Python: which everyday calls turn into vulnerabilities?

Before your next release, move off Python 3.10 and older[2], keep pickle-based loaders away from outside data, and run Ruff's S rules and `pip-audit` in CI. Judge everything else by whether input stays data: never code, a shell command, SQL text, a template or a path outside the directory you meant.

Reviewed	Next review	Version	Written for
5 Nov 2026	**5 Dec 2026**	**1.1**	**Developers and tech leads writing production Python, and the security people who review it**

Latest version
sheets.einblick.dev/secure-python/

If you only do five things

1. **Never load outside data with pickle-based loaders.** `pickle`, `shelve`, unsafe `yaml.load()` and `torch.load(weights_only=False)` can run code while loading[5,6,7]. Use `json`, `yaml.safe_load()` and safetensors instead.
2. **Pass data as arguments, never as code.** Bind SQL values as parameters, call `subprocess` with a list and no `shell=True`, and never render user text as a template. SQL injection ranks 2nd and OS command injection 9th in the 2025 CWE Top 25[1].
3. **Run a supported Python and patch it.** Python 3.10 reached end of life on 1 October 2026[2] and missed October's fix for compromised certificate authority keys[3]. Patch within the patching sheet's deadlines.
4. **Let tools find the obvious ones.** Run Ruff's S rules and `pip-audit` in CI. Ruff flags pickle, `yaml.load()`, `shell=True`, SQL built from strings, missing timeouts and `verify=False`[23].
5. **Validate at every boundary, at runtime.** Type hints aren't enforced at runtime[21]. Use pydantic models with `extra="forbid"` and strict types: by default pydantic ignores unknown fields and turns `"123"` into `123`[22].

Review a Python pull request

1. **Where does outside data enter?** Find request handlers, CLI arguments, queue consumers, uploads and downloaded models. Each should become a pydantic model or form before other code uses it.
2. **Does it reach a loader?** Search for `pickle`, `shelve`, `yaml.load`, `torch.load` and `from_pretrained`. Data from outside needs a format that can't run code.
3. **Does it reach something that runs strings?** Check SQL text, `subprocess`, `eval`, template source and `**` unpacking into ORM calls.
4. **Does it reach the file system?** Paths from input need `resolve()` plus `is_relative_to()`. Archives need `filter="data"`, an empty directory and size limits.
5. **Does it reach the network?** Calls need a timeout and certificate checks. URLs from users need an allowlist against server-side request forgery, 22nd in the CWE Top 25[1].

Python traps and what to use instead

- **`pickle.loads()`, `shelve` and `yaml.load()`.** Unpickling can run arbitrary code[5], and `shelve` and `multiprocessing`'s `Connection.recv()` use pickle[6]. Use `json`, and `yaml.safe_load()` for YAML[5].
- **`torch.load()` and `trust_remote_code=True`.** PyTorch has defaulted to `weights_only=True` since 2.6[7], but before 2.10.0 that mode could still run code[8]. Prefer safetensors[10]. `trust_remote_code=True` runs code from the Hub repository on your machine[9].
- **`eval()` and `ast.literal_eval()`.** Ruff's S307 suggests `literal_eval()` in place of `eval()`[23], but Python's docs advise against it for untrusted data: small inputs can exhaust memory or crash the process[11]. Use `json.loads()`.
- **`subprocess.run(cmd, shell=True)`.** With a shell, quoting is your job[12]. Pass a list such as `["convert", src, dst]` without `shell=True`[12], and put `--` before values that may start with a dash, which programs read as options.
- **SQL in f-strings, `%` or `.format()`.** Pass values as parameters (`%s` in psycopg) or in a Python 3.14 `t"..."` string, which psycopg 3.3 parameterizes; an `f` in its place pastes values into the SQL[13]. In Django, never unpack a user-supplied dict with `**` into QuerySet methods: that still allowed SQL injection in February 2026[14].
- **Templates built from user input.** `render_template_string()` or `from_string()` on user text can lead to remote code execution[15]. Pass input as variables to fixed template files, and set `autoescape=select_autoescape()`, since Jinja doesn't escape by default[16].
- **`tarfile.extractall()`.** Pass `filter="data"` before Python 3.14, which made it the default[17]. It isn't a sandbox: from June to October 2026 the PSF published seven advisories for archives that got past the filters[4]. Extract into a new empty directory with OS-level limits[17].
- **`os.path.join(base, name)` and `Path(base) / name`.** An absolute `name` discards `base`[18]. Resolve the result, then check `is_relative_to(base.resolve())`; on unresolved paths that check ignores `..`[18].
- **`requests.get(url)` without `timeout=`, or with `verify=False`.** Requests never times out unless told to, and `verify=False` accepts any certificate. It trusts the CA list in `certifi`, so keep certifi updated[19].
- **`assert` for checks.** `python -O` and `PYTHONOPTIMIZE=1` skip asserts; aiohttp's POST parsing could then loop forever (CVE-2025-69227)[20]. Raise exceptions for permission and input checks.

Tools that catch it

- **Ruff's S rules.** Add `"S"` to `extend-select` in `[tool.ruff.lint]`. Fix S301 (pickle), S506 (YAML), S602 (shell), S608 (SQL), S202 (tar), S113 (timeouts) and S311 (`random`) first[23].
- **Bandit for ML code.** B614 flags `torch.load()`[10]; Ruff has no such rule[23].
- **pip-audit.** It checks environments, requirements and, with `--locked`, `pylock.*.toml` files against PyPI or OSV advisories[24].

What you can skip

- **defusedxml for standard library parsers.** Expat doesn't read local files or open connections by default, and Python's XML docs warn of entity expansion only for Expat before 2.7.2[27]. Check `pyexpat.EXPAT_VERSION` in CI and turn off S313 to S319[23].
- **Noisy lint rules.** Turn off S603, which fires on every `subprocess` call without a shell, and preview rules S403 and S404, which fire on any import of `pickle` or `subprocess`[23].
- **A `find_class` allowlist for pickle.** Change the format. PyTorch's own restricted unpickler needed a code execution fix in 2026[8].

What changed in 2026

- **Python 3.10 is end of life.** Support ended on 1 October 2026, and 3.15.0 shipped on 9 October[2].
- **A CA compromise reached Python's trust stores.** After October's emSign incident, install certifi 2026.10.19, pip 26.2.2 and uv 0.13.6 or later in every virtual environment, move to Python 3.11.18, 3.12.16, 3.13.18, 3.14.10 or 3.15.2 and rebuild images without cached layers[3].

Mistakes that cost the most

- **Shipping with the debugger on.** Flask's debugger runs Python sent from the browser[28]. Django's `DEBUG = True` leaks source excerpts and settings; run `manage.py check --deploy` in CI[29].
- **Trusting FastAPI's `OAuth2PasswordBearer`.** It only extracts the token. Verify the signature, `exp`, `iss` and `aud` with PyJWT and a fixed `algorithms` list[22].
- **Ignoring `.pth` files.** Their `import` lines run at every interpreter start, as do the `.start` files Python 3.15 adds to replace them[25]. The malicious LiteLLM 1.107.0 shipped `cook.pth` and `sitecustomize.py`[26]. The supply-chain sheet covers the rest.

What this looks like for your code

New project

- Start on Python 3.14 or 3.15, where tar extraction uses the `data` filter by default[17].
- Turn on Ruff's S rules and pip-audit in the first CI pipeline.
- Give every input type a pydantic model with `extra="forbid"`.

Existing codebase

- Move every image off Python 3.10 or older first.
- Fix Ruff's S301, S506, S602 and S608 hits in code that handles outside input first.
- Set `TORCH_FORCE_WEIGHTS_ONLY_LOAD=1` wherever models load[7].

High-assurance or regulated

- Block new CodeQL `security-extended` alerts; private code needs paid Code Security[30].
- Parse untrusted archives and models in a separate process with OS-level limits[17].
- Subscribe to security-announce@python.org and close every advisory[4].

Version history

5 Nov 2026, version 1.1 Reviewed for company readers: CodeQL on private code needs a paid GitHub Code Security license.

Sources

1. 2025 CWE Top 25 Most Dangerous Software Weaknesses. MITRE, updated 15 Dec 2025. cwe.mitre.org
2. Status of Python versions. Python Developer's Guide, checked 3 Nov 2026. devguide.python.org
3. Information for Python users regarding eMudhra / emSign CA certificate security incident. Python Discourse, 20 Oct 2026, updated 31 Oct 2026. discuss.python.org
4. security-announce@python.org advisory archive. PSF, checked 3 Nov 2026. mail.python.org
5. Deserialization Cheat Sheet. OWASP, checked 3 Nov 2026. cheatsheetseries.owasp.org
6. Security Considerations. Python docs, updated 2 Nov 2026. docs.python.org
7. Serialization semantics. PyTorch docs (2.15), updated 8 May 2026. docs.pytorch.org
8. Loading a malicious PyTorch checkpoint with weights_only=True can result in arbitrary code execution. PyTorch advisory, 26 Jan 2026. github.com
9. Auto classes (trust_remote_code). Hugging Face docs, checked 3 Nov 2026. huggingface.co
10. B614: pytorch_load. Bandit docs, checked 3 Nov 2026. bandit.readthedocs.io
11. ast: Abstract syntax trees (literal_eval). Python docs, updated 2 Nov 2026. docs.python.org
12. subprocess: Security Considerations. Python docs, updated 2 Nov 2026. docs.python.org
13. Template string queries. psycopg docs, checked 3 Nov 2026. psycopg.org
14. Django security releases issued: 6.0.2, 5.2.11, and 4.2.28. Django Project, 3 Feb 2026. djangoproject.com
15. Server Side Template Injection Prevention Cheat Sheet. OWASP, checked 3 Nov 2026. cheatsheetseries.owasp.org
16. API: Autoescaping. Jinja docs, checked 3 Nov 2026. jinja.palletsprojects.com
17. tarfile: Extraction filters. Python docs, updated 2 Nov 2026. docs.python.org
18. pathlib: Object-oriented filesystem paths. Python docs, updated 2 Nov 2026. docs.python.org
19. Advanced Usage. Requests docs, checked 3 Nov 2026. requests.readthedocs.io
20. DoS when bypassing asserts. aiohttp security advisory, 5 Jan 2026. github.com
21. typing: Support for type hints. Python docs, updated 2 Nov 2026. docs.python.org
22. FastAPI Security Cheat Sheet. OWASP, checked 3 Nov 2026. cheatsheetseries.owasp.org
23. Rules: flake8-bandit (S). Ruff docs, checked 3 Nov 2026. docs.astral.sh
24. pip-audit. PyPA, version 2.10.1, Jun 2026. pypi.org
25. site: Site-specific configuration hook. Python docs, updated 2 Nov 2026. docs.python.org
26. Contained: Oct 21, 2026 LiteLLM release v1.107.0 Compromised. LiteLLM, 21 Oct 2026. docs.litellm.ai
27. XML processing modules: XML security. Python docs, updated 2 Nov 2026. docs.python.org
28. Debugging Application Errors. Flask docs, checked 3 Nov 2026. flask.palletsprojects.com
29. Deployment checklist. Django docs (6.1), checked 3 Nov 2026. docs.djangoproject.com
30. CodeQL CLI. GitHub Docs, checked 5 Nov 2026. docs.github.com