Open-Source-Software BSI-Grundschutzfähig machen

Stefan Schumacher

Chemnitzer Linux-Tage 2025

$Id: CLT2025-FLOSS-BSI.tex,v 1.7 2025/03/20 17:50:46 stefan Exp $

Siehe auch

- Bitkom Forum Open Source 2016: Sicherheit und Vertrauen
- CLT2017: Sicherheit und Vertrauen - Erfahrungen als Open Source Hacker

Schumacher, S. (2011). Sicherheit messen: Eine Operationalisierung als latentes soziales Konstrukt. In S. Adorf, J.-F. Schaffeld & D. Schössler (Hrsg.), *Die sicherheitspolitische Streitkultur in der Bundesrepublik Deutschland: Beiträge zum 1. akademischen Nachwuchsförderpreis Goldene Eule des Bundesverbandes Sicherheitspolitik an Hochschulen (BSH)* (S. 1–38). Magdeburg: Meine Verlag. Verfügbar unter https://portal.dnb.de/opac.htm?method=simpleSearch&cqlMode=true&query=idn%3D1016863993

Inhaltsverzeichnis

Hintergrund

- Dataport Sicherheitsarchitekt, Schwerpunkt Open Source und Kryptographie
- https://www.dphoenixsuite.de/
- Open Source basierter Online-Arbeitsplatz
- Grundlage für den Souveränen Arbeitsplatz / OpenDesk des ZenDiS
- U.a. mit Dateiablage, Video-Konferenzen, Instant Messaging, Online-Office
- Z.B. für Schulen, Gesundheitswesen, Robert-Koch-Institut, weitere Bedarfsträger
- BSI Grundschutz Hoch, Evaluation VS NfD
- Entwurf der Sicherheitsarchitektur für den Souveränen Arbeitsplatz OpenDesk (SouvAP Steckbrief D-03-DP-102)

BSI-Gesetz

- Bundesoberbehörde: Zuständig für die IT-Sicherheit auf Bundesebene
- Nachgeordnete Behörde des BMI
- §3: Förderung der Sicherheit in der Informationstechnik
- §8: Mindeststandards für die Sicherheit der Informationstechnik des Bundes
- §8c Besondere Anforderungen an Anbieter digitaler Dienste
- §9a: nationale Zertifizierungsstelle der Bundesverwaltung für IT-Sicherheit
- §9b: Untersagung des Einsatzes kritischer Komponenten
- §10: Kritische Infrastruktur: Sicherheit unter Berücksichtigung des Standes der Technik

BSI Grundschutz

- Vorgehensweise
- Aufbau eines Informationssicherheitsmanagementsystem
- technische, infrastrukturelle, organisatorische und personelle Sicherheitsanforderungen
- BSI Standards, IT-Grundschutzkompendium, Technische Richtlinien
- Standards: Empfehlungen zu Methoden, Prozessen und Verfahren sowie Vorgehensweisen und Maßnahmen
- Grundschutzbausteine: Prozesse und Systeme (z.B. APP.3.4 Samba; APP.4.4 Kubernetes)
- Technische Richtlinie: Empfehlungen, Verbindlichkeiten durch Vorgaben des Bedarfsträgers
- ISO-27001-Zertifizierung auf Basis von IT-Grundschutz möglich

Inhaltsverzeichnis

Modalverben

- Jedes SOLLEN ist ein MÜSSEN wenn man KANN ;-)
- Notwendigkeit einer Anforderung
- MUSS, DARF NUR: muss unbedingt erfüllt werden
- SOLLTE: normalerweise erfüllt werden, bei stichhaltigen Gründen aber vernachlässigbar
- NICHT: Negation

APP.3.2 Webserver

APP.3.2.A11 Verschlüsselung über TLS (B)

Der Webserver **MUSS** für alle Verbindungen durch nicht vertrauenswürdige Netze eine sichere Verschlüsselung über TLS anbieten (HTTPS). [...]

BSI TR-02102-2

Kryptographische Verfahren: Verwendung von Transport Layer Security (TLS)

Empfohlene Cipher-Suiten für TLS 1.3

- TLS_AES_128_GCM_SHA256 0x13,0x01 [RFC 8446] 2030+
- TLS_AES_256_GCM_SHA384 0x13,0x02 [RFC 8446] 2030+
- TLS_AES_128_CCM_SHA256 0x13,0x04 [RFC 8446] 2030+
- TLS_CHACHA20_POLY1305_SHA256?
- TLS1.3 und (fast) alles ist OK
- TLS1.2 erfordert ausgiebige Konfigurationsarbeit

BSI TR-02102-2

Kryptographische Verfahren: Verwendung von Transport Layer Security (TLS)

Empfohlene Cipher-Suiten für TLS 1.3

- TLS_AES_128_GCM_SHA256 0x13,0x01 [RFC 8446] 2030+
- TLS_AES_256_GCM_SHA384 0x13,0x02 [RFC 8446] 2030+
- TLS_AES_128_CCM_SHA256 0x13,0x04 [RFC 8446] 2030+
- TLS_CHACHA20_POLY1305_SHA256?
- TLS1.3 und (fast) alles ist OK
- TLS1.2 erfordert ausgiebige Konfigurationsarbeit

APP6: Allgemeine Software

BSI APP.6.A3 Sichere Beschaffung von Software

Wenn Software beschafft wird, MUSS auf Basis des Anforderungskatalog eine geeignete Software ausgewählt werden. Die ausgewählte Software MUSS aus vertrauenswürdigen Quellen beschafft werden. Die vertrauenswürdige Quelle SOLLTE eine Möglichkeit bereitstellen, die Software auf Integrität zu überprüfen. Darüber hinaus SOLLTE die Software mit einem geeigneten Wartungsvertrag oder einer vergleichbaren Zusage des Herstellers oder Software-Anbieters beschafft werden. Diese Verträge oder Zusagen SOLLTEN insbesondere garantieren, dass auftretende Sicherheitslücken und Schwachstellen der Software während des gesamten Nutzungszeitraums zeitnah behoben werden.

Hintergrund

Sichere Quelle in der Praxis

- Wartungsvertrag?!
- Enterprise Distributionen! LTE oft ohne aktuelle Applikationen
- Software hoffnungslos veraltet
- Vorallem in einem DevSecOps-Prozess

Hintergrund

Lösung

- Betrieb will Sofware selber bauen
  - Tarball von irgendeiner Webseite
  - Quellcode aus irgendeinem Repository
  - Container aus irgendeinem Container Hub
- unsichere Quellen $\rightsquigarrow$ Prozess
- Richtlinie samt Checkliste zur Bewertung von Christian Tramnitz et. al. erstellt
- als benutzerdefinierter Baustein beim BSI eingereicht

Inhaltsverzeichnis

FLOSS-Bewertungsprozess

1. FLOSS kommt mit SiKo-konformem Wartungsvertrag $\rightsquigarrow$ kann eingesetzt werden
2. FLOSS kommt ohne SiKo-konformem Wartungsvertrag
   - selbst entwickelte Checkliste, um die BSI-Anforderungen abzudecken
   - Mitigationsmaßnahmen
3. Checkliste nicht erfüllt: Möglichkeit der Entwicklungsübernahme

Checkliste

- inspiriert durch FLOSS Score Card Prozess
- teil-automatisiert prüfbar durch https://github.com/ossf/scorecard
- wenn Github.com genutzt wird
- sonst Handarbeit (z.B. Nginx in HG)

Checkliste

Ist die Lizenz FLOSS? (Lex Redis)

- White-/Blacklisting von Lizenzen
- GPL, LGPL, MIT, ISC,
- BSD: 0/1/2/3/4 Clause, Patente ...
- https://opencode.de/de/wissen/rahmenbedingungen/standardisierte-open-source-lizenzen
- https://www.bitkom.org/sites/main/files/2022-06/220624-Bitkom-Leitfaden-Open%20Source-3.0_0.pdf
- Problemfeld: Protokolle und deren Lizenzierbarkeit

Checkliste

- Ist die Software wichtig, komplex und/oder einfach ersetzbar? (Migrierbarkeit) (Lex Imaginary)
- Gibt es Aktivität? (Bus-Faktor)
- Wie wird mit Sicherheitsproblemen umgegangen? (Projekthistorie)
- Unterstützung durch etablierte NPO (Lex ValKey)
- Integrierbarkeit in interne Prozesse (z.B. Versionsverwaltungssystem)

Offene Probleme

- Abhängigkeiten
- Abhängigkeiten von Abhängigkeiten

Offene Probleme

- Abhängigkeiten
- Abhängigkeiten von Abhängigkeiten
- Abhängigkeiten von Abhängigkeiten von Abhängigkeiten

Offene Probleme

- Abhängigkeiten
- Abhängigkeiten von Abhängigkeiten
- Abhängigkeiten von Abhängigkeiten von Abhängigkeiten
- Toolchain / Buildchain / Software Supply Chain Security
- *Reflections on Trusting Trust* $\rightsquigarrow$ Compiler?
- SBOM nach TR-03183: Cyber Resilience Requirements for Manufacturers and Products

Offene Probleme

- Abhängigkeiten
- Abhängigkeiten von Abhängigkeiten
- Abhängigkeiten von Abhängigkeiten von Abhängigkeiten
- Toolchain / Buildchain / Software Supply Chain Security
- *Reflections on Trusting Trust* $\rightsquigarrow$ Compiler?
- SBOM nach TR-03183: Cyber Resilience Requirements for Manufacturers and Products

Inhaltsverzeichnis

BSI Anforderungen

Die meisten relevanten Anforderungen sind im Betrieb!

- Ist TLS aktiviert?
- Sind nur die freigegebenen Cipher-Suiten aktiviert?
- Terminiert TLS im ALG?
- Gibt es die geforderte P-A-P-Struktur?
- Außerhalb des Einflussbereiches des Projektes

BSI-Anforderungen

Kryptographie

- Technische Richtlinien beachten
- 02102: Februar 2025 aktualisiert
- Kryptoagilität: Austauschbarkeit der Cipher-Suiten (Bibliotheken) beachten
- Never roll your own crypto!
- TLS1.3 und (fast) alles ist schick
- Quantensichere asymmetrische Key Encapsulation Mechanism
  - FrodoKEM Classic
  - McEliece Schlüsseleinigung
  - ML-KEM

TLS in Projekten des Bundes

- Testspezifikation TR-03116-TS für IT-Projekte des Bundes.
- TLS-Testtools TaSK
- prüft die Konformität der TLS-Implementierung
- https://github.com/BSI-Bund/TaSK
- ECC $\rightsquigarrow$ Brainpool!
- https://github.com/gematik/ecc-brainpool-how-to
- BouncyCastle; OpenSSL implementiert (und diskutiert) noch

Tips für FLOSS-Projekte

- verschlüsselte Meldewege für Sicherheitslücken einrichten
- Prozess zur Behebung von Sicherheitslücken dokumentieren
- (signierte) Security Advisories anbieten
- SBOMs mitliefern
- Release Notes!
  - insbesondere wenn Sicherheitslücken geschlossen werden
  - CVE referenzieren, wenn vorhanden
  - OpenSSF Scorecard automatisiert einbinden (Github-Action)

Inhaltsverzeichnis

OpenSSF Scorecard

- https://scorecard.dev/
- Scorecard: Konzept zur Messung und Bewertung, *immer* eine Vereinfachung
- Automatisiertes Testen auf Sicherheitsprobleme
  - Malicious maintainers
  - Build system compromises
  - Malicious packages
  - Source code compromises

OSSF Scorecard

- OpenSSL-Scorecard im Shortpaper
- https://scorecard.dev/viewer/?uri=github.com/ossf/scorecard
- https://scorecad.dev/viewer/?uri=gitlab.com/fdroid/fdroidclient
- Via Github-Action in Pipeline aktivierbar
- Derzeit 1 310 498 Projekte bereits erfasst, Ergebnisse per API auswertbar
- in AUR und HomeBrew verfügbar oder Container per Podman laufen lassen
- Github-Access-Token nötig für Github-API

Fragen?

- stefan.schumacher2 <at> dataport.de
- https://mastodon.social/@0xKaishakunin
- @schumaste:matrix.org