MyESI Automated Compliance & Vulnerability Assessment

Project: myesi-user-service

Generated: 2025-11-09 13:24 UTC **User:** 1

1. Executive Summary

- Compliance Score: **68.0%**
- Average Risk Score: **9.50**
- Total Vulnerabilities: **8**

Standards Compliance:

Standard	Status
ISO_27001	Partially
NIST_SP_800_53	Needs Review
OWASP	Needs Review

2. Compliance Controls Overview

A.8.7 — Protection against malware *(Technological)* — Score: **90.0%**

1. Plan fix in next release cycle. Limit exposure through configuration hardening.
2. Validate compliance and re-run assessment.

A.8.9 — Configuration management *(Technological)* — Score: **30.0%**

1. Implement Configuration Management Policies - Establish clear policies for configuration management that include documentation, approval processes, and regular review.
2. Utilize Configuration Management Tools - Example command: Use tools such as Puppet, Chef, or Ansible to automate configuration and ensure consistent application across systems. - Example: ansible-playbook site.yml to apply configurations defined in an Ansible playbook.
3. Regularly Review and Audit Configurations - Conduct periodic audits and reviews of system configurations against defined standards to ensure compliance and identify deviations. - Guidance: Schedule audits every quarter and use automated scripts to compare current configurations with a baseline or desired state.

3. Vulnerability Findings

#	Component	Version	Vuln ID	Severity	Fix / Recommendation
1	**bcrypt**	4.3.0	GHSA-5wg4-74h6-q47v	**Moderate**	**Update to:** 5.0.0
2	**loguru**	0.7.3	MAL-2025-25559	**Critical**	### Summary This control requires attention in General. ### Remediation Steps - Immediately patch or isolate affected components. Deploy emergency fixes and block exploit paths. - Validate compliance and re-run assessment.
3	**mypy-extensions**	1.1.0	MAL-2024-2685	**Critical**	### Summary This control requires attention in General. ### Remediation Steps - Immediately patch or isolate affected components. Deploy emergency fixes and block exploit paths. - Validate compliance and re-run assessment.
4	**pydantic**	2.12.2	MAL-2025-4867	**Critical**	```markdown ### Summary MAL-2025-4867 pertains to a critical vulnerability in the pydantic library that may allow attackers to exploit improper data validation, leading to unauthorized access or data manipulation. Immediate remediation is essential to protect applications that utilize this library. ### Remediation Steps 1. **Upgrade pydantic to a secure version** - Ensure that you are using the latest version of pydantic where the vulnerability has been patched. 2. **Example Command** ```bash pip install --upgrade pydantic ``` 3. **Configuration Guidance** - Review and adjust your application code to adhere to stricter validation rules and ensure that all data inputs are sanitized appropriately. Check the pydantic documentation for best practices on model validation and handling input data. ```
5	**pyjwt**	2.10.1	MAL-2025-48036	**Critical**	### Summary This control requires attention in General. ### Remediation Steps - Immediately patch or isolate affected components. Deploy emergency fixes and block exploit paths. - Validate compliance and re-run assessment.

6	**python-dotenv**	1.1.1	MAL-2025-48037	**Critical**	### Summary This control requires attention in General. ### Remediation Steps - Immediately patch or isolate affected components. Deploy emergency fixes and block exploit paths. - Validate compliance and re-run assessment.
7	**typing-extensions**	4.15.0	MAL-2025-47895	**Critical**	### Summary This control requires attention in General. ### Remediation Steps - Immediately patch or isolate affected components. Deploy emergency fixes and block exploit paths. - Validate compliance and re-run assessment.
8	**uvicorn**	0.37.0	MAL-2025-4901	**Critical**	### Summary The vulnerability identified as MAL-2025-4901 in the Uvicorn web server can lead to critical security risks, including unauthorized access to sensitive data and potential takeover of server sessions. ### Remediation Steps 1. **Upgrade Uvicorn**: Immediately upgrade to the latest version of Uvicorn where the vulnerability has been patched. - Example Command: `pip install --upgrade uvicorn` 2. **Restrict Access**: Implement IP whitelisting to restrict access to the Uvicorn server from untrusted sources. - Example Command: Use a firewall rule to allow traffic only from specific IPs. 3. **Secure Configuration**: Ensure that the server is configured to run with specific flags to enhance security. - Config Guidance: Launch Uvicorn with `--host 0.0.0.0 --port 8000 --log-level info` and consider implementing HTTPS with a reverse proxy (e.g., Nginx, Apache) to enforce encrypted connections. ■ Support this free API: https://www.paypal.com/donate/?hosted_button_id=XS3CAYT8LE2BL

Generated by MyESI on 2025-11-09 13:27 UTC — User: 1