LU03g - Sicherheitsanforderungen

Merke! Eine Applikation kann nur so sicher sein, wie das Umfeld, in dem sie betrieben wird. Dies gilt für die Infrastruktur genauso, wie für die Architektur und die Implementierung.

1. Einleitung

Die Anforderungen an die Sicherheit einer Applikation (Programm) sollen bereits zu Beginn der Entwicklung ermittelt und abgestimmt werden. Eine nachträgliche Implementierung von Sicherheitsmassnahmen ist bedeutend teurer und bietet im Allgemeinen weniger Schutz als Sicherheit, die von Beginn an in den Systementwicklungsprozess oder in den Auswahlprozess für ein Produkt integriert wurde.

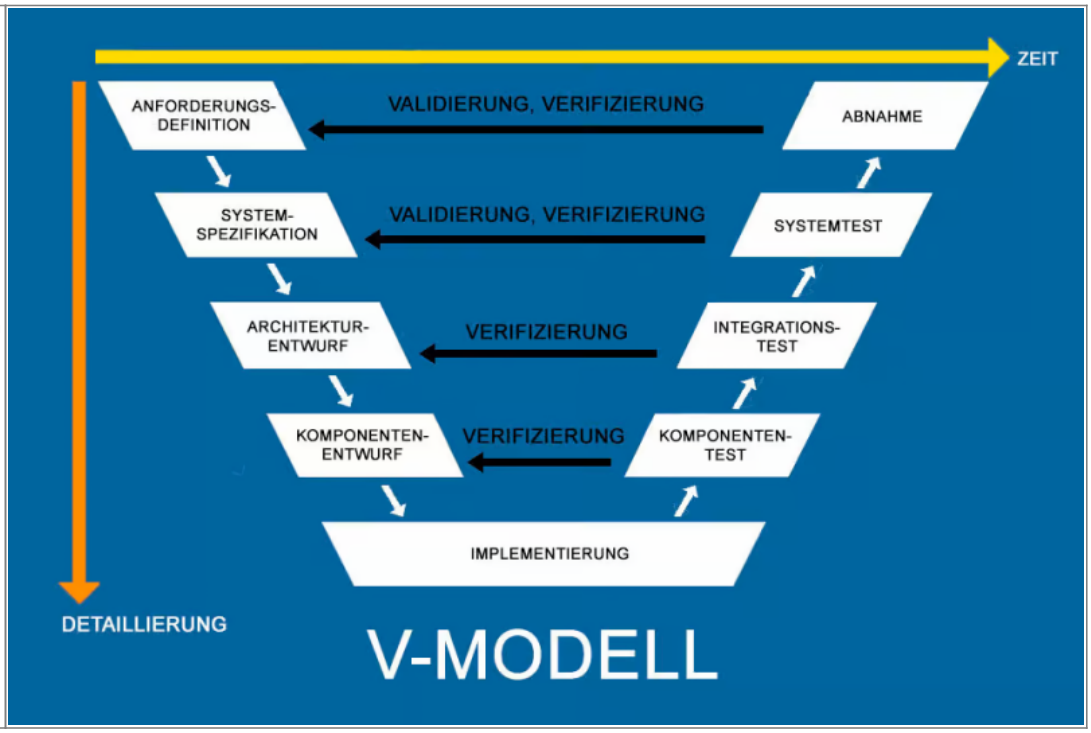

Abbildung: Sicherheitsanforderungen können nur greifen, wenn diese im gesamten SW-Entwicklungszyklus berücksichtigt werden

Sicherheit sollte daher integrierter Bestandteil des gesamten Lebenszyklus einer Applikation (SDLC) bzw. Produktes sein. Eine Applikation (Programm) besitzt ein Umfeld, in welchem sie betrieben wird. Ein sich ereignender Systemprozess (Geschäftsprozess) stösst eine Applikation zur Bearbeitung an (Anfrage). Diese erfüllt die angeforderte Dienstleistung und gibt ein erarbeitetes und erwartetes Ergebnis (Resultat) zurück. Dabei greifen die Funktionen einer Applikation über ein logisches und physisches Datenmodell auf die Daten zu. In diesem Verfahren gibt es verschiedene Bereiche, die kontrolliert ablaufen müssen. Nebst der Qualität der Applikationsfunktionen (korrekte Abläufe, Berechnungen, etc.) sind die Sicherheitsbelangen auf allen Ebenen.

2. Grundprinzip der sicheren SW-Entwicklung

Die folgenden Prinzipien sollten in jedem Projekt von Anfang an berücksichtigt werden:

- **Security by Design:** Sicherheit ist kein nachträgliches Feature, sondern muss in Architektur und Design integriert sein.
- **Least Privilege:** Jeder Code-Abschnitt, jede Komponente, jeder Benutzer bekommt nur so viel Rechte wie absolut nötig.
- **Fail Secure:** Im Fehlerfall muss das System in einen sicheren Zustand übergehen.
- **Defense in Depth:** Mehrstufige Sicherheitsmaßnahmen bieten auch dann Schutz, wenn eine Ebene versagt.
- **Minimierung der Angriffsfläche:** Weniger exponierte Funktionen bedeuten weniger potenzielle Angriffsvektoren.

3. Sicherheitsanforderungen

3.1 Authentifizierung und Autorisierung

- Unterstützung starker Authentifizierungsmechanismen (z. B. OAuth2, MFA)
- Rollenkonzepte zur feingranularen Rechtevergabe
- Session-Management mit begrenzter Lebensdauer

3.2 Daten- und Kommunikation

- Verschlüsselung sensibler Daten (ruhend und in Übertragung)
- Einsatz von TLS 1.2 oder höher
- Keine sensiblen Daten im Klartext loggen oder speichern

3.3 Eingabeverarbeitung

- Validierung aller Benutzereingaben (Client- & Server-seitig)
- Schutz gegen SQL-Injection, Cross-Site Scripting (XSS), CSRF etc.
- Whitelisting statt Blacklisting

3.4 Fehlerbehandlung und Logging

- Fehlermeldungen dürfen keine internen Informationen preisgeben
- Logging sicherheitsrelevanter Ereignisse mit Integritätsschutz
- Protokollrotation und -archivierung

3.5 Drittanbieter-Software und Bibliotheken

- Nur verifizierte, regelmäßig gepflegte Bibliotheken verwenden
- Schwachstellenprüfung mit Tools wie OWASP Dependency-Check
- Automatisiertes Patch- und Update-Management

4. Entwicklungsprozess-bezogene Anforderungen

4.1 Schulung und Awareness

- Alle Entwickler müssen regelmäßig in Secure Coding geschult werden
- Awareness für aktuelle Bedrohungen (z. B. OWASP Top 10)

4.2 Codeanalyse und Tests

- Statische und dynamische Codeanalyse verpflichtend
- Security Unit-Tests und Penetrationstests
- Regelmäßige Code-Reviews mit Fokus auf Sicherheitsaspekte

4.3 DevSecOps

- Integration von Sicherheit in die CI/CD-Pipeline
- Automatisierte Schwachstellenprüfungen bei jedem Build
- Secrets-Management (z. B. HashiCorp Vault)

5. Dokumentation und Nachvollziehbarkeit

- Sicherheitsanforderungen müssen dokumentiert und versioniert sein
- Architekturentscheidungen im Hinblick auf Sicherheit müssen nachvollziehbar sein
- Änderungsmanagement mit Sicherheitsbewertung

6. Compliance und gesetzliche Anforderungen

- Einhaltung relevanter Standards (z. B. ISO 27001, BSI-Grundschutz, GDPR/DSGVO)
- Umsetzung branchenspezifischer Vorgaben (z. B. KRITIS, PCI-DSS)
- Auditierbarkeit und Nachweisführung über implementierte Sicherheitsmaßnahmen

7. Fazit

Zusammenfassend kann gesagt werden, dass Sicherheitsanforderungen kein lästiger Ballast, sondern ein integraler Bestandteil professioneller Softwareentwicklung, sind. Wer hier spart, zahlt später – sei es mit Datenschutzpannen, Imageschäden oder rechtlichen Konsequenzen.

9. Quellennachweis Bilder

- https://t2informatik.de/wissen-kompakt/v-modell/