, make it difficult to distinguish legitimate files from obsolete ones. This risk has been mitigated as part of the previous risk mitigations in the same document.

I recommend implementing a script to automate with crontab to delete obsolete files with the identification of: temporary files older than 30 days, system cache files, backup files older than a year

_____

Identifying obsolete packages with
apt autoremove

_____
Check the configuration of deborphan (already suggested RISK-009) and logrotate (RISK-002).

Residual Risk

The residual risk analysis for the Ubuntu 22.04.2 LAMP server shows a significant improvement in the security profile following the implementation of the twenty-four proposed mitigations. The overall residual risk score is low-medium, indicating a substantial reduction compared to the initial condition, which was characterized by critical risks with scores of up to 25. The main remaining risks relate to physical access to the system and unidentified zero-day vulnerabilities, factors that are inherently difficult to completely eliminate through technical controls. The adopted mitigations, which range from firewall configuration to SSH hardening, from the encrypted backup system to the Web Application Firewall, have converted critical risks into manageable vulnerabilities. Residual risk requires an ongoing management approach based on proactive monitoring, timely updates, and well-defined incident response procedures. The strategy of transferring risk to professional cloud providers, recommended for physical security risks, represents the natural progression for achieving an enterprise-level level of security. Formal acceptance of this level of residual risk by management is essential, given that it represents an optimal balance between data protection, GDPR compliance, and system operation. Quarterly review of security metrics and continuous updating of control baselines will ensure this risk profile is maintained over time. In short, the server is now adequately protected for a production environment, with a manageable and documented residual risk.

Further considerations and best practices

To strengthen the security of a LAMP server, in addition to technical mitigations, it is crucial to adopt several best practices. Staff training and awareness are essential to mitigate the risks of human error and negligence. The principle of least privilege must be applied, granting only strictly necessary permissions to users and processes to reduce the attack surface. Network

segmentation, via VLANs or firewalls, limits the lateral movement of attackers in the event of a compromise.

Continuous monitoring with SIEM systems and a well-defined incident response plan are essential to proactively detect threats and effectively manage breaches. Regular penetration tests and vulnerability assessments allow you to identify and correct weaknesses before they are exploited. It is imperative to encrypt sensitive data at rest and rigorously manage configurations, using automation tools to ensure consistency and compliance.

Integrating security into the software development lifecycle (DevSecOps) is crucial for internal applications. Finally, a robust patch and update management process is essential to protect servers from known vulnerabilities. The combined application of these practices creates a multi-layered defense, improving resilience and GDPR compliance.

Date: 07/28/2025

_____ Andrea Emanuele Peluso