# Security-Goal-Modelling – Vorgehensweise

## Voraussetzung: Lastenheftartige Beschreibung eines Software-Projekts

Eine Beschreibung des Zwecks, des Einsatzszenarios (einschließlich Betriebsumgebung) und der grundlegenden Datenverarbeitungsschritte muss vorliegen. Dies bedeutet, dass eine erste Anforderungsanalyse mit den relevanten Stakeholdern vorausgesetzt wird, als deren Ergebnis eine (vorläufige) Liste der funktionalen Anforderungen erstellt wurde. Diese funktionalen Anforderungen müssen prinzipiell bereits mit dem Rechtsrahmen abgeglichen sein. Die nachfolgend beschriebene Methode ist nicht dazu geeignet, funktionale Anforderungen zu identifizieren, die prinzipiell nicht erfüllbar sind, weil sie rechtlich nicht zulässig oder zwischen verschiedenen Stakeholdern widersprüchlich sind.

Eine (vorläufige) Liste der nicht-funktionalen Anforderungen ist ebenfalls wünschenswert, da diese für die Auswahl konkreter Sicherheitsmechanismen wichtig bzw. hilfreich sein können. Beispiele für relevante nicht-funktionale Anforderungen sind Mechanismen oder Schlüssellängen, die per Regulierung vorgeschrieben sind, erwünschte Eigenschaften von Protokollen wie etwa (Perfect-) Forward-Secrecy oder auch Bedingungen hinsichtlich des Laufzeitverhaltens bzw. von Antwortzeiten eines Dienstes, etc.

## Schritt 0: Leeres Projekt erzeugen

Der Editor fordert den Benutzer auf, einen Namen für das zu modellierende Software-Projekt anzugeben und erzeugt ein leeres Projekt-Objekt, das in den folgenden Schritten mit modelliert wird

## Schritt 1: Identifizieren relevanter Stakeholder

**Stakeholder** werden in einer Liste geführt, in der verschiedene Benutzergruppen des Systems oder von Teilsystemen explizit gekennzeichnet werden sollten (nicht jeder Stakeholder tritt zwingend als Benutzer auf). Für die Modellierung rechtlicher Rahmenbedingungen als nicht-verhandelbare/verpflichtende nicht-funktionale Anforderungen wird das **Recht als separater Stakeholder** eingeführt.

Der Editor fordert den Benutzer auf, eine Liste von Stakeholdern anzulegen:

* Stakeholder sind Personengruppen, die in irgendeiner Beziehung zum geplanten System stehen (Benutzer, Betreiber, Betroffene, Administratoren, etc.)
* Stakeholder müssen nicht zwangsläufig selbst Benutzer des Systems sein
* Das "Recht"/der “Gesetzgeber“ ist in dieser Liste schon vorgegeben
* Stakeholder, zu denen in Schritt 3 keine Soft-Goals zugeordnet werden, können anschließend wieder gelöscht werden

## Schritt 2: Identifizieren der Schutzgüter

Auf Basis der vorausgesetzten Beschreibung des Projekts sind nun die **Schutzgüter** zu identifizieren. Sie sind eine Teilmenge der durch das Zielsystem verarbeiteten Daten.

Der Editor fordert den Benutzer auf, eine Liste von Schutzgütern anzulegen:

* Ein Schutzgut ist eine Klasse/Kategorie von Daten, die das geplante System verarbeiten soll und die für einen der Stakeholder irgendeinen Wert hat

## Schritt 3: Identifizieren von Schutzzielen und Formulierung als Soft-Goals

**Soft-Goals** ordnen den Schutzgütern **Schutzziele** aus der Menge **{Vertraulichkeit, Integrität und Authentizität}** zu. Ein Soft-Goal wird somit dargestellt als Datum bzw. Datenklasse, die einem Schutzziel und einem oder mehreren Stakeholdern zugeordnet ist. Diese Zuordnung zu Stakeholdern wird für die Erstellung eines Vertrauensmodells benötigt (vgl. Schritt 6).

Der Editor fordert den Benutzer auf, den Schutzgütern Schutzziele zuzuordnen:

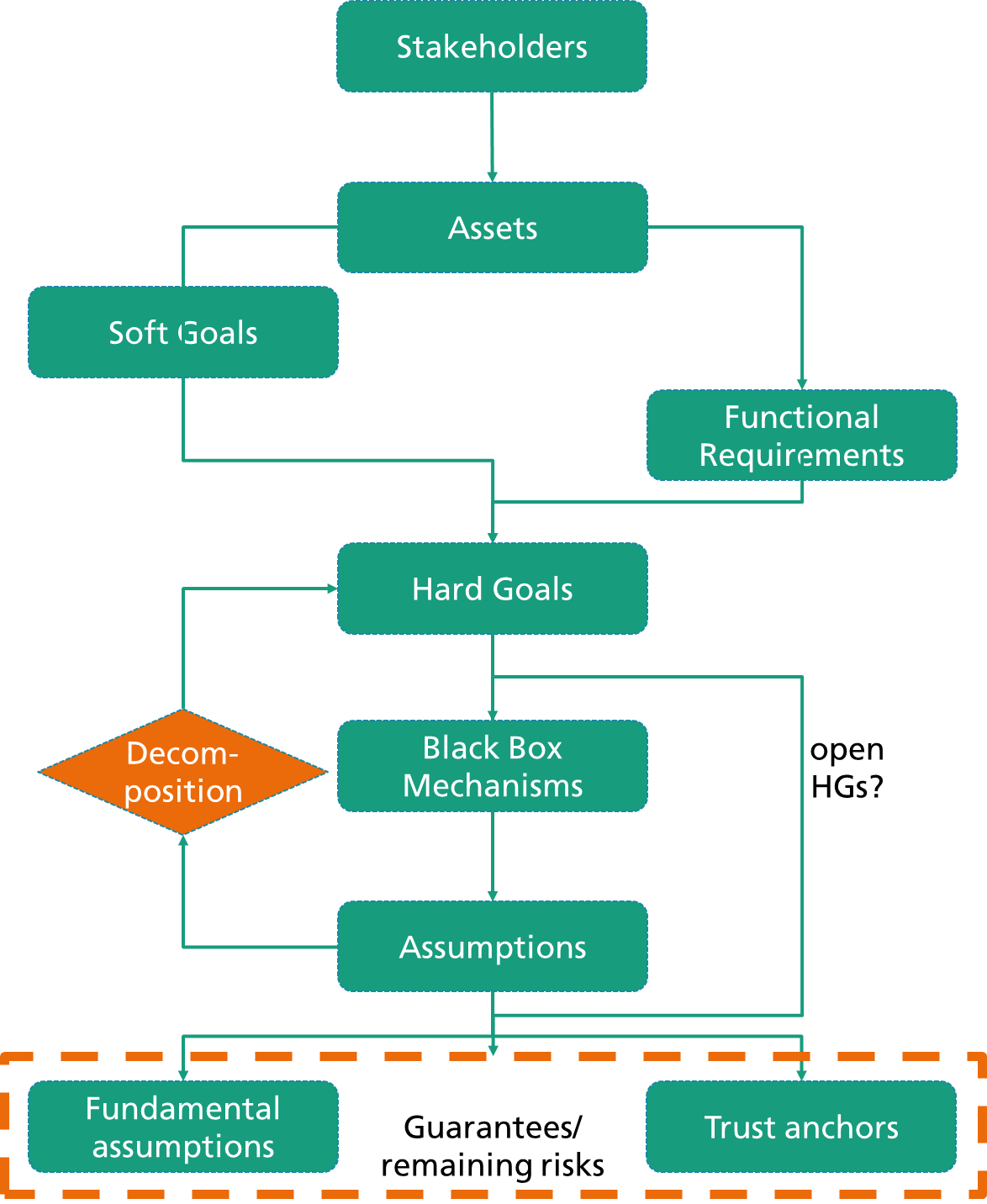
* Für jedes Schutzgut soll der Benutzer selektieren können, welche der Schutzziele "Vertraulichkeit", "Integrität" und "Authentizität" durch das geplante System erreicht werden sollen
* Ein Tupel (Schutzgut, Schutzziel) nennen wir Soft-Goal
* Jedes Soft-Goal muss der Benutzer des Editors einem Stakeholder zuordnen, so dass nachvollzogen werden kann, woher das Soft-Goal kommt

## Schritt 4: Identifizieren sicherheitsrelevanter funktionaler Anforderungen

Aus der vorausgesetzten Dokumentation des Projekts werden **funktionale Anforderungen** identifiziert, welche Verarbeitungen von Schutzgütern darstellen. Idealerweise kann eine funktionale Anforderung bereits zu einer konkreten **Software-Komponente** assoziiert werden.

Der Editor fordert den Benutzer auf, eine Liste funktionaler Anforderungen anzulegen:

* Der Benutzer des Editors sollte hier z.B. ein Pflichtenheft durchgehen und alle funktionalen Anforderungen identifizieren, die eines der Schutzgüter betreffen bzw. verarbeiten
* Optional soll der Benutzer jede funktionale Anforderung einer Komponente des geplanten Systems zuordnen können (z.B. Benutzeroberfläche, Backend, Dienst, …)



## Schritt 5: Identifizieren von Hard-Goals

Soft-Goals und funktionale Anforderungen werden in einer Matrix dargestellt. Durch die Beziehung der funktionalen Anforderungen auf SGs werden **Hard-Goals (HG)** identifiziert. HGs werden anschließend als Anforderung an eine Komponente formuliert, die bei der Implementierung einer bestimmten Funktionalität ein bestimmtes Soft-Goal sicherstellt **(hierfür wird im Anhang ein Vokabular vorgeschlagen**).

Der Editor zeigt dem Benutzer in Schritt 5 eine Matrix mit den Soft-Goals und den funktionalen Anforderungen an. Durch die Zuordnung von SGs zu funktionalen Anforderungen erzeugt der Benutzer des Editors Hard-Goals. Um diese HGs genauer zu spezifizieren, muss der Benutzer die in der Matrix markierten Kombinationen nochmals durchgehen und jeweils eine Komponente angeben (falls noch nicht geschehen), in der die funktionale Anforderung implementiert werden soll, und eine durch das Vokabular (s. Anhang) vorgegebene Formulierung auswählen (orange eingefärbte Blöcke, also z.B. „stellt Schutzziel sicher“).

## Schritt 6: Dekomposition von Hard-Goals und Wahl von Black-Box-Mechanismen

Die Dekomposition von Hard-Goals stellt eine iterative Abarbeitung der HG-Liste da. Hierbei wird für jedes HG ein **Black-Box-Mechanismus (BBM)** gewählt. Ein BBM steht stellvertretend für eine Klasse von Mechanismen (bspw. asymmetrische Verschlüsselung), die die Erfüllung eines Schutzziels garantieren. Die Sicherheit eines BBM beruht üblicherweise auf **Annahmen**. Annahmen sind entweder fundamentaler oder separierender Natur. **Fundamentale Annahmen** sind grundlegende Voraussetzungen für die Sicherheit von BBM (bspw. das Faktorisierungsproblem), die zu dokumentieren sind**. Separierende Annahmen** beziehen sich auf nachgelagerte Sicherheitsprobleme, für welche die Existenz von Lösungen angenommen wird, so dass diese als zusätzliche HGs (**abgeleitete HGs**) behandelt und weiter dekomponiert werden müssen.

Es kann sich herausstellen, dass **ein HG nicht adressiert** werden kann, bspw. wenn dem Benutzer einer Komponente und der Laufzeitumgebung nicht vollständig vertraut wird oder auch weil der Sicherheitsgewinn den durch die Integration geeigneter Mechanismen zu erwartenden Aufwand nicht rechtfertigt. **Bei einem abgeleiteten HG bedeutet dies, dass alle Vorgänger-HGs im betreffenden Ableitungsteilbaum nicht erfüllt sind. Dies muss explizit zusammen mit den Annahmen dokumentiert werden, so dass das Restrisiko vollständig abgebildet wird.**

BBM können weiterhin **zusätzliche Schutzgüter** benötigen (bspw. kryptographisches Schlüsselmaterial, Passwort) oder **zusätzliche funktionale Anforderungen** (bspw. Importieren von Zertifikaten oder sonstiger Authentifizierungsmerkmale). Wurden zusätzliche Schutzgüter oder funktionale Anforderungen eingeführt, so muss eine **weitere Iteration des Prozesses ab Schritt 2** erfolgen.

## Schritt 7: Garantien

Fundamentale Annahmen und nicht behandelte separierende Annahmen bzw. nicht erfüllte HGs (bspw. falls die Sicherheit des Compilers oder des Betriebssystems nicht überprüft wird bzw. werden kann) müssen im Zuge der Dekomposition gesammelt und dokumentiert werden. Aus dieser Dokumentation ergeben sich die **Sicherheitsgarantien** bzw. die **verbleibenden Risiken** des Systementwurfs.

## Anhang

HG-Vokabular (vorläufig):

* Komponente stellt Schutzziel für Schutzgut bei funktionaler Anforderung sicher
* Komponente erlaubt funktionale Anforderung für Schutzgut
  + Für Stakeholder?
* Komponente attestiert Schutzziel für Schutzgut bei funktionaler Anforderung
* Schutzgut fließt nicht aus Komponente
* Schutzgut darf nur benutzt werden von Komponente
  + Von Stakeholder?
* Schutzgut darf nicht benutzt werden von Komponente
  + Von Stakeholder?
* Schutzgut fließt nur in Komponente
* Komponente verbietet funktionakle Anforderung für Schutzgut
  + Für Stakeholder?