

eks-anywhere

github.com/aws/eks-anywhere

Scanned 24 September 2026 at 07:51 · Report generated 29 October 2026 · Go · 2,099 stars

OVERALL ASSESSMENT

High

Combined score 294 · 140 findings after filtering

Summary

The repository has a high vulnerability score of 291.0, which indicates that the code contains many dangerous patterns an attacker could easily use to break things or steal data. While the malicious-pattern score is low at 3.0, suggesting the code isn't intentionally trying to harm others, the high vulnerability score means there are serious security flaws that need immediate attention. Most of these issues stem from common mistakes like forgetting to unlock locks or hardcoding secrets, rather than deliberate attempts to bypass security controls. Because some findings have a very low confidence level, they might be false positives caused by how the scanner works instead of real problems in your codebase.

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 (291)

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	17	77	11	105

DETECTOR	LOOKS FOR	HIGH	MEDIUM	LOW	TOTAL
yara	Signatures of malicious behaviour in source files	25	0	7	32
guarddog	Malicious package and install-time behaviour	0	0	3	3

78 of 140 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	Missing mutex unlock (`pc.mu` variable) before returning from a function. This could result in panics resulting from double lock operations	2	/pkg/curatedpackages/packagecontrollerclient.go:161
HIGH	semgrep	AWS Secret Access Key detected	10	/designs/snow-integration.md:126
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	/pkg/executables/executables.go:107
HIGH	semgrep	Private Key detected. This is a sensitive credential and should not be hardcoded here. Instead, store this in a separate, private file.	3	/pkg/executables/testdata/nonemptyprivatekey:1
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	/designs/Dockerfile:18

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
HIGH	yara	YARA rule matched: detect_download_and_execute — Downloads a remote resource AND executes it - the actually dangerous combination, not just downloading alone	2	/Makefile
HIGH	yara	YARA rule matched: detect_persistence_mechanism — Modifies a startup/scheduling location - legitimate application code essentially never does this	23	/pkg/ providers/ nutanix/ testdata/ expected_results_external_etcd.yaml
MEDIUM	semgrep	Variable `result` is likely modified and later used on error. In some cases this could result in panics due to a nil dereference	12	/pkg/ executables/ cmk.go:155
MEDIUM	semgrep	Variable `stopResult` is likely modified and later used on error. In some cases this could result in panics due to a nil dereference	1	/pkg/ executables/ cmk.go:451
MEDIUM	semgrep	Variable `destroyResult` is likely modified and later used on error. In some cases this could result in panics due to a nil dereference	1	/pkg/ executables/ cmk.go:457
MEDIUM	semgrep	Pod is sharing the host process ID namespace. When paired with ptrace this can be used to escalate privileges outside of the container. Remove the 'hostPID' key to disable this functionality.	3	/pkg/ nodeupgrader/ testdata/ expected_result_control_plane_upgrader_pod.yaml:25
MEDIUM	semgrep	This Dependabot configuration does not set a cooldown period. Newly published packages can be malicious or unstable. Add a `cooldown` block with `default-days: 7` to each `package-ecosystem` entry under `updates` to wait 7 days before proposing updates to newly published package versions. Reference:	2	/.github/ dependabot.yaml:6
MEDIUM	semgrep	Pod may use the node network namespace. This gives the pod access to the loopback device, services listening on localhost, and could be used to snoop on network activity of other pods on the same node. Remove the 'hostNetwork' key to disable this functionality.	4	/pkg/ awsiamauth/ testdata/ UpgradeAWSIAMAuth-manifest.yaml:93

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	semgrep	A session cookie was detected without setting the 'HttpOnly' flag. The 'HttpOnly' flag for cookies instructs the browser to forbid client-side scripts from reading the cookie which mitigates XSS attacks. Set the 'HttpOnly' flag by setting 'HttpOnly' to 'true' in the Cookie.	1	/cmd/eks-a-tool/cmd/vsphereologoutsessions.go:97
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	/pkg/certificates/ssh.go:84
MEDIUM	semgrep	Detected SHA1 hash algorithm which is considered insecure. SHA1 is not collision resistant and is therefore not suitable as a cryptographic signature. Use SHA256 or SHA3 instead.	1	/internal/pkg/oidc/server.go:44
MEDIUM	semgrep	Container or pod is running in privileged mode. This grants the container the equivalent of root capabilities on the host machine. This can lead to container escapes, privilege escalation, and other security concerns. Remove the 'privileged' key to disable this capability.	4	/pkg/nodeupgrader/testdata/expected_first_control_plane_upgrade_r_pod.yaml:24
MEDIUM	semgrep	Detected MD5 hash algorithm which is considered insecure. MD5 is not collision resistant and is therefore not suitable as a cryptographic signature. Use SHA256 or SHA3 instead.	1	/internal/thirdparty/capc/api/v1beta3/cloudstackfaileddomain_types.go:32
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 unsupported	10	/pkg/registry/client.go:47

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.	1	/cmd/eks-a-tool/cmd/vsphereologoutsessions.go:97
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	13	/.github/workflows/golangci-lint.yml:20
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	5	/pkg/templater/templater.go:7
MEDIUM	semgrep	Deserializing into `interface{}` allows arbitrary data structures and types, which can lead to security vulnerabilities (CWE-502). Use a concrete struct type instead.	3	/release/cli/pkg/signature/manifest.go:211
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/cherry_pick_pull.sh:85
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.	4	/pkg/certificates/scanner.go:154
MEDIUM	semgrep	Detected a possible denial-of-service via a zip bomb attack. By limiting the max bytes read, you can mitigate this attack. `io.CopyN()` can specify a size.	1	/pkg/tar/untar.go:57

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	semgrep	Do not use <code>`math/rand`</code> . Use <code>`crypto/rand`</code> instead.	2	/pkg/networkutils/ipgenerator.go:5
MEDIUM	semgrep	In Kubernetes, each pod runs in its own isolated environment with its own set of security policies. However, certain container images may contain <code>`setuid`</code> or <code>`setgid`</code> binaries that could allow an attacker to perform privilege escalation and gain access to sensitive resources. To mitigate this risk,	2	/pkg/networking/cilium/testdata/cilium_manifest.yaml:499
MEDIUM	semgrep	In Kubernetes, each pod runs in its own isolated environment with its own set of security policies. However, certain container images may contain <code>`setuid`</code> or <code>`setgid`</code> binaries that could allow an attacker to perform privilege escalation and gain access to sensitive resources. To mitigate this risk,	1	/pkg/nodeupgrader/testdata/expected_rest_control_plane_upgrader_pod.yaml:23
MEDIUM	semgrep	In Kubernetes, each pod runs in its own isolated environment with its own set of security policies. However, certain container images may contain <code>`setuid`</code> or <code>`setgid`</code> binaries that could allow an attacker to perform privilege escalation and gain access to sensitive resources. To mitigate this risk,	1	/pkg/nodeupgrader/testdata/expected_first_control_plane_upgrade_r_pod.yaml:23
MEDIUM	semgrep	In Kubernetes, each pod runs in its own isolated environment with its own set of security policies. However, certain container images may contain <code>`setuid`</code> or <code>`setgid`</code> binaries that could allow an attacker to perform privilege escalation and gain access to sensitive resources. To mitigate this risk,	1	/config/manager/manager_image_patch.yaml:11
MEDIUM	semgrep	In Kubernetes, each pod runs in its own isolated environment with its own set of security policies. However, certain container images may contain <code>`setuid`</code> or <code>`setgid`</code> binaries that could allow an attacker to perform privilege escalation and gain access to sensitive resources. To mitigate this risk,	1	/pkg/networking/cilium/testdata/cilium_manifest.yaml:742

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
LOW	guarddog	Detects reading of environment variables (credential access, often contains secrets) [threat-runtime-environment-read]	3	/pkg/certificates/config.go:55
LOW	semgrep	When running containers in Kubernetes, it's important to ensure that they are properly secured to prevent privilege escalation attacks. One potential vulnerability is when a container is allowed to run applications as the root user, which could allow an attacker to gain access to sensitive resou	11	/pkg/networking/cilium/testdata/cilium_manifest.yaml:402
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	7	/hack/cherry_pick_pull.sh

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.