TrustGet

Secure File Downloader for the Command Line

Project Blueprint • v1.0

Built for: SysAdmin • DevOps • Homelab • Linux Dev

> ***"wget yang punya otak keamanan."***
> Trustget = download + verify + trust analysis — satu perintah, nol drama.

1. Vision & Core Philosophy

Trustget lahir dari satu masalah nyata: download file dari internet itu mudah, tapi aman adalah cerita lain. Sysadmin harus manual cek SHA256, cross-reference signature, lalu bertanya-tanya apakah release ini legitimate. Trustget mengotomatiskan seluruh alur ini.

Filosofi Desain

Prinsip	Implementasi	Trade-off
Zero Config	Deteksi otomatis checksum, platform, dan signature	Kadang bisa salah deteksi → fallback manual tersedia
One Command	sg <url> cukup untuk full verification	Feature advanced via subcommand
Transparent	Tampilkan setiap langkah verifikasi	Output lebih verbose dari wget biasa
Composable	Exit code standar, JSON output support	Tidak cocok untuk scripting sederhana tanpa --quiet
Lightweight	Pure Python, deps minimal	Tidak ada GUI, hanya terminal

Target User Persona

Persona	Use Case	Pain Point Solved
SysAdmin Enterprise	Download tools ke server production	Manual verify SHA256 buang waktu
DevOps Engineer	Pipeline CI/CD butuh artifact dari GitHub	Tidak ada verifikasi otomatis di

Persona	Use Case	Pain Point Solved
	Releases	wget/curl
Homelab Enthusiast	ISO, AppImage, binary dari sumber tidak familiar	Tidak tahu cara cek GPG signature
Security Engineer	Audit toolchain yang dipakai tim	Trust analysis terpusat, bisa log ke SIEM

2. Arsitektur Proyek

Struktur repo dirancang untuk maintainability jangka panjang. Setiap modul punya tanggung jawab tunggal (Single Responsibility), testable secara independen, dan bisa di-swap tanpa breaking change di CLI.

Struktur Direktori

```
Trustget/
├── Trustget/
│   ├── __init__.py          # version, public API
│   ├── cli.py               # click commands, UX layer
│   ├── downloader.py        # streaming download + progress
│   ├── verifier.py          # SHA256/SHA512/MD5 + GPG
│   ├── trust.py             # trust score engine
│   ├── github.py            # GitHub Release API integration
│   ├── scanner.py           # URL dir scanning for checksums
│   ├── sandbox.py           # isolated execution environment
│   ├── reporter.py          # JSON/text output formatter
│   └── utils.py             # shared helpers
├── tests/
│   ├── unit/
│   ├── integration/
│   └── fixtures/
├── docs/
│   ├── architecture.md
│   └── trust-scoring.md
├── .github/workflows/
│   └── ci.yml               # test + lint + release
├── README.md
├── pyproject.toml
├── CHANGELOG.md
└── LICENSE
```

Dependency Graph Antar Modul

```
cli.py  (entry point)
  ├──▶  downloader.py  ──▶  utils.py
  ├──▶  verifier.py    ──▶  scanner.py
  ├──▶  github.py      ──▶  verifier.py
  ├──▶  trust.py       ◀──  semua modul di atas
  ├──▶  reporter.py    ◀──  trust.py + verifier.py
  └──▶  sandbox.py     (opsional, Phase 5)
```

3. Feature Roadmap — Detail Implementasi

Setiap phase didesain agar bisa di-ship dan di-demo secara independen. Ini penting untuk momentum project di GitHub.

Phase 1 Smart Downloader [MVP — WAJIB]

Fondasi Trustget. Harus terasa profesional dari hari pertama.

- Streaming download dengan chunk size adaptif (optimal untuk file besar)
- Progress bar real-time via Rich: kecepatan, ETA, ukuran
- Auto-detect filename dari Content-Disposition header atau URL
- Resume support: deteksi partial file, kirim Range header
- Timeout handling + retry dengan exponential backoff (3x default)
- Checksum awal file: simpan metadata ke .Trustget-meta.json

Phase 2 Auto Hash Detection & Verification [Core Value]

Ini yang membedakan sg dari wget. Tanpa konfigurasi apapun.

- Scan direktori URL untuk file: .sha256, .md5, checksums.txt, SHA256SUMS, B2SUMS
- Parse format GNU coreutils checksum (hash filename) dan format alternatif
- Multi-algorithm: SHA256, SHA512, SHA1, MD5 — auto-detect dari konten file
- Support inline checksum di URL sama (e.g. file.tar.gz.sha256)
- Batch verify: satu checksums.txt untuk banyak file sekaligus
- Output human-readable + machine-readable (--json flag)

Phase 3 GitHub Smart Mode [Differentiator]

Feature killer: zero-config security untuk GitHub Releases.

- Auto-detect URL GitHub Release dari pattern github.com/*/releases/download/*
- GitHub API: ambil release metadata, tag, publisher, tanggal publish
- Scan release assets untuk checksum file yang matching
- Verifikasi apakah release dibuat oleh maintainer repo (bukan fork)
- Deteksi pre-release, draft release, dan release yang terlalu lama (> 2 tahun)
- Support GitHub token via env var Trustget_GITHUB_TOKEN untuk rate limit
- Tampilkan release notes singkat (3 baris pertama) di output

Phase 4 Trust Score Engine [Unique Selling Point]

Sistem skoring transparan yang bisa di-audit. Bukan black box.

- Skor 0-100 dengan breakdown per faktor (bukan cuma angka akhir)
- Risk level: CRITICAL (<40), HIGH (40-59), MEDIUM (60-79), LOW (80-100)
- Faktor positif: HTTPS, checksum verified, GPG signed, known domain, recent release, maintainer verified
- Faktor negatif: HTTP only, no checksum, unknown domain, new repo (<6 bulan), redirect chain
- Domain reputation database (lokal, updateable): github.com, kernel.org, apache.org, dst
- Export trust report ke JSON untuk integrasi CI/CD dan SIEM
- Custom policy file: organisasi bisa set minimum trust score untuk auto-approve

Phase 5 — Safe Execute (Sandbox) [Power User]

Eksekusi file dengan risiko diminimalisir. Bukan VM, tapi lebih dari sekadar run biasa.

- Temporary working directory yang di-cleanup otomatis setelah exit
- Subprocess isolation: TMPDIR override, PATH terbatas
- Linux namespaces via unshare (jika tersedia) untuk filesystem isolation
- Log semua syscall ke file audit (via strace jika tersedia)
- Prompt konfirmasi dengan trust score sebelum eksekusi
- Support AppImage, shell script, binary executable
- Fallback graceful jika unshare tidak tersedia (non-root env)

4. CLI Design & UX

CLI Trustget mengikuti Unix philosophy: lakukan satu hal dengan baik, composable, dan bisa dipakai dalam pipeline.

Command Reference Lengkap

Command	Fungsi	Flag Penting
sg <url>	Download + auto-verify + trust score	--output, --no-verify, --json, --quiet
sg verify <file>	Verify file lokal terhadap checksum	--checksum <hash>, --algo sha256
sg trust <url>	Analisis trust tanpa download	--json, --min-score <n>
sg run <file>	Execute di sandbox	--no-sandbox, --audit-log
sg info <url>	Tampilkan metadata file/release	--json
sg config	Manage konfigurasi Trustget	--set, --get, --reset

Flag Global

Flag	Shorthand	Default	Keterangan
--json	-j	false	Output mesin-readable JSON
--quiet	-q	false	Hanya tampilkan error
--verbose	-v	false	Debug info lengkap
--no-color		false	Untuk piping ke file/log
--timeout	-t	30s	HTTP request timeout
--retry	-r	3	Jumlah retry otomatis
--config	-c	~/.Trustget.toml	Path file konfigurasi custom

Contoh Output Terminal — Full Flow

5. Trust Score Engine — Spesifikasi Detail

Trust Score adalah fitur pembeda Trustget. Berikut spesifikasi lengkap algoritma scoring dan cara mengekstensinya.

Scoring Matrix Lengkap

Faktor	Bobot	Sumber Data	Kondisi
HTTPS connection	+20	URL scheme	https:// vs http://
Checksum tersedia	+10	URL dir scan / GitHub API	Ada file checksum
Checksum verified	+25	Verifier module	Hash match 100%
GPG signature	+25	File .asc / .sig	Signature valid dari keyserver
Known platform	+10	Domain allowlist	github.com, kernel.org, dst
Maintainer verified	+20	GitHub API	Release oleh repo owner/org
Repo age > 1 tahun	+7	GitHub API	created_at < now - 365d
Recent release	+10	GitHub API	published_at < 30 hari lalu
HTTP redirect	-10	requests history	Setiap redirect ke domain lain
Unknown domain	-20	Domain allowlist	Domain tidak ada di allowlist
No checksum	-15	Verifier module	Tidak ada checksum ditemukan
New repo < 3 bulan	-20	GitHub API	Repo baru, belum track record
Pre-release / draft	-10	GitHub API	Flag is_prerelease atau draft

Risk Level Classification

Skor	Level	Rekomendasi Tindakan
< 40	CRITICAL	Abort download, tampilkan warning mencolok, require --force flag untuk lanjut
40–59	HIGH	Prompt konfirmasi interaktif, jelaskan faktor negatif spesifik
60–79	MEDIUM	Lanjut dengan warning, tampilkan faktor yang bisa diperbaiki
80–100	LOW	Lanjut tanpa interupsi, log sukses ke ~/.Trustget/history.log

6. Tech Stack & Dependencies

Runtime Dependencies

Library	Versi Min	Kegunaan	Alasan Dipilih
requests	2.28	HTTP download + GitHub API	Stable, familiar, sync cukup untuk MVP
rich	13.0	Progress bar, colored output, tables	Best-in-class terminal UI untuk Python
click	8.0	CLI framework	Lebih ergonomis dari argparse, composable
python-gnupg	0.5	GPG signature verification	Wrapper tipis untuk gpg binary
tomllib	builtin 3.11+	Parse config file .toml	Builtin, zero dependency
platformdirs	3.0	Path config/cache cross-platform	XDG compliant di Linux

Dev Dependencies

Tool	Kegunaan
pytest + pytest-cov	Unit dan integration test, coverage report
pytest-httpserver	Mock HTTP server untuk test download + checksum
ruff	Linter + formatter (pengganti flake8 + black, jauh lebih cepat)
mypy	Static type checking (gunakan type hints dari awal)
pre-commit	Auto-lint sebelum commit, jaga kualitas kode
twine + build	Publish ke PyPI

pyproject.toml — Struktur Kunci

```
[project]
name = "Trustget"
version = "0.1.0"
description = "Secure file downloader with automatic verification"
requires-python = ">=3.11"
dependencies = ["requests>=2.28", "rich>=13.0", "click>=8.0", "python-gnupg>=0.5"]

[project.scripts]
sg = "Trustget.cli:main"

[project.optional-dependencies]
dev = ["pytest", "pytest-cov", "pytest-httpserver", "ruff", "mypy", "pre-commit"]

[tool.ruff]
line-length = 100
select = ["E", "F", "I", "N", "W"]
```

7. Testing Strategy

Project yang dianggap serius butuh test coverage yang serius. Target: 80%+ coverage sebelum ship ke PyPI.

Level Test	Tool	Coverage Target	Yang Di-test
Unit	pytest	90%+	Setiap fungsi verifier.py, trust.py, scanner.py secara isolasi
Integration	pytest + httpserver	80%+	Full flow download → verify → score dengan mock server
E2E	pytest + real URLs	Manual	Test terhadap GitHub release sungguhan (jalankan pre-release)
Security	bandit	100% pass	Scan kode untuk common security issues (subprocess injection, dll)

Test Case Wajib untuk Phase 1-2

- Download file valid: cek file tersimpan, ukuran benar
- Download URL tidak valid: harus raise dengan pesan jelas
- Resume download: simulasi partial file, cek bytes dilanjutkan
- Checksum match: SHA256 benar → return True
- Checksum mismatch: SHA256 salah → raise VerificationError
- Checksum file tidak ditemukan: return None, bukan crash
- Trust score GitHub valid: score harus > 60
- Trust score HTTP plain: score harus < 40

8. GitHub Launch Strategy

Commit Message Convention (Conventional Commits)

```
# Format: <type>(<scope>): <description>

feat(downloader): add resume support with Range header
feat(verifier): auto-detect SHA256/SHA512 from checksums.txt
feat(github): parse release assets for checksum files
feat(trust): implement domain allowlist scoring
fix(scanner): handle checksums.txt with Windows line endings
test(verifier): add edge cases for malformed hash files
docs: add architecture decision records (ADR)
chore: bump requests to 2.31.0 (security patch)
```

Release Timeline Realistis

Milestone	Target Waktu	Deliverable	GitHub Action
v0.1.0	Minggu 1-2	Phase 1 selesai, test 90%	Tag release, publish ke TestPyPI
v0.2.0	Minggu 3-4	Phase 2 selesai, docs update	Tag release, announce di Reddit r/selfhosted

Milestone	Target Waktu	Deliverable	GitHub Action
v0.3.0	Minggu 5-6	Phase 3 selesai, README lengkap	Tag release, post di HN Show HN
v0.4.0	Bulan 2	Phase 4 selesai, blog post	Publish ke PyPI stable, social media
v0.5.0	Bulan 3	Phase 5 beta, community feedback	Cari kontributor, open issues

README Harus Punya

- Demo GIF/Asciicast di bagian atas (pakai asciinema, gratis)
- Badges: build passing, coverage, PyPI version, license
- Satu-liner install: pip install Trustget
- Quick start: 3 contoh langsung copy-paste
- Comparison table: Trustget vs wget vs curl
- Security model explanation: bagaimana trust score dihitung
- Contributing guide: setup dev env, run tests

9. Future Expansion — Roadmap Jangka Panjang

Feature	Kompleksitas	Impact	Milestone
Cosign / Sigstore support	Tinggi	Tinggi	v1.0 - DevSecOps standard
SBOM (CycloneDX/SPDX) verification	Tinggi	Tinggi	v1.1 - Supply chain security
VirusTotal API integration	Sedang	Tinggi	v0.6 - Malware scan
Docker image digest verify	Sedang	Tinggi	v0.7 - Container security
Homebrew tap + AUR package	Rendah	Tinggi	v0.5 - Distribusi lebih luas
VS Code extension	Tinggi	Sedang	v1.2 - Developer experience
Trustget.json policy file	Rendah	Sedang	v0.5 - Enterprise policy
Async download engine	Sedang	Sedang	v0.8 - Performance
Notarization check (macOS)	Sedang	Rendah	v0.9 - Multi-platform
Web dashboard (trust audit log)	Tinggi	Rendah	v2.0 - Enterprise tier

10. Success Metrics

Ukur progress secara objektif. Ini juga berguna kalau mau pitch project ke employer atau komunitas.

Metric	Target 3 Bulan	Target 6 Bulan	Cara Ukur

Metric	Target 3 Bulan	Target 6 Bulan	Cara Ukur
GitHub Stars	50+	200+	GitHub API / star-history.com
PyPI Downloads/bulan	100+	1000+	pypistats.org
Test Coverage	80%+	85%+	pytest-cov + Codecov badge
Open Issues Response Time	< 3 hari	< 24 jam	GitHub Insights
Supported Platforms	3 (Ubuntu, Arch, Fedora)	5+	CI matrix test
Contributors	Solo	2-3 external	GitHub Contributors graph

Trustget Project Blueprint
"Make secure the default, not the exception."