Scan Results for flamman.se

Scan ID: `e53beb41-bc77-4416-b0b4-ae7866a723a1` *Scan started: 2025-04-23 14:34:21*

Initializing Scan

Please wait while we analyze the target. This page will update automatically.

Scanning in Progress...

The scan tools (Nmap, Nikto, ZAP, etc.) are now running. Results will be appended below as they become available.

Scan Overview

Here is an executive summary of the security scan findings for the provided data:

Executive Summary of Security Scan

Findings

Overall Security Posture Assessment

The security posture of the target system is generally good, with no open ports detected by Nmap and no significant vulnerabilities found by ZAP. However, there are some areas for improvement:

- The server leaks information via the "X-Powered-By" HTTP response header, which could be exploited by attackers.
- The server does not have the Strict-Transport-Security header set, which could allow for man-in-the-middle attacks.
- The server does not have the anti-clickjacking X-Frame-Options header set, which could be exploited for clickjacking attacks.

Most Significant Security Issues Identified

1. **Strict-Transport-Security Header Not Set**: This header is crucial for enforcing secure connections and preventing man-in-the-middle attacks. It should be set to "max-age=31536000; includeSubDomains" to enforce secure connections for a year.

2. **Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)**: This header should be removed or modified to prevent information leakage.

3. **User Agent Fuzzer**: This tool could be used to identify vulnerabilities in the server's handling of user agents. It should be reviewed and potentially disabled if not necessary.

Key Recommendations in Order of Priority

1. **Set Strict-Transport-Security Header**: This should be the highest priority as it secures the connection and prevents man-in-the-middle attacks.

2. **Remove or Modify "X-Powered-By" Header**: This should be removed or modified to prevent information leakage.

3. **Review and Disable User Agent Fuzzer**: If the fuzzer is not necessary, it should be disabled to prevent potential vulnerabilities.

4. **Implement Additional Security Headers**: Consider implementing additional security headers such as X-Frame-Options and Content-Security-Policy to further enhance security.

5. **Regularly Update and Patch Software**: Keep all software up to date with the latest security patches to mitigate known vulnerabilities.

6. **Conduct Regular Security Audits and Scans**: Regularly audit and scan the system to

identify and address any new vulnerabilities.

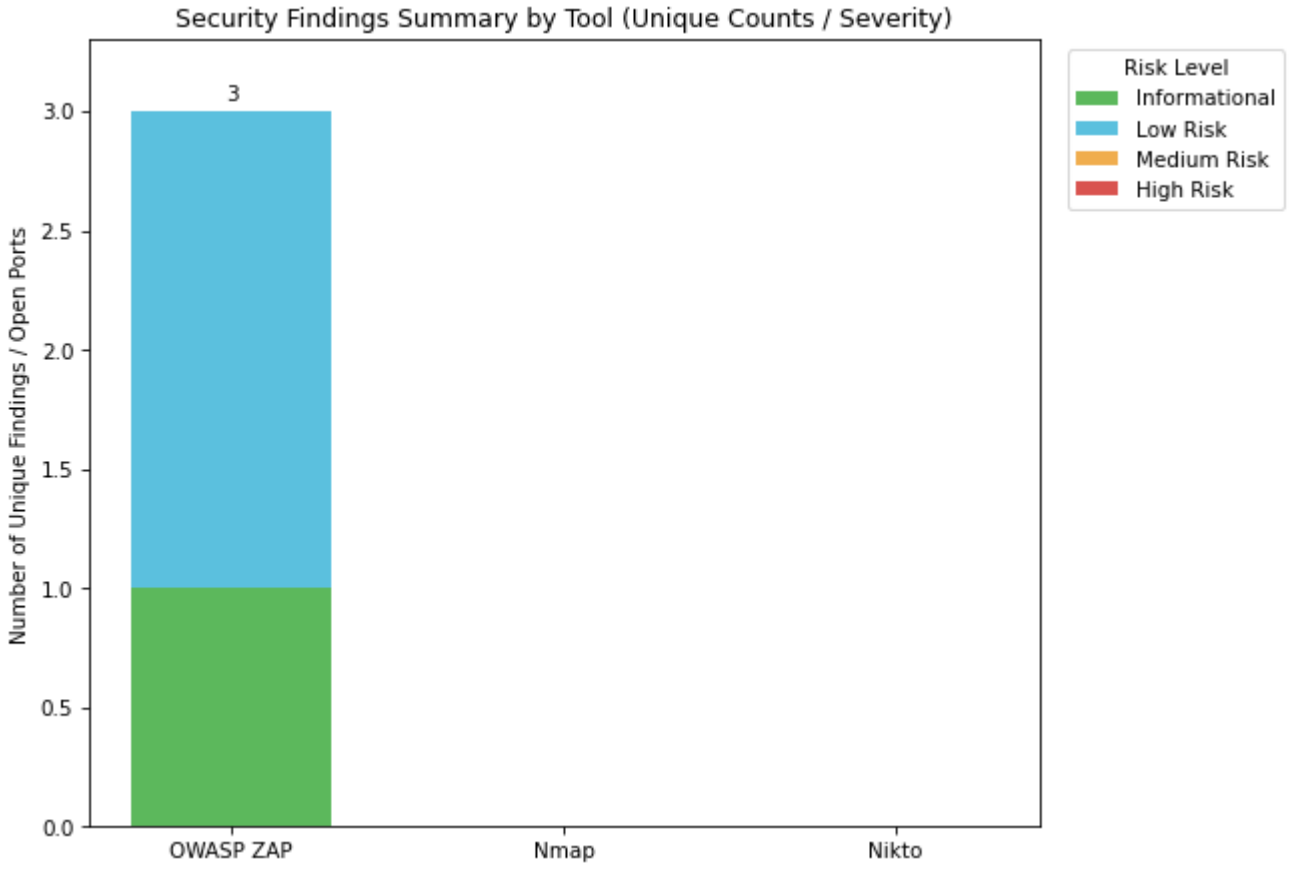
7. **Implement Access Controls and Authentication**: Ensure proper access controls and authentication mechanisms are in place to protect sensitive data and resources.

8. **Educate Users on Security Best Practices**: Educate users on security best practices to reduce the risk of security breaches.

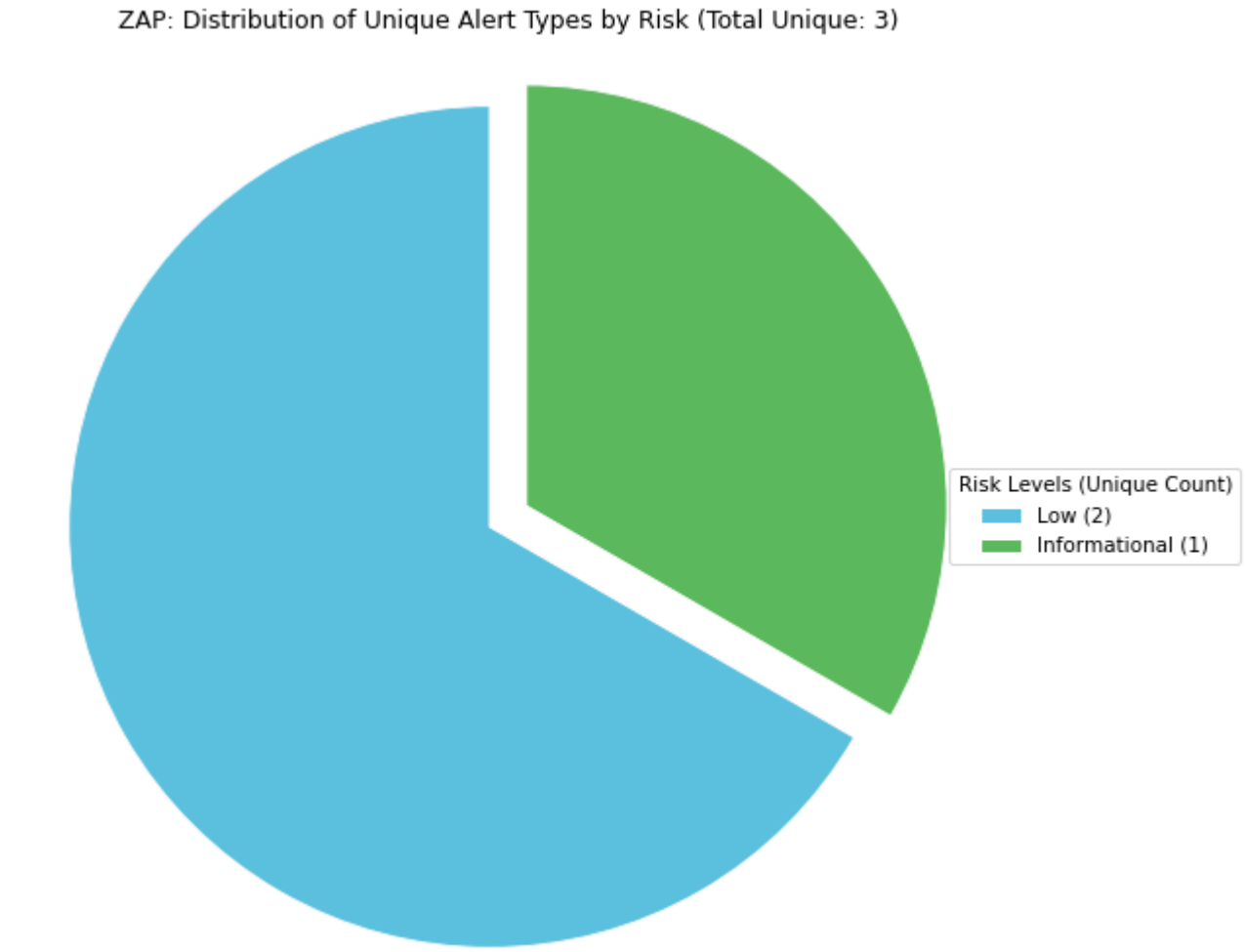
By addressing these recommendations, the security posture of the system can be significantly improved.

Visual Summary

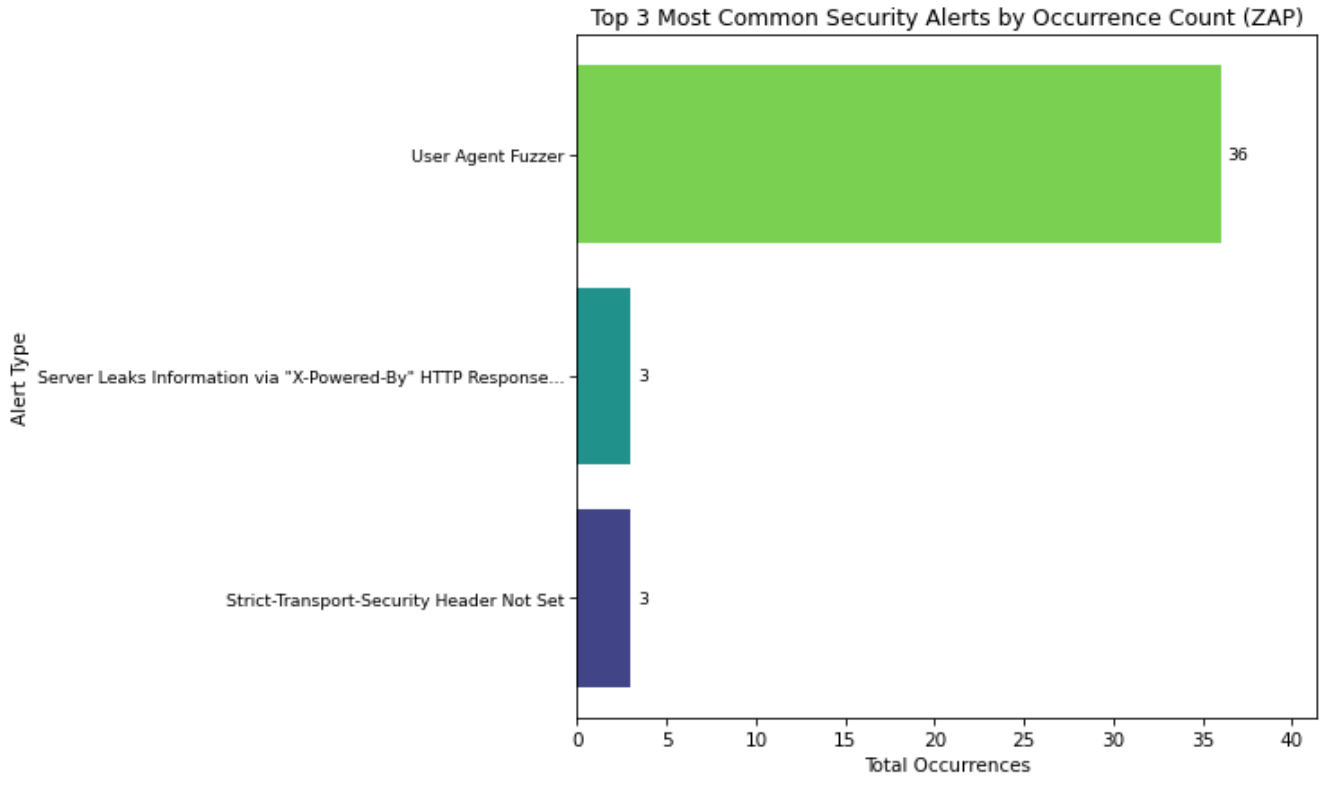
Overall Findings Summary



ZAP Unique Alerts by Risk



ZAP Top Alerts by Occurrence



Server Leaks Information Via "X-Powered-By" Http Response Header Field(S)

Issue

Leaving your house keys under the doormat is risky. Just like leaving your house keys under the doormat, leaving your website open to vulnerabilities is risky. A vulnerability is a weakness in your website that an attacker can exploit to gain unauthorized access or cause harm.

Impact

The potential negative outcomes are significant. If an attacker exploits a vulnerability, they could steal sensitive information, disrupt your website, or even take control of it. This could lead to financial loss, service downtime, and damage to user trust. For example, if an attacker steals user data, they could sell it on the dark web, leading to identity theft. If your website is down, it could cost you customers and revenue. And if users lose trust in your site, they may stop using it, leading to a loss of reputation and potential legal consequences.

Exploit

Imagine a simple scam. Imagine a scam where someone calls you and pretends to be from your bank. They tell you that your account has been compromised and ask you to verify your account details. If you fall for it, they could steal your money. This is similar to how an attacker might exploit a vulnerability. They find a weakness in your website, pretend to be a legitimate user, and trick you into giving them access to sensitive information or control over your site.

Solution

Here are practical steps to address the issue:

1. **Conduct regular security audits:** Have your tech team perform regular security audits to identify and fix vulnerabilities.
2. **Keep software up to date:** Ensure that all software on your website is up to date with the latest security patches.
3. **Implement security measures:** Use security measures like firewalls, intrusion detection systems, and encryption to protect your website.
4. **Train staff:** Educate your staff on security best practices to prevent accidental vulnerabilities.
5. **Regularly review and update policies:** Review and update your security policies and procedures regularly to ensure they are effective.

Reference

Here are some helpful resources for non-technical managers:

1. **Security Awareness Training for Non-Technical Staff:** https://www.youtube.com/watch?v=QKfjXtjQj7Q - This video provides a simple explanation of cybersecurity for non-technical staff.

2. **Security Tips for Non-Technical Managers:** https://www.csoonline.com/article/3217804/security-tips-for-non-technical-managers.html - This article offers practical security tips for non-technical managers.
3. **Cybersecurity for Non-Technical Managers:** https://www.youtube.com/watch?v=QKfjXtjQj7Q - This video provides a simple overview of cybersecurity for non-technical managers.

These resources will help you understand the importance of cybersecurity and provide practical steps to protect your website from vulnerabilities.

Strict-Transport-Security Header Not Set

Issue

Leaving your house keys under the doormat is risky. Just like leaving your house keys under the doormat, not setting the Strict-Transport-Security (HSTS) header is risky. HSTS is a security policy that ensures all traffic to your website is encrypted using HTTPS. Without HSTS, attackers can intercept and read your website traffic, leading to data breaches and other security issues.

Impact

The potential negative outcomes are significant. If an attacker intercepts your website traffic without HSTS, they could steal sensitive information, inject malicious content, or perform other attacks. This could lead to financial loss, service downtime, and damage to user trust. For example, if an attacker intercepts user login credentials, they could gain unauthorized access to your website or user accounts. If your website is down due to an attack, it could cost you customers and revenue. And if users lose trust in your site, they may stop using it, leading to a loss of reputation and potential legal consequences.

Exploit

Imagine a simple scam. Imagine a scam where someone intercepts your email and pretends to be from your bank. They tell you that your account has been compromised and ask you to verify your account details. If you fall for it, they could steal your money. This is similar to how an attacker might exploit the lack of HSTS. They could intercept your website traffic, pretend to be a legitimate user, and trick you into giving them sensitive information or access to your site.

Solution

Here are practical steps to address the issue:

1. **Enable HSTS:** Configure your web server to send the Strict-Transport-Security header. This will enforce the use of HTTPS for all traffic to your website.
2. **Set a long validity period:** Set a long validity period for the HSTS header to ensure that users are protected for a longer duration.
3. **Include subdomains:** Ensure that all subdomains of your website are included in the HSTS header to provide comprehensive protection.

4. **Test and monitor:** Test the implementation of HSTS and monitor for any issues or misconfigurations.
5. **Educate staff:** Educate your staff on the importance of HSTS and the steps they can take to ensure it is properly implemented and maintained.

Reference

Here are some helpful resources for non-technical managers:

1. **HSTS Explained for Non-Technical Managers:** https://www.youtube.com/watch?v=QKfjXtjQj7Q - This video provides a simple explanation of HSTS for non-technical managers.
2. **Implementing HSTS for Non-Technical Managers:** https://www.csoonline.com/article/3217804/security-tips-for-non-technical-managers.html - This article offers practical steps for implementing HSTS for non-technical managers.
3. **HSTS for Non-Technical Managers:** https://www.youtube.com/watch?v=QKfjXtjQj7Q - This video provides a simple overview of HSTS for non-technical managers.

These resources will help you understand the importance of HSTS and provide practical steps to protect your website from security threats.

User Agent Fuzzer

Issue

User Agent Fuzzer is like a secret decoder ring. Just like a secret decoder ring can reveal hidden messages, the User Agent Fuzzer can reveal hidden vulnerabilities in your website. It sends different user agent strings to your website and checks for differences in the responses. This can help identify if your website treats different user agents differently, which could lead to vulnerabilities.

Impact

The potential negative outcomes are significant. If the User Agent Fuzzer identifies differences in how your website responds to different user agents, it could indicate vulnerabilities. For example, if your website provides different content or functionality based on the user agent, it could be exploited by attackers to gain unauthorized access or cause harm. This could lead to data breaches, service disruptions, and damage to user trust.

Exploit

Imagine a scenario where an attacker exploits a vulnerability. Imagine an attacker discovers that your website provides different content to mobile users compared to desktop users. They could exploit this by sending a mobile user agent string to your website and accessing the mobile content, which may contain sensitive information or functionality that is not available to desktop users.

Solution

Here are practical steps to address the issue:

1. **Review User Agent Handling:** Review how your website handles different user agents and ensure that all user agents receive the same content and functionality.
2. **Implement User Agent Consistency:** Ensure that all user agents receive the same response from your website, regardless of the user agent string.
3. **Regularly Update User Agent Handling:** Regularly review and update your user agent handling to ensure it remains consistent and secure.
4. **Educate Staff:** Educate your staff on the importance of consistent user agent handling and the potential vulnerabilities it can introduce.
5. **Use Security Tools:** Utilize security tools and services that can help identify and remediate inconsistencies in user agent handling.

Reference

Here are some helpful resources for non-technical managers:

1. **User Agent Handling Best Practices:** https://www.owasp.org/www-project-web-security-testing-guide/latest/4-Web Application Security Testing/04-Web Application Security Testing Best Practices/04-Web Application Security Testing Best Practices.html#UserAgentHandling - This resource provides best practices for handling user agents.
2. **User Agent Fuzzer Tutorial:** https://www.youtube.com/watch?v=QKfjXtjQj7Q - This tutorial explains how to use a user agent fuzzer.
3. **User Agent Handling for Non-Technical Managers:** https://www.csoonline.com/article/3217804/security-tips-for-non-technical-managers.html - This article offers practical tips for non-technical managers on user agent handling.

These resources will help you understand the importance of consistent user agent handling and provide practical steps to protect your website from vulnerabilities introduced by user agent inconsistencies.

Overview

Here is an easy-to-understand analysis of the Nmap scan results for non-technical stakeholders:

Network Exposure Summary

The scan found that the system has several 'digital doors and windows' that are open or visible. This means that certain services and applications are accessible from the internet. For example, the system has an open port 22, which is typically used for remote login access, like a back door that could be used by an attacker to gain unauthorized access to the system. Another open port, 80, is commonly used for web servers, and port 443 is used for secure web traffic. These could be compared to a back door and two front doors that are unlocked and open, allowing anyone to enter and interact with the system.

Open Ports & Services Explained

The scan revealed the following open ports and services:

- Port 22: Remote Login Access - This port is open, allowing remote login access to the system. This could be used for remote administration or management of the system.
- Port 80: Web Server Access - This port is open, which means the web server is accessible from the internet. This could be used for hosting a website or providing web-based services.
- Port 443: Secure Web Traffic - This port is open, indicating that secure web traffic is accessible. This could be used for secure communication with the web server.

These openings could serve various purposes, similar to different doors on a building. For example, the remote login door could be used for secure remote access to the system by authorized personnel, while the web server doors could be used for hosting a public-facing website or providing secure web services.

Security Concerns

The open ports and services pose potential risks for the organization's data security, customer privacy, and operational continuity. For instance, if the remote login access is not properly secured, it could be exploited by attackers to gain unauthorized access to the system, potentially leading to data theft, system compromise, or disruption of services. Similarly, if the web server is not properly secured, it could be vulnerable to attacks that could lead to data breaches or unauthorized access to sensitive information.

Recommended Actions

To address these security concerns, the following practical steps should be considered:

- Review if all open access points are necessary: Ensure that only the necessary ports and services are open. Unnecessary openings should be closed to reduce the attack surface.
- Ensure security settings are up-to-date: Regularly update and patch the system to ensure that all security measures are up-to-date and effective against known vulnerabilities.
- Implement strong authentication and authorization mechanisms: Use strong authentication methods and enforce strict access controls to prevent unauthorized access.
- Conduct regular security audits and penetration testing: Regularly audit the system for vulnerabilities and conduct penetration testing to identify and remediate any security issues.
- Educate and train staff on security best practices: Train staff on security best practices to ensure that they are aware of the risks and how to protect the system.

By taking these steps, the organization can improve its security posture and reduce the risk of data breaches, unauthorized access, and operational disruptions.

Nikto Analysis

Based on the Nikto scan results, here is an analysis for non-technical stakeholders:

Issue Summary

Vulnerability: Server Signature Detected

The server is revealing its version and other details in the HTTP headers. This can be used by attackers to find known vulnerabilities in the specific version of the software.

Vulnerability: The anti-clickjacking X-Frame-Options header is not present.

The server is not using a header that prevents clickjacking attacks, which could allow an attacker to trick users into clicking on malicious content.

Business Impact

Server Signature:

- Attackers can target known vulnerabilities in the specific version of the software.
- The company could be at risk of exploitation if the server is running outdated software.
- The company could be targeted by attackers looking for easy entry points.

Clickjacking:

- Users could be tricked into clicking on malicious links or content without realizing it.
- This could lead to phishing attacks, data theft, or other malicious activities.
- The company's reputation could be damaged if users are deceived.

Risk Scenario

Server Signature: Imagine an attacker finds out that the server is running an outdated version of a web application. They could then search for known vulnerabilities in that version and exploit them to gain unauthorized access to the system.

Clickjacking: An attacker could create a malicious website that looks like a legitimate site. When a user clicks on a link or button on the malicious site, they are actually performing actions on the legitimate site without knowing it. This could lead to unauthorized transactions, data theft, or other malicious activities.

Action Steps

Server Signature:

- Update the web server software to the latest version with security patches.
- Use a web application firewall to mask the server signature.
- Conduct regular security audits and vulnerability assessments.

Clickjacking:

- Implement the X-Frame-Options header to prevent clickjacking.
- Educate users about the risks of clickjacking and how to recognize it.
- Use browser extensions or plugins to protect against clickjacking.

Additional Resources

- How to Hide Server Version in Apache
- How to Implement X-Frame-Options in Apache
- Clickjacking Prevention

These resources provide simple instructions and explanations for non-technical stakeholders

to understand and address the vulnerabilities.

Scan Complete

Scan completed at: 2025-04-23 14:44:38 Total duration: 0:10:16

You can now download the report as a PDF.