

prysm

github.com/OffchainLabs/prysm

Scanned 24 September 2026 at 08:55 · Report generated 29 October 2026 · Go · 3,782 stars

OVERALL ASSESSMENT

High

Combined score 175 · 86 findings after filtering

Summary

This repository has a high vulnerability risk score of 172.0, which means there are several serious coding errors that an attacker could easily exploit to break your system. The most critical issues involve using unencrypted connections for gRPC and WebSocket traffic, as well as running shell commands directly from the internet without any safety checks. While some findings appear to be malicious patterns like downloading files and executing them, the code also contains many common mistakes such as missing locks on variables and unsafe variable interpolation that could lead to security flaws. The low malicious-pattern score of 3.0 suggests that these issues are mostly due to poor coding habits rather than intentional attempts to harm the project. Because of these significant vulnerabilities, this repository should not be used in a production environment without careful review and remediation by a security expert.

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

High (172)

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

MALICIOUS PATTERN

Low (3)

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	9	60	1	70
yara	Signatures of malicious behaviour in source files	4	0	11	15
guarddog	Malicious package and install-time behaviour	0	1	0	1

54 of 86 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	semgrep	Found an insecure gRPC connection using 'grpc.WithInsecure()'. This creates a connection without encryption to a gRPC server. A malicious attacker could tamper with the gRPC message, which could compromise the machine. Instead, establish a secure connection with an SSL certificate using the 'grpc.Wi	3	/tools/forkchecker/forkchecker.go:50
HIGH	semgrep	A `run:` step pipes the output of `curl` or `wget` directly into a shell interpreter. This is the "curl bash" install pattern — if the remote server is compromised or the URL is hijacked, an attacker can execute arbitrary code in your CI runner. Consider downloading the file first, verifying its c	1	/.github/workflows/horusec.yaml:20
HIGH	semgrep	Missing `RUnlock` on an `RWMutex` (`s.headLock` variable) lock before returning from a function	1	/beacon-chain/blockchain/service.go:249
HIGH	semgrep	Untrusted input could be used to tamper with a web page rendering, which can lead to a Cross-site scripting (XSS) vulnerability. XSS vulnerabilities occur when untrusted input executes malicious JavaScript code, leading to issues such as account compromise and sensitive information leakage. To preve	1	/validator/web/handler.go:34

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
HIGH	semgrep	Using variable interpolation `\${{...}}` with `github` context data in a `run:` step could allow an attacker to inject their own code into the runner. This would allow them to steal secrets and code. `github` context data can have arbitrary user input and should be treated as untrusted. Instead, use	1	/.github/workflows/sbom-export.yaml:57
HIGH	semgrep	Insecure WebSocket Detected. WebSocket Secure (wss) should be used for all WebSocket connections.	1	/cmd/helpers.go:64
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	/runtime/prereqs/prereq.go:29
HIGH	yara	YARA rule matched: detect_download_and_execute — Downloads a remote resource AND executes it - the actually dangerous combination, not just downloading alone	3	/.github/workflows/horusec.yaml
HIGH	yara	YARA rule matched: detect_persistence_mechanism — Modifies a startup/scheduling location - legitimate application code essentially never does this	1	/tools/cross-toolchain/Dockerfile
MEDIUM	semgrep	Variable `sum` is likely modified and later used on error. In some cases this could result in panics due to a nil dereference	1	/beacon-chain/db/filesystem/pruner.go:47
MEDIUM	semgrep	Variable `b` is likely modified and later used on error. In some cases this could result in panics due to a nil dereference	1	/beacon-chain/sync/backfill/service.go:156

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	semgrep	Data is being eval'd from a `curl` command. An attacker with control of the server in the `curl` command could inject malicious code into the `eval`, resulting in a system compromise. Avoid eval'ing untrusted data if you can. If you must do this, consider checking the SHA sum of the content returned	1	/hack/upload-github-release-asset.sh:58
MEDIUM	semgrep	Iteration over a possibly empty map `subnetsRequiringPeers`. This is likely a bug or redundant code	1	/beacon-chain/sync/subscriber.go:913
MEDIUM	guarddog	Detects URLs to URL shorteners, file sharing, and suspicious services [threat-network-outbound-shady-links]	1	/tools/analyzers/maligned/maligned.go:31
MEDIUM	semgrep	Detected 'Fprintf' or similar writing to 'http.ResponseWriter'. This bypasses HTML escaping that prevents cross-site scripting vulnerabilities. Instead, use the 'html/template' package to render data to users.	4	/validator/rpc/handlers_health.go:148
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	33	/.github/workflows/go.yml:75
MEDIUM	semgrep	When working with web applications that involve rendering user-generated content, it's important to properly escape any HTML content to prevent Cross-Site Scripting (XSS) attacks. In Go, the `text/template` package does not automatically escape HTML content, which can leave your application vuln	3	/tools/beacon-fuzz/main.go:11
MEDIUM	semgrep	The special variable IFS affects how splitting takes place when expanding unquoted variables. Don't set it globally. Prefer a dedicated utility such as 'cut' or 'awk' if you need to split input data. If you must use 'read', set IFS locally using e.g. 'IFS=',' read -a my_array'.	1	/hack/codecov.sh:1122

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	semgrep	Detected directly writing or similar in 'http.ResponseWriter.write()'. This bypasses HTML escaping that prevents cross-site scripting vulnerabilities. Instead, use the 'html/template' package and render data using 'template.Execute()'.	8	/monitoring/prometheus/content_negotiation.go:43
MEDIUM	semgrep	If an attacker can supply values that the application then uses to determine which method or field to invoke, the potential exists for the attacker to create control flow paths through the application that were not intended by the application developers. This attack vector may allow the attacker to	2	/encoding/ssz/query/proof_collector.go:278
MEDIUM	semgrep	Do not use `math/rand`. Use `crypto/rand` instead.	5	/tools/analyzers/cryptorand/testdata/rand_new.go:4
LOW	semgrep	Detected stack traces being printed or included in an HTTP response. This could expose sensitive information if deployed to a production environment in a user facing manner.	1	/monitoring/prometheus/service.go:122
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	11	/third_party/go-bip39/wordlists/chinese_simplified.go

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.