

EINFÜHRUNG



Bedrohungsanalysen sind Teil der Designphase im Software Development Lifecycle (SDLC). Im agilen Umfeld werden diese Designphasen zyklisch wiederholt. Die Bedrohungsanalysen helfen dabei, Design-Flaws und Bedrohungen frühzeitig zu erkennen und Gegenmaßnahmen zu finden.

Die im Folgenden beschriebene Bedrohungsanalyse enthält Elemente aus Microsoft STRIDE, DREAD usw., vereinfacht diese aber stark. Ein weiteres Werkzeug sind die beschriebenen Evil-User Stories bzw. Abuse-Stories, die für jeden Feature Request angewandt werden sollten.

VORBEREITUNG

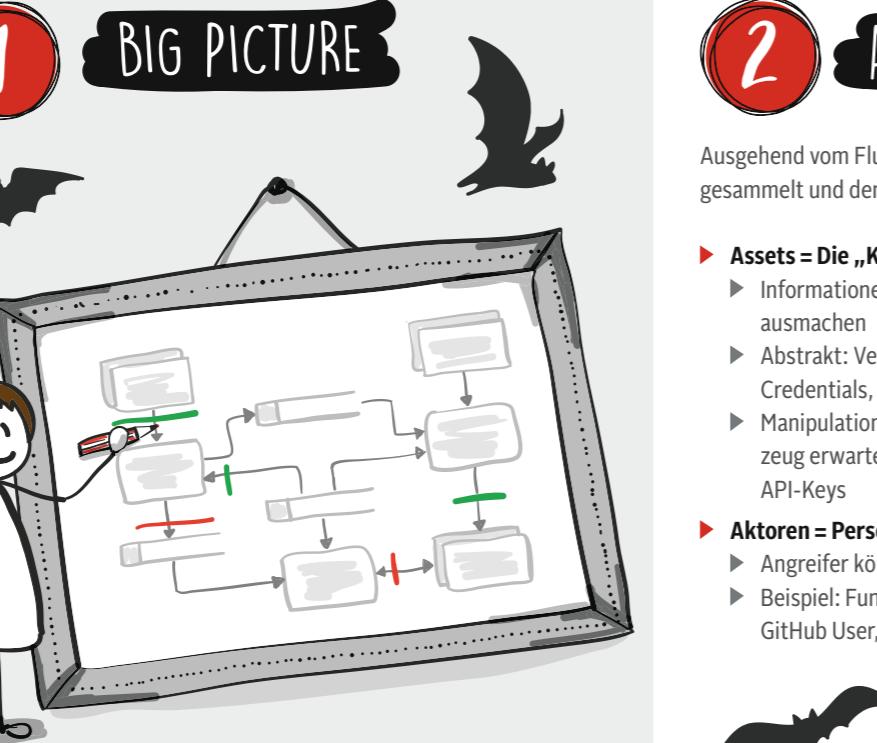
Die Durchführung einer Bedrohungsanalyse ist Teamsache. Gerade initial sollte sie vom ganzen Team durchgeführt werden, da die Bedrohungen häufig nicht nur auf technischer Ebene lauern. Bei „größeren“ Änderungen oder spätestens nach sechs Monaten sollte die Analyse wiederholt werden, um eine stetige Verbesserung zu forcieren. Der Raum für die Durchführung sollte genug Platz für Diagramm-Zeichnungen und Post-Its-Tapeten bieten. Genauso verhält es sich mit der Zeit: Initial dauert eine Bedrohungsanalyse häufig mind. 3 Stunden. Das Bedrohungsanalyse-Poster kann euch helfen, um selbst Bedrohungsanalysen in kleiner Runde durchzuführen. Für Unterstützung bei der Moderation kommt gerne auf Q-SEC (q-sec@otto.de) zu.

DURCHFÜHRUNG

Bei der Durchführung ist es ratsam, nicht bei jedem Schritt in eine tiefe technische Diskussion zu verfallen - die findet am Ende auf Basis von Fakten statt. Des Weiteren sollte der Scope der Betrachtung festgelegt werden: Wollen wir das gesamte System oder einen spezifischen Service analysieren? Build-System und Umfeld sollten jeweils als einzelne Services analysiert werden.



Das Flussdiagramm von Beginn an im Team entwickeln. Am besten mit Post-Its, um später alles übersichtlich anordnen zu können.

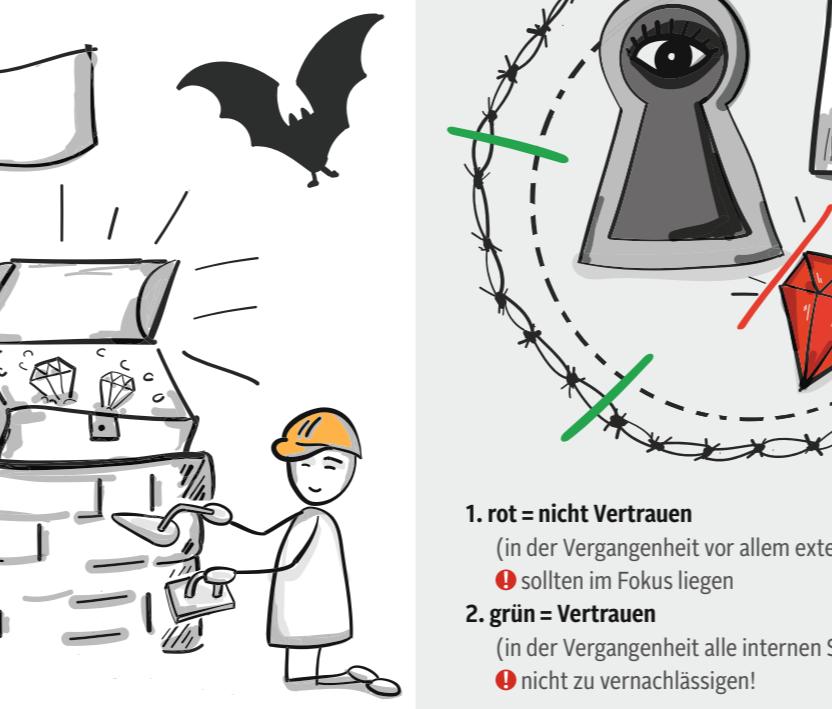


Das Flussdiagramm von Beginn an im Team entwickeln. Am besten mit Post-Its, um später alles übersichtlich anordnen zu können.

2 ASSETS UND AKTOREN

Ausgehend vom Flussdiagramm werden alle Assets und Aktoren gesammelt und den Elementen im Flussdiagramm zugeordnet.

- ▶ **Assets = Die „Kronjuwelen“ des Unternehmens**
 - Informationen und Eigenschaften, die das System ausmachen
 - Abstrakt: Verfügbarkeit, Integrität, Reputation, Credentials,
 - Manipulation von Kontoständen, kein POrn wo Kinderspielzeug erwartet wird, Erreichbarkeit und Antwortzeiten, API-Keys
- ▶ **Aktoren = Personas = Rollen**
 - Angreifer können nur in vorgesehene Rollen schlüpfen
 - Beispiel: Funktions-User, AWS Admin, Team-Mitglied, GitHub User, jeder Mitarbeiter mit Firmen-Account



3 VERTRAUENSGRENZEN

Nachdem wir das System detailliert aufbereitet haben, lassen wir unserer „bösen Kreativität“ freien Lauf.

Jetzt steht der Perspektivwechsel zum Angreifer an: Stellt euch vor, dass euch jemand für einen erfolgreichen Angriff eine hohe Summe Geld bietet! Folgend einige Impulse:



Threat	Desired property	Beschreibung
Spoofing (Vortäuschung)	Authenticity	Phishing, IP Pakete fälschen, ► Tchibo statt OTTO.DE wird angezeigt
Tampering (Manipulation)	Integrity	Log File Änderung, Alarne verschleiern ► Daten werden hinterlegt, um die Reputation zu schädigen ► es werden nur Produkte eines Händlers angezeigt
Repudiation (Nachweisbarkeit)	Nonrepudiability	Keine Zuordnung, wer was getan hat - Funktionsuser, Mehrfachvergabe von API Keys ► Zugang bekommen, ohne dass es auf dich zurückfällt
Information disclosure (Ungewollte Informations-Offenlegung)	Confidentiality	Sensible Daten bekannt geben ► Mitarbeiterdaten auf Public S3 ► Benutzerkonten werden an ihn übermittelt
Denial of Service (Verfügbarkeit)	Availability	Service mit unnötigen Anfragen bis zum Ausfall an Kapazitätsgrenzen bringen ► die Antwortzeiten werden komplett in den Keller gedrückt
Elevation of Privilege (Berechtigung erweitern)	Authorization	Mit vorhandenen Zugriffsrechten weitere Rechte erlangen, um z. B. das Build System zu kompromittieren

- 1. rot = nicht Vertrauen**
(in der Vergangenheit vor allem externe Zugriffe und Services)
! sollten im Fokus liegen
- 2. grün = Vertrauen**
(in der Vergangenheit alle internen Services)
! nicht zu vernachlässigen!

5 BEDROHUNGEN PRIORISIEREN

Die Bedrohungsszenarien werden nacheinander vorgestellt, sodass sie jeder versteht und anschließend priorisiert. Ähnliche Szenarien werden vor der Priorisierung zusammengefasst.

Für die Priorisierung werden sechs Risikoaspekte getrennt bewertet. Die Bewertung reicht von 1 (sehr niedrig) bis 5 (sehr hoch).

- ▶ **1-Komplexität, Know-How**
 - 1. niedrige Komplexität = 5
 - 2. hohe Komplexität = 1
- ▶ **2-Erkennbarkeit und Reaktionsmöglichkeit**
 - 1. schnell zu erkennen und schnell zu reagieren = 1
 - 2. schnell erkennen und langsam reagieren = 3
 - 3. nicht erkennen und viel zu spät reagieren= 5
- ▶ **3-Anzahl Aktores (in Relation zum System sehen)**
 - 1. wenige Aktoren = 1
 - 2. viele Aktoren = 5
- ▶ **4-Schadenshöhe**
 - Wie hoch ist der zu erwartende Schaden bei Eintritt
 - 4% des Umsatz bei Verlust von Personenbezogenen-Daten
- ▶ **5-Wiederholbarkeit**
 - wie häufig kann das Vorgehen wiederholt werden
 - lässt die Fachlichkeit eine Mitigation zu oder nicht
- ▶ **6-Anzahl Betroffener Benutzer**
 - je mehr Benutzer betroffen sind desto höher sind die wirtschaftlichen Auswirkungen
- ▶ **7-Motivation des Angreifers (OPTIONAL)**
 - wie hoch ist die Motivation eines Angreifers ausgerechnet diese Bedrohung auszunutzen = welchen Wert hat Sie für ihn

Der Durchschnitt zu jeder Bedrohung (Gruppe) gibt nun die Reihenfolge der Priorisierung vor. Bei Bedarf können die einzelnen Dimensionen gewichtet werden. Bitte nicht zu wissenschaftlich vorgehen und vertraut eurem Bauch. Sollte es zu stark abweichenden Einschätzungen im Team geben, kann darüber, ähnlich wie im Planning-Poker, diskutieren.

