

fail2ban-ui

github.com/swissmakers/fail2ban-ui

Scanned 24 September 2026 at 09:00 · Report generated 29 October 2026 · Go · 352 stars

OVERALL ASSESSMENT

Medium

Combined score 61 · 34 findings after filtering

Summary

The repository has a medium vulnerability score of 30.0 and a medium malicious-pattern score of 30.0, indicating that while the code contains some risky patterns, it does not appear to be intentionally designed to harm users. The most significant issues involve base64-encoded scripts that are decoded and executed by JavaScript files like echarts.min.js, which suggests an attempt to hide malicious payloads from security scanners. Additionally, there are several high-priority findings related to potential command injection vulnerabilities in Go code and disabled host key verification in Docker configurations that could allow attackers to compromise the system. These patterns look more like sophisticated obfuscation techniques used by malware authors than simple insecure coding errors, as they rely on encoding tricks to bypass standard security checks. However, because these issues are already flagged by multiple independent tools with high confidence scores, they represent real risks that should be addressed before deploying the software in a production environment.

Written by a language model running locally on the scanning machine, from the findings in this report. The risk levels above and below were decided by the scoring model, not by this summary.

Risk, by type

MalDet scores two different questions separately, because they call for different responses: code can be badly written without being hostile, and hostile code is not always badly written.

VULNERABILITY

Medium (30)

Insecure coding an attacker could exploit. Usually mistakes, and usually fixable in place.

MALICIOUS PATTERN

Medium (30)

Signs the code may be trying to do harm: hidden payloads, credential access, persistence, exfiltration.

What each detector found

DETECTOR	LOOKS FOR	HIGH	MEDIUM	LOW	TOTAL
semgrep	Insecure coding patterns across many languages	2	14	0	16
guarddog	Malicious package and install-time behaviour	1	6	3	10
yara	Signatures of malicious behaviour in source files	6	0	1	7
dep_checker	Vulnerable, unpinned or typosquatted dependencies	0	0	1	1

23 of 34 findings were rated more likely genuine than noise by the classifier.

Evidence

Identical findings are grouped, so a single rule firing many times takes one row. Ordered worst first. "Seen" is how many times that finding occurred.

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
HIGH	guarddog	Detects base64 decoding followed by code execution [threat-runtime-obfuscation-base64exec]	1	/pkg/web/static/vendor/echarts/echarts.min.js:45
HIGH	semgrep	Detected non-static command inside Command. Audit the input to 'exec.Command'. If unverified user data can reach this call site, this is a code injection vulnerability. A malicious actor can inject a malicious script to execute arbitrary code.	1	/internal/fail2ban/connector_local.go:274
HIGH	semgrep	By not specifying a USER, a program in the container may run as 'root'. This is a security hazard. If an attacker can control a process running as root, they may have control over the container. Ensure that the last USER in a Dockerfile is a USER other than 'root'.	1	/Dockerfile:81
HIGH	yara	YARA rule matched: detect_base64_exec — Encoded payload decoded and then executed	2	/pkg/web/static/vendor/echarts/echarts.min.js

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
HIGH	yara	YARA rule matched: detect_obfuscated_string_execution — Reconstructs a string character-by-character or from hex escapes, then immediately executes it - fromCharCode/chr() alone are common in ordinary text-processing code, so this requires both the reconstruction and an execution call together	3	/pkg/web/static/vendor/jquery/jquery-3.6.0.min.js
HIGH	yara	YARA rule matched: detect_persistence_mechanism — Modifies a startup/scheduling location - legitimate application code essentially never does this	1	/deployment/systemd/README.md
MEDIUM	semgrep	RegExp() called with a `query` function argument, this might allow an attacker to cause a Regular Expression Denial-of-Service (ReDoS) within your application as RegExp blocks the main thread. For this reason, it is recommended to use hardcoded regexes instead. If your regex is run on user-controlled	1	/pkg/web/static/js/Utils.js:119
MEDIUM	guarddog	Detects unicode homoglyphs and uncommon characters used for obfuscation [threat-runtime-obfuscation-unicode]	1	/pkg/web/static/vendor/globe/globe.globals.in.js:20
MEDIUM	guarddog	Detects URLs to URL shorteners, file sharing, and suspicious services [threat-network-outbound-shady-links]	4	/pkg/web/static/js/core.js:235
MEDIUM	guarddog	Detects log/console suppression combined with obfuscated code, a common malware evasion pattern [threat-runtime-obfuscation-log-suppress]	1	/pkg/web/static/vendor/globe/globe.globals.in.js:5
MEDIUM	semgrep	Disabled host key verification detected. This allows man-in-the-middle attacks. Use the 'golang.org/x/crypto/ssh/knownhosts' package to do host key verification. See https://skarlso.github.io/2019/02/17/go-ssh-with-host-key-verification/ to learn more about the problem and how to fix it.	1	/internal/integrations/mikrotik.go:193

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	semgrep	A session cookie was detected without setting the 'Secure' flag. The 'secure' flag for cookies prevents the client from transmitting the cookie over insecure channels such as HTTP. Set the 'Secure' flag by setting 'Secure' to 'true' in the Options struct.	4	/pkg/web/handlers_auth.go:92
MEDIUM	semgrep	String-formatted SQL query detected. This could lead to SQL injection if the string is not sanitized properly. Audit this call to ensure the SQL is not manipulable by external data.	4	/internal/storage/storage.go:1013
MEDIUM	semgrep	`MinVersion` is missing from this TLS configuration. By default, as of Go 1.22, TLS 1.2 is currently used as the minimum. General purpose web applications should default to TLS 1.3 with all other protocols disabled. Only where it is known that a web server must support legacy clients with unsuppor	2	/pkg/web/handlers_email.go:260
MEDIUM	semgrep	GitHub Actions step uses a mutable tag or branch reference. Tags and branch names can be silently repointed by the action owner, enabling supply-chain attacks — as seen in the trivy-action and kics-github-action compromises. Pin the reference to a full 40-character commit SHA instead, e.g. `uses: ac	1	/.gitea/workflows/build-harbor.yml:12
MEDIUM	semgrep	Checks for disabling of TLS/SSL certificate verification. This should only be used for debugging purposes because it leads to vulnerability to MTM attacks.	1	/internal/httpx/httpx.go:34
LOW	guarddog	Detects reading of environment variables (credential access, often contains secrets) [threat-runtime-environment-read]	3	/internal/config/settings.go:693
LOW	dep_checker	Loose version constraint: 'tailwindcss^3.4.19' — attacker could push malicious update	1	package.json
LOW	yara	YARA rule matched: detect_downloader — Downloads a remote resource - common and low-signal alone; see detect_download_and_execute for the composite version	1	/Dockerfile

How to read this report

- **This is a static scan.** The code was read, never executed. Behaviour that only appears at runtime is outside what any of these detectors can see.
- **Findings are not all equally real.** A classifier rates each finding on how likely it is to be genuine rather than noise, and that rating weights the scores. Measured against a hand-reviewed sample it reached precision 0.52 and recall 0.62 — useful for ranking, not a verdict on any single finding.
- **A high score is not proof of malice.** Security tools, penetration-testing frameworks and malware research repositories contain the same patterns as real malware by design. The two risk types above exist to keep that distinction visible.
- **Absence of findings is not proof of safety.** A detector that is not installed, or a language none of them parse, produces silence rather than a warning.
- **This scan is a snapshot.** The repository may have changed since 24 September 2026.