

mlflow

github.com/mlflow/mlflow

Scanned 29 July 2026 at 15:18 · Report generated 29 October 2026 · Python · 25,068 stars

OVERALL ASSESSMENT

Critical

Combined score 1658 · 33,923 findings after filtering

Summary

This repository has two critical security issues that are likely caused by ordinary coding mistakes rather than malicious intent. The most serious problem involves the use of FTP, which is an insecure protocol that attackers can exploit to steal data or access files; developers should switch this to a secure method like SFTP or SSH immediately. Additionally, there are 33,923 findings related to command injection vulnerabilities where code runs user-supplied commands directly, which could allow attackers to execute arbitrary code and steal secrets if the input is not properly validated. While some dependencies have unversioned versions that pose supply chain risks, these are less urgent than the immediate threats of insecure protocols and dangerous command execution patterns.

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

Critical (1536)

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

MALICIOUS PATTERN

Critical (122)

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
bandit	Insecure coding patterns in Python source	9	588	33006	33,603

DETECTOR	LOOKS FOR	HIGH	MEDIUM	LOW	TOTAL
semgrep	Insecure coding patterns across many languages	37	133	16	186
dep_checker	Vulnerable, unpinned or typosquatted dependencies	3	44	33	80
yara	Signatures of malicious behaviour in source files	28	0	26	54

779 of 33,923 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	A formatted or concatenated string was detected as input to a ProcessBuilder call. This is dangerous if a variable is controlled by user input and could result in a command injection. Ensure your variables are not controlled by users or sufficiently sanitized.	1	/mlflow/java/client/src/main/java/org/mlflow/artifacts/CliBasedArtifactRepository.java
HIGH	semgrep	Found 'importlib.resources', which is a module only available on Python 3.7+. This does not work in lower versions, and therefore is not backwards compatible. Use importlib_resources instead for older Python versions.	4	/mlflow/utils/providers.py
HIGH	dep_checker	Typosquatting: 'pymssql' is suspiciously similar to 'pymysql'	1	pyproject.toml
HIGH	dep_checker	Typosquatting: 'cryptography<51,' is suspiciously similar to 'cryptography'	1	pyproject.toml
HIGH	dep_checker	Typosquatting: 'sqlalchemy<3,' is suspiciously similar to 'sqlalchemy'	1	pyproject.toml
HIGH	semgrep	Detected calls to child_process from a function argument `cmd`. This could lead to a command injection if the input is user controllable. Try to avoid calls to child_process, and if it is needed ensure user input is correctly sanitized or sandboxed.	2	/.github/workflows/uv.js

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
HIGH	bandit	FTP-related functions are being called. FTP is considered insecure. Use SSH/SFTP/SCP or some other encrypted protocol.	1	/mlflow/store/artifact/ftp_artifact_repo.py
HIGH	bandit	A FTP-related module is being imported. FTP is considered insecure. Use SSH/SFTP/SCP or some other encrypted protocol.	4	/mlflow/store/artifact/ftp_artifact_repo.py
HIGH	bandit	tarfile.extractall used without any validation. Please check and discard dangerous members.	4	/mlflow/pyfunc/dbconnect_artifact_cache.py
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	2	/.github/actions/setup-java/action.yml
HIGH	semgrep	sqlalchemy.text passes the constructed SQL statement to the database mostly unchanged. This means that the usual SQL injection protections are not applied and this function is vulnerable to SQL injection if user input can reach here. Use normal SQLAlchemy operators (such as `or_()`, `and_()`, etc.)	16	/mlflow/store/tracking/dbmodels/models.py
HIGH	semgrep	Generic API Key detected	1	/docs/docusaurus.config.ts
HIGH	semgrep	Functions reliant on pickle can result in arbitrary code execution. Consider loading from `state_dict`, using fickling, or switching to a safer serialization method like ONNX	7	/mlflow/pytorch/__init__.py
HIGH	semgrep	Possible path traversal through `tarfile.open(\$PATH).extractall()` if the source tar is controlled by an attacker	1	/examples/flower_classifier/train.py

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
HIGH	yara	YARA rule matched: detect_credential_harvester — References credential/secret material by name AND a deliberate outbound-send API in the same file - narrower than 'reads any env var', which large legitimate codebases do constantly regardless of intent. Still not proof the two are connected (YARA c	6	/dev/ update_model_ catalog.py
HIGH	yara	YARA rule matched: detect_base64_exec — Encoded payload decoded and then executed	6	/mlflow/utils/ search_utils. py
HIGH	semgrep	User controlled data in methods like `innerHTML`, `outerHTML` or `document.write` is an anti-pattern that can lead to XSS vulnerabilities	3	/libs/ typescript/ integrations/ openai/docs/ assets/main.js
HIGH	yara	YARA rule matched: detect_download_and_execute — Downloads a remote resource AND executes it - the actually dangerous combination, not just downloading alone	14	/dev/ run_dev_server .py
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	2	/mlflow/ server/js/ yarn/releases/ yarn-4.12.0.c js
MEDIUM	semgrep	This GitHub Actions workflow file uses `pull_request_target` and checks out code from the incoming pull request. When using `pull_request_target`, the Action runs in the context of the target repository, which includes access to all repository secrets. Normally, this is safe because the Action only	1	/.github/ workflows/ui- preview.yml

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	semgrep	RegExp() called with a `shouldBeBold` 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-co	2	/mlflow/server/js/src/experiment-tracking/components/EntitySearchAutoComplete.utils.tsx
MEDIUM	semgrep	RegExp() called with a `routePath` 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-contr	1	/mlflow/server/js/src/experiment-tracking/components/experiment-page/hooks/useGetExperimentPageRoute.test.tsx
MEDIUM	semgrep	RegExp() called with a `variableName` 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-co	1	/mlflow/server/js/src/experiment-tracking/pages/experiment-scorers/utils/templateUtils.ts
MEDIUM	semgrep	RegExp() called with a `searchQuery` 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-con	1	/mlflow/server/js/src/shared/web-shared/genai-traces-table/utils/DisplayUtils.tsx
MEDIUM	semgrep	RegExp() called with a `runId` 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-controlle	1	/mlflow/server/js/src/model-registry/utils/VersionUtils.ts

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	semgrep	Insecure HostnameVerifier implementation detected. This will accept any SSL certificate with any hostname, which creates the possibility for man-in-the-middle attacks.	1	/mlflow/java/client/src/main/java/org/mlflow/tracking/MlflowHttpClient.java
MEDIUM	semgrep	These permissions `0o750` are widely permissive and grant access to more people than may be necessary. A good default is `0o644` which gives read and write access to yourself and read access to everyone else.	3	/mlflow/pyfunc/__init__.py
MEDIUM	semgrep	Found a template created with string formatting. This is susceptible to server-side template injection and cross-site scripting attacks.	2	/mlflow/server/auth/__init__.py
MEDIUM	semgrep	Detects direct creations of SSLConnectionSocketFactories that don't disallow SSL v2, SSL v3, and TLS v1. SSLSocketFactory can be used to validate the identity of the HTTPS server against a list of trusted certificates. These protocols are deprecated due to POODLE, man in the middle attacks, and othe	1	/mlflow/java/client/src/main/java/org/mlflow/tracking/MlflowHttpClient.java
MEDIUM	semgrep	If possible, it is better to rely on automatic pinning in PyTorch to avoid undefined behavior and for efficiency	9	/examples/pytorch/HPOExample/hpo_mnist.py
MEDIUM	semgrep	RegExp() called with a `value` 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-controlle	4	/mlflow/server/js/src/experiment-tracking/components/evaluation-artifacts-compare/components/EvaluationArtifactCompareTable.test.tsx
MEDIUM	bandit	Use of unsafe yaml load. Allows instantiation of arbitrary objects. Consider yaml.safe_load().	1	/mlflow/utils/yaml_utils.py

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	bandit	Chmod setting a permissive mask 0o750 on file (tmp_nfs_dir).	1	/mlflow/utils/file_utils.py
MEDIUM	bandit	Chmod setting a permissive mask 0o750 on file (tmp_model_dir).	1	/mlflow/pyfunc/__init__.py
MEDIUM	bandit	Chmod setting a permissive mask 0o750 on file (tmp_dir).	1	/mlflow/utils/file_utils.py
MEDIUM	semgrep	Service 'minio-create-bucket' allows for privilege escalation via setuid or setgid binaries. Add 'no-new-privileges:true' in 'security_opt' to prevent this.	1	/examples/mlflow_artifacts/docker-compose.yml
MEDIUM	semgrep	Service 'minio' allows for privilege escalation via setuid or setgid binaries. Add 'no-new-privileges:true' in 'security_opt' to prevent this.	1	/examples/mlflow_artifacts/docker-compose.yml
MEDIUM	semgrep	If an attacker can supply values that the application then uses to determine which class to instantiate or which method 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 a	1	/mlflow/java/client/src/main/java/org/mlflow/tracking/Utils/DatabricksContext.java
MEDIUM	semgrep	Be careful with `flask.make_response()`. If this response is rendered onto a webpage, this could create a cross-site scripting (XSS) vulnerability. `flask.make_response()` will not autoescape HTML. If you are rendering HTML, write your HTML in a template file and use `flask.render_template()` which	22	/mlflow/server/auth/__init__.py
MEDIUM	semgrep	Service 'minio' is running with a writable root filesystem. This may allow malicious applications to download and run additional payloads, or modify container files. If an application inside a container has to save something temporarily consider using a tmpfs. Add 'read_only: true' to this service t	1	/examples/mlflow_artifacts/docker-compose.yml

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	semgrep	Service 'minio-create-bucket' is running with a writable root filesystem. This may allow malicious applications to download and run additional payloads, or modify container files. If an application inside a container has to save something temporarily consider using a tmpfs. Add 'read_only: true' to	1	/examples/mlflow_artifacts/docker-compose.yml
MEDIUM	semgrep	Service 'postgres' allows for privilege escalation via setuid or setgid binaries. Add 'no-new-privileges:true' in 'security_opt' to prevent this.	1	/examples/mlflow_artifacts/docker-compose.yml
MEDIUM	semgrep	Service 'postgres' is running with a writable root filesystem. This may allow malicious applications to download and run additional payloads, or modify container files. If an application inside a container has to save something temporarily consider using a tmpfs. Add 'read_only: true' to this servic	1	/examples/mlflow_artifacts/docker-compose.yml
MEDIUM	semgrep	RegExp() called with a `filter` 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-controll	1	/mlflow/server/js/src/experiment-tracking/components/runs-charts/components/sections/RunsChartsSectionAccordion.tsx
MEDIUM	dep_checker	Unpinned dependency: 'azure-storage-file-datalake' has no version lock — supply chain risk	1	pyproject.toml
MEDIUM	dep_checker	Unpinned dependency: 'fastapi' has no version lock — supply chain risk	3	pyproject.toml
MEDIUM	dep_checker	Unpinned dependency: 'uvicorn' has no version lock — supply chain risk	3	pyproject.toml
MEDIUM	dep_checker	Unpinned dependency: 'tiktoken' has no version lock — supply chain risk	2	pyproject.toml
MEDIUM	dep_checker	Unpinned dependency: 'botocore' has no version lock — supply chain risk	2	pyproject.toml

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	dep_checker	Unpinned dependency: 'watchfiles' has no version lock — supply chain risk	2	pyproject.toml
MEDIUM	dep_checker	Unpinned dependency: 'boto3' has no version lock — supply chain risk	2	pyproject.toml
MEDIUM	dep_checker	Unpinned dependency: 'pyarrow' has no version lock — supply chain risk	1	pyproject.toml
MEDIUM	dep_checker	Unpinned dependency: 'Flask-CORS' has no version lock — supply chain risk	1	pyproject.toml
MEDIUM	dep_checker	Unpinned dependency: 'matplotlib' has no version lock — supply chain risk	1	pyproject.toml
MEDIUM	dep_checker	Unpinned dependency: 'skops' has no version lock — supply chain risk	1	pyproject.toml
MEDIUM	dep_checker	Unpinned dependency: 'pymssql' has no version lock — supply chain risk	1	pyproject.toml
MEDIUM	dep_checker	Unpinned dependency: 'pandas' has no version lock — supply chain risk	1	pyproject.toml
MEDIUM	dep_checker	Unpinned dependency: 'starlette' has no version lock — supply chain risk	1	pyproject.toml
MEDIUM	dep_checker	Unpinned dependency: 'graphene' has no version lock — supply chain risk	1	pyproject.toml
MEDIUM	dep_checker	Unpinned dependency: 'scipy' has no version lock — supply chain risk	1	pyproject.toml

Showing the 60 most significant of 487 distinct findings. The counts in the previous section cover all of them.

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 29 July 2026.