Introduction

"Threat modeling is the key to a focused defense. Without threat modeling, you can never stop playing whack-a-mole." — Adam Shostack [14]

Almost all software systems today face a variety of threats, and more are being added constantly as technology changes. These threats can come from outside or within organizations, and their impact has the potential to be devastating. Systems could be prevented from working entirely or sensitive information could be leaked, which would impact consumer trust in the system provider. To prevent threats from taking advantage of system flaws, threat modeling methods can be used to think defensively.

Threat modeling methods are used to create an abstraction of the system; profiles of potential attackers, including their goals and methods; and a catalog of potential threats that may arise. There are many threat modeling methods that have been developed. Not all of them are comprehensive; some focus on the abstraction and encourage granularity while others are more people-centric. Some methods focus specifically on risk or privacy concerns. Threat modeling methods can be combined to create a more robust and well-rounded view of potential threats.

Software systems are increasingly being integrated into physical infrastructures, such as smart cars. These hybrids are often referred to as cyber-physical systems; this term accounts for their multiple components. While innovative, cyber-physical systems are vulnerable to threats that manufacturers of traditional physical infrastructures may not consider. Performing threat modeling on cyber-physical systems with a variety of stakeholders can help catch threats across a wide spectrum of threat types.

To best use threat modeling, it should be performed early in the development cycle. This means that potential issues can be caught early and remedied, preventing a much costlier fix down the line. Thinking about security requirements with threat modeling can lead to proactive architectural decisions that allow for threats to be reduced from the start.

The twelve threat modeling methods discussed in this paper come from a variety of sources and target different parts of the process. No one threat modeling method is recommended over another; the decision of which method(s) to use should be based on the needs of the project and its specific concerns.
Table of Contents

STRIDE and Associated Derivations 1
PASTA 3
LINDDUN 5
CVSS 7
Attack Trees 8
Persona non Grata 9
Security Cards 11
hTMM 12
Quantitative Threat Modeling Method 13
Trike 15
VAST Modeling 16
OCTAVE 17
Conclusion 18
Bibliography 20

List of Figures

Figure 1: Data Flow Diagram with System Boundaries 1
Figure 2: PASTA Stages 3
Figure 3: LINDDUN Methodology Steps [34] 5
Figure 4: LINDDUN Mapping Step [12] 6
Figure 5: CVSS v3.0 Metric Groups [37] 7
Figure 6: Attack Tree Examples [2] 8
Figure 7: Examples of Personae non Grata [15] 9
Figure 8: Security Card Example [15] 11
Figure 9: Component Attack Tree [3] 13
Figure 10: CVSS Scoring for Tampering Attack Tree [3] 13
Figure 11: OCTAVE Phases [36] 17

List of Tables

Table 1: STRIDE Threat Categories 2
Table 2: Security Cards Dimensions 11
Table 3: Threat Modeling Methods Features 18
STRIDE and Associated Derivations

STRIDE is currently the most mature threat modeling method. Invented by Loren Kohnfelder and Praerit Garg in 1999 and adopted by Microsoft in 2002, STRIDE has evolved over time to include new threat-specific tables and the variants STRIDE-per-Element and STRIDE-per-Interaction [14, 20, 40].

STRIDE evaluates the system detail design. (See the example in Figure 1.) The goal of step one is to model the in-place system. By building data flow diagrams (DFDs), you identify system entities, events, and boundaries of the system [26]. Accurate DFDs dictate how successful your STRIDE will be [15]. However, using DFDs as the only input to threat modeling is limiting because it does not provide a means for representing security-related architectural decisions [13].

Figure 1: Data Flow Diagram with System Boundaries

The goal of step two is to find threats. STRIDE uses a general set of known threats based on its name, STRIDE, which stands for Spoofing identity, Tampering with data, Repudiation, Information disclosure, Denial of service, and Elevation of privilege. (See Table 1 for threat type definitions.) This acronym can be used as a mnemonic for discovering threats while navigating the system's model created in phase one [14, 20]. To help in this step, some sources offer checklists and tables [14, 28] that assist in describing threats, property violations, typical victims, and what an attacker does. After gathering discovered threats and mitigation strategies, this information should be documented and prioritized [13, 15].
Table 1: STRIDE Threat Categories

<table>
<thead>
<tr>
<th>Threat</th>
<th>Property Violated</th>
<th>Threat Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>Spoofing</td>
<td>Authentication</td>
</tr>
<tr>
<td>T</td>
<td>Tampering with data</td>
<td>Integrity</td>
</tr>
<tr>
<td>R</td>
<td>Repudiation</td>
<td>Non-repudiation</td>
</tr>
<tr>
<td>I</td>
<td>Information disclosure</td>
<td>Confidentiality</td>
</tr>
<tr>
<td>D</td>
<td>Denial of service</td>
<td>Availability</td>
</tr>
<tr>
<td>E</td>
<td>Elevation of privilege</td>
<td>Authorization</td>
</tr>
</tbody>
</table>

This method is easy to adopt but can be time consuming [15, 14]. STRIDE’s main issue is that the number of threats can grow rapidly as a system increases in complexity. Scandariato et al., in their descriptive study of Microsoft’s threat modeling technique, show that the STRIDE method has a moderately low rate of false positives and a moderately high rate of false negatives [28]. STRIDE has been successfully applied to cyber-only and cyber-physical systems [14, 15, 20, 28, 40].

Even though Microsoft no longer maintains STRIDE, it is implemented as part of the Microsoft Secure Development Lifecycle (SDL) with the Threat Modeling Tool, which is still available [29].

Several authors represent modified STRIDE methods. Martins et al., in their presentation Towards a Systematic Threat Modeling Approach for Cyber-Physical Systems, use the STRIDE method with NIST guidelines instead of Microsoft security mediation strategies [30]. Microsoft developed another similar method called DREAD, which is also a mnemonic (Damage potential, Reproducibility, Exploitability, Affected users, Discoverability) with a different approach for assessing threats. It assigns one of three values (0, 5, 10) to the first four categories and one of four values (0, 5, 9, 10) to the last category, which “allows for an average value to be calculated to represent the risk of the entire system” [3, 33].
The Process for Attack Simulation and Threat Analysis (P.A.S.T.A) is a risk-centric threat modeling framework developed in 2012 by Tony UcedaVélez. It contains seven stages, each with multiple activities, which are illustrated in Figure 2 [31, 32, 16].

**Figure 2: PASTA Stages**

PASTA aims to bring business objectives and technical requirements together [22]. It uses a variety of design and elicitation tools in different stages. For example, high-level architectural diagrams are used
during stage two for identifying the technical scope. DFDs are used in stage three. During stage six, attack trees and use and abuse cases are built for analysis and attack modeling [31, 16].

This method elevates the threat modeling process to a strategic level by involving key decision makers and requiring security input from operations, governance, architecture, and development [21]. Widely regarded as a risk-centric framework, PASTA has an attacker-centric perspective. In the end, the process produces an asset-centric output in the form of threat enumeration and scoring [31, 21].

UcedaVélez and Marco Morana developed very rich documentation for the method to help with this laborious and extensive process [32].
LINDDUN

LINDDUN (Linkability, Identifiability, Non-Repudiation, Detectability, Disclosure of Information, Unawareness, Non-Compliance) is a threat modeling method that focuses on privacy concerns and can be used for data security [12]. Similar to STRIDE, this method is a mnemonic, meaning the threat categories in question are coded in the method name. Consisting of six steps, (see Figure 3), LINDDUN provides a systematic approach to privacy assessment [34, 35].

![Figure 3: LINDDUN Methodology Steps [34]](image)

LINDDUN starts with a DFD of the system that defines the system’s data flows, data stores, processes, and external entities. Systematically iterating over all model elements and analyzing them from the threat categories point of view, LINDDUN users identify a threat’s applicability to the system and build threat trees [12, 34, 35].

Steps 2 and 3 are essentially questionnaires that guide the user through the initial analysis process of identifying the threats in the system. Step 2 involves mapping threat categories to the parts of the system where they may appear. (See Figure 4 for an example.) Step 3 involves identifying scenarios in which these threats could occur. The rest of the process finds solutions and mitigation strategies [12].
One of the strong features of the LINDDUN method is its extensive privacy knowledgebase and documentation [34]. The LINDDUN method is labor intensive and time consuming. It suffers from the same issues as STRIDE—the number of threats can grow rapidly as a system increases in complexity. Wuyts et al., in their presentation *Effective and Efficient Privacy Threat Modeling Through Domain Refinements*, also noticed that efficiency and effectiveness of the method is negatively impacted by generically applicable threats [12].

![Figure 4: LINDDUN Mapping Step [12]](image)
The Common Vulnerability Scoring System (CVSS) is a method that “capture[s] the principal characteristics of a vulnerability, and produce[s] a numerical score reflecting its severity” [37].

It was developed by NIST [38] and is maintained by the Forum of Incident Response and Security Team (FIRST) [37] with support and contributions from the CVSS Special Interest Group [44]. The CVSS provides users of the method with a common and standardized scoring system within different cyber and cyber-physical platforms [37, 3]. A CVSS score can be computed by a calculator that is available online [38].

As illustrated in Figure 5, the CVSS consists of three metric groups (Base, Temporal, and Environmental) with a set of metrics in each [37].

A CVSS score is computed based on values assigned by an analyst for each metric. The equations used for this process are not clear, but all metrics are explained in the documentation quite extensively.

The method is widely used, despite some concerns related to the non-transparent score calculations and possible inconsistencies produced by different judging “experts” [3]. The CVSS method is often used in combination with other threat modeling methods.
Attack Trees

Using attack trees to model threats is one of the oldest and most widely applied techniques on cyber-only systems as well as cyber-physical and physical systems [3]. Developed by Bruce Schneider in 1999, it was initially applied as its own method and has since been combined with other methods and frameworks [17, 3].

Attack trees are essentially diagrams that depict attacks on a system in tree form. The tree root is the goal for the attack, and the leaves are ways to achieve that goal [17]. Each goal is represented as a separate tree. Thus, the system threat analysis produces a set of attack trees.

![Figure 6: Attack Tree Examples](image)

Usually it takes a few iterations of decomposing the goal to build the tree. Once all leaf nodes are identified, markers of possibility can be assigned. These values should be assigned only after relevant research on the step is done [17]. While examining different methods to achieve the goal, it may become obvious that this can be accomplished in multiple ways.

To incorporate these different options into the tree, AND and OR nodes should be used. (See Figure 6; AND nodes are connected by a note that says “and,” indicating that both nodes must be performed to move to the next step. OR nodes are all other nodes.) In the case of a complex system, attack trees can be built for each component instead of for the whole system [2]. When attack trees are built, they can be used to make security decisions, see if the systems are vulnerable to an attack, and evaluate a specific type of attack [17].

Attack trees are easy to understand and adopt but are only useful when the system and security concerns are well understood. The method assumes that analysts have high cybersecurity expertise and thus does not provide guidelines for assessing sub-goals, attacks, or risks [2].

In recent years, this method has often been used in combination with other techniques and within frameworks like STRIDE, CVSS, and PASTA [3, 2, 31].
Persona non Grata

As a threat modeling method, Persona non Grata (PnG) focuses on the motivations and skills of human attackers. It characterizes users as archetypes that can misuse the system and forces analysts to view the system from an unintended use point of view [23].

When used, PnG can help visualize threats from the counterpart side, which can be helpful in the early stages of the threat modeling [15]. The idea is to “introduce” a technical expert to a potential attacker of the system and their skills, motivations, and goals that help the expert to see the system’s vulnerabilities and points of compromise from the other side [15].
PnG is easy to adopt but is rarely used or researched. It produces few false positives and has high consistency but tends to detect only a certain subset of threat types [15]. This technique fits well into the agile approach, which incorporates personas.  

1  http://www.agilemodeling.com/artifacts/personas.htm
Security Cards

Security Cards is a technique that centers on identifying unusual and complex attacks. It is not a formal method but more of a brainstorming technique [41]. With help from a deck of cards (see an example in Figure 8), analysts can answer questions about an attack, such as “by whom?” “why might the system be attacked?” “what assets are of interest?” and “how can these attacks be implemented?” [15].

![Security Card Example](Image)

This method uses a deck of 42 cards to facilitate threat discovery activities: Human Impact (9 cards), Adversary’s Motivations (13 cards), Adversary Resources (11 cards), and Adversary’s Methods (9 cards). The different categories within each dimension are shown in Table 2.

<table>
<thead>
<tr>
<th>Human Impact</th>
<th>Adversary’s Motivations</th>
<th>Adversary’s Resources</th>
<th>Adversary’s Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>the biosphere</td>
<td>access or convenience</td>
<td>expertise</td>
<td>attack cover-up</td>
</tr>
<tr>
<td>emotional well-being</td>
<td>curiosity or boredom</td>
<td>a future world</td>
<td>indirect attack</td>
</tr>
<tr>
<td>financial well-being</td>
<td>desire or obsession</td>
<td>impunity</td>
<td>manipulation or coercion</td>
</tr>
<tr>
<td>personal data</td>
<td>diplomacy or warfare</td>
<td>inside capabilities</td>
<td>multi-phase attack</td>
</tr>
<tr>
<td>physical well-being</td>
<td>malice or revenge</td>
<td>inside knowledge</td>
<td>physical attack</td>
</tr>
<tr>
<td>relationships</td>
<td>money</td>
<td>money</td>
<td>processes</td>
</tr>
<tr>
<td>societal well-being</td>
<td>politics</td>
<td>power and influence</td>
<td>technological attack</td>
</tr>
<tr>
<td>unusual impacts</td>
<td>protection</td>
<td>time</td>
<td>unusual methods</td>
</tr>
<tr>
<td></td>
<td>religion</td>
<td>tools</td>
<td></td>
</tr>
<tr>
<td></td>
<td>self-promotion</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>world view</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>unusual motivations</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Security Cards activities help identify almost all of the threat types but produce a high number of false positives and are better used to address non-standard situations [15]. Also, this method is rarely used in industry.
hTMM

The Hybrid Threat Modeling Method (hTMM) was developed by the Software Engineering Institute in 2018. It consists of a combination of SQUARE (Security Quality Requirements Engineering Method) [45], Security Cards, and PnG activities. The targeted characteristics of the method include no false positives, no overlooked threats, a consistent result regardless of who is doing the threat modeling, and cost-effectiveness [15].

The following are the main steps of the method:
1. Identify the system to be threat-modeled.
2. Apply Security Cards based on developer suggestions.
3. Remove unlikely PnGs (i.e., there are no realistic attack vectors).
4. Summarize the results using tool support.
5. Continue with a formal risk assessment method.

The hTMM was applied on one scenario of a cyber-physical system [15].
Quantitative Threat Modeling Method

This hybrid method consists of Attack Trees, STRIDE, and CVSS methods applied in synergy. It was introduced during the HotSoS² conference in Pittsburgh, PA in April 2016 by Bradley Potteiger, Goncalo Martins, and Xenofon Koutsoukos. The authors aimed to address a few pressing issues with threat modeling for cyber-physical systems that had complex interdependences among their components [3].

The first step of the Quantitative Threat Modeling Method (Quantitative TMM) is to build component attack trees for the five threat categories of STRIDE. This activity shows the dependencies among attack categories and low-level component attributes. (See Figure 9 for an example.) After that, the CVSS method is applied and scores are calculated for the components in the tree. (See Figure 10.)

Figure 9: Component Attack Tree [3]

Figure 10: CVSS Scoring for Tampering Attack Tree [3]
An additional goal for the method is to generate attack ports for individual components. These attack ports (effectively root nodes for the component attack trees) illustrate activities that can pass risk to the connected components. The scoring assists with the process of performing a system risk assessment. If an attack port is dependent on a component root node with a high-risk score, that attack port also has a high-risk score and has a high probability of being executed. The opposite is also true [3].

This method was used in a case study for a railway communications network. The resulting article, *Software and attack centric integrated threat modeling for quantitative risk assessment*, provides a detailed walkthrough of the method [3].
Trike was created as a security audit framework in 2005 that uses threat modeling as a technique [15]. It looks at threat modeling from a risk management and defensive perspective [42].

Trike, as with many other methods, starts with defining a system. The analyst must build a requirement model by enumerating and understanding the system’s actors, assets, intended actions, and rules. As a result of this step, an actor-asset-action matrix can be created, where the columns represent assets, and the rows represent actors.

Each cell of the matrix should be divided into four parts, one for each action of CRUD (creating, reading, updating, and deleting). In these cells, the analyst should assign one of three values: allowed action, disallowed action, or action with rules. A rule tree should be attached to each cell [42, 43].

After defining requirements, a DFD is built. Each element is mapped to a selection of actors and assets. Iterating through the DFD, the analyst identifies threats, which fall into one of two categories: elevations of privilege or denials of service [42, 43]. Each discovered threat becomes a root node in an attack tree [42].

To assess the risk of attacks that may impact assets through CRUD, Trike uses a five-point scale for each action, based on its probability. Actors are rated on five-point scales for the risks they are assumed to present (lower number = higher risk) to the asset. Also, actors are evaluated on a three-dimensional scale (always, sometimes, never) for each action they may perform on each asset.

The Trike scale system seems too vague to represent a formal method. Unfortunately, Trike version 2.0 is not well maintained, and there is no documentation, even though its site is up and running.
VAST Modeling

The Visual, Agile, and Simple Threat (VAST) Modeling method was created by Anurag Agarwal and is based on ThreatModeler, an automated threat modeling platform [15]. The fundamental value of the method is the scalability and usability that allow it to be adopted in large organizations throughout the entire infrastructure to produce actionable and reliable results for different stakeholders [22].

Recognizing differences in operations and concerns among development and infrastructure teams, VAST requires creating two types of models: application threat models and operational threat models. Application threat models use process flow diagrams, representing the architectural point of view. Operational threat models are created with an attacker point of view in mind based on DFDs [15, 22]. This approach allows for the integration of VAST into the organization’s development and DevOps lifecycles [22].
OCTAVE

The Operationally Critical Threat, Asset, and Vulnerability Evaluation (OCTAVE) method is a risk-based strategic assessment and planning method for cybersecurity [36]. It was created by the CERT Division of the Software Engineering Institute in 2003 and refined in 2005. OCTAVE focuses on assessing organizational risks and does not address technological risks. Its main aspects are operational risk, security practices, and technology [22, 43].

OCTAVE has three phases, which can be seen in Figure 11 [36]:
1. Build asset-based threat profiles. (This is an organizational evaluation.)
2. Identify infrastructure vulnerability. (This is an evaluation of the information infrastructure.)
3. Develop a security strategy and plans. (This is an identification of risks to the organization’s critical assets and decision making.)

OCTAVE evaluates activities, not continuous processes. It is primarily designed for large organizations, but the OCTAVE-S method was created specifically for small (20-80 people) organizations [36]. The method is in-depth but flexible. The downsides of OCTAVE are that the process requires a significant time commitment, and the documentation is large and vague [43]. There are planned updates to OCTAVE that may impact its downsides, but the exact effects are currently unknown.
Conclusion

Threat modeling can help make your product more secure and trustworthy. This paper presented twelve different threat modeling methods. Some are typically used alone, some are usually used in conjunction with others, and some are examples of how different methods can be combined.

Choosing what method is best for a project requires thinking about if there are any specific areas you want to target (risk, security, privacy), how long you have to perform threat modeling, how much experience you have with threat modeling, how involved stakeholders want to be, and more. Some features of each threat modeling method are summarized in Table 3. These methods can all be used within an agile environment, depending on the timeframe of the sprint and how often the modeling is repeated.

Table 3: Threat Modeling Methods Features

<table>
<thead>
<tr>
<th>Threat Modeling Method</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>STRIDE</td>
<td>• Helps identify relevant mitigating techniques</td>
</tr>
<tr>
<td></td>
<td>• Is the most mature</td>
</tr>
<tr>
<td></td>
<td>• Is easy to use but is time consuming</td>
</tr>
<tr>
<td>PASTA</td>
<td>• Helps identify relevant mitigating techniques</td>
</tr>
<tr>
<td></td>
<td>• Directly contributes to risk management</td>
</tr>
<tr>
<td></td>
<td>• Encourages collaboration among stakeholders</td>
</tr>
<tr>
<td></td>
<td>• Contains built-in prioritization of threat mitigation</td>
</tr>
<tr>
<td></td>
<td>• Is laborious but has rich documentation</td>
</tr>
<tr>
<td>LINDDUN</td>
<td>• Helps identify relevant mitigation techniques</td>
</tr>
<tr>
<td></td>
<td>• Contains built-in prioritization of threat mitigation</td>
</tr>
<tr>
<td></td>
<td>• Can be labor intensive and time consuming</td>
</tr>
<tr>
<td>CVSS</td>
<td>• Contains built-in prioritization of threat mitigation</td>
</tr>
<tr>
<td></td>
<td>• Has consistent results when repeated</td>
</tr>
<tr>
<td></td>
<td>• Automated components</td>
</tr>
<tr>
<td></td>
<td>• Has score calculations that are not transparent</td>
</tr>
<tr>
<td>Attack Trees</td>
<td>• Helps identify relevant mitigation techniques</td>
</tr>
<tr>
<td></td>
<td>• Has consistent results when repeated</td>
</tr>
<tr>
<td></td>
<td>• Is easy to use if you already have a thorough understanding of the system</td>
</tr>
<tr>
<td>Persona non Grata</td>
<td>• Helps identify relevant mitigation techniques</td>
</tr>
<tr>
<td></td>
<td>• Directly contributes to risk management</td>
</tr>
<tr>
<td></td>
<td>• Has consistent results when repeated</td>
</tr>
<tr>
<td></td>
<td>• Tends to detect only some subsets of threats</td>
</tr>
<tr>
<td>Security Cards</td>
<td>• Encourages collaboration among stakeholders</td>
</tr>
<tr>
<td></td>
<td>• Targets out-of-the-ordinary threats</td>
</tr>
<tr>
<td></td>
<td>• Leads to many false positives</td>
</tr>
<tr>
<td>Threat Modeling Method</td>
<td>Features</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------</td>
</tr>
</tbody>
</table>
| hTMM                   | - Contains built-in prioritization of threat mitigation  
|                        | - Encourages collaboration among stakeholders  
|                        | - Has consistent results when repeated |
| Quantitative TMM       | - Contains built-in prioritization of threat mitigation  
|                        | - Has automated components  
|                        | - Has consistent results when repeated |
| Trike                  | - Helps identify relevant mitigation techniques  
|                        | - Directly contributes to risk management  
|                        | - Contains built-in prioritization of threat mitigation  
|                        | - Encourages collaboration among stakeholders  
|                        | - Has automated components  
|                        | - Has vague, insufficient documentation |
| VAST Modeling          | - Helps identify relevant mitigation techniques  
|                        | - Directly contributes to risk management  
|                        | - Contains built-in prioritization of threat mitigation  
|                        | - Encourages collaboration among stakeholders  
|                        | - Has consistent results when repeated  
|                        | - Has automated components  
|                        | - Is explicitly designed to be scalable  
|                        | - Has little publicly available documentation |
| OCTAVE                 | - Helps identify relevant mitigation techniques  
|                        | - Directly contributes to risk management  
|                        | - Contains built-in prioritization of threat mitigation  
|                        | - Encourages collaboration among stakeholders  
|                        | - Has consistent results when repeated  
|                        | - Is explicitly designed to be scalable  
|                        | - Is time consuming and has vague documentation |
Bibliography

URLs are valid as of the publication date of this document.


