Reference <https://insights.sei.cmu.edu/blog/threat-modeling-12-available-methods/>

**Threat Modeling: 12 Available Methods**

****[**NATALIYA SHEVCHENKO**](https://insights.sei.cmu.edu/authors/nataliya-shevchenko/)

**DECEMBER 3, 2018**

Almost all software systems today face a variety of threats, and the number of threats grows as technology changes. [Malware that exploits software vulnerabilities grew 151 percent in the second quarter of 2018](https://www.computerweekly.com/news/252449265/WannaCry-and-NotPetya-inspiring-new-attacks), and [cyber-crime damage costs are estimated to reach $6 trillion annually by 2021](https://www.forbes.com/sites/cognitiveworld/2018/09/05/a-scoville-heat-scale-for-measuring-cybersecurity/#4a2bced43327). Threats can come from outside or within organizations, and they can have devastating consequences. Attacks can disable systems entirely or lead to the leaking of sensitive information, which would diminish consumer trust in the system provider. To prevent threats from taking advantage of system flaws, administrators can use [threat-modeling](https://en.wikipedia.org/wiki/Threat_model) methods to inform defensive measures. In this blog post, I summarize 12 available threat-modeling methods.

Threat-modeling methods are used to create

* an abstraction of the system
* profiles of potential attackers, including their goals and methods
* a catalog of potential threats that may arise

Many threat-modeling methods have been developed. They can be combined to create a more robust and well-rounded view of potential threats. Not all of them are comprehensive; some are abstract and others are people-centric. Some methods focus specifically on risk or privacy concerns.

Threat modeling should be performed early in the development cycle when potential issues can be caught early and remedied, preventing a much costlier fix down the line. Using threat modeling to think about security requirements can lead to proactive architectural decisions that help reduce threats from the start. Threat modeling can be particularly helpful in the area of [cyber-physical systems](https://en.wikipedia.org/wiki/Cyber-physical_system).

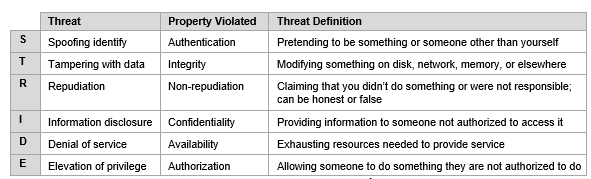
Cyber-physical systems integrate software technology into physical infrastructures, such as smart cars, smart cities, or smart grids. While innovative, cyber-physical systems are vulnerable to threats that manufacturers of traditional physical infrastructures may not consider. Performing threat modeling on cyber-physical systems with a variety of stakeholders can help catch threats across a wide spectrum of threat types.

The 12 threat-modeling methods summarized in this post come from a variety of sources and target different parts of the process. No one threat-modeling method is recommended over another; organizations should choose which method to use based on the specific needs of their project. I encourage readers interested in more detailed information about these methods to read our [SEI white paper](https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=524448) on the same topic.

**STRIDE and Associated Derivations**

Invented in 1999 and adopted by Microsoft in 2002, [STRIDE](https://en.wikipedia.org/wiki/STRIDE_%28security%29) is currently the most mature threat-modeling method. STRIDE has evolved over time to include new threat-specific tables and the variants [STRIDE-per-Element](https://cloudblogs.microsoft.com/microsoftsecure/2007/10/29/the-stride-per-element-chart/) and [STRIDE-per-Interaction](https://www.ffri.jp/assets/files/monthly_research/MR201610_STRIDE_Variants_and_Security_Requirements-based_Threat_Analysis_ENG.pdf).

STRIDE evaluates the system detail design. It models the in-place system. By building [data-flow diagrams](http://en.wikipedia.org/wiki/Data_flow_diagram) (DFDs), STRIDE is used to identify system entities, events, and the boundaries of the system. STRIDE applies a general set of known threats based on its name, which is a mnemonic, as shown in the following table:

[](https://insights.sei.cmu.edu/media/images/Table_1.original.png)

**Table 1: STRIDE Threat Categories**

STRIDE has been successfully applied to cyber-only and cyber-physical systems. Although Microsoft no longer maintains STRIDE, it is implemented as part of the [Microsoft Security Development Lifecycle](https://www.microsoft.com/en-us/sdl/default.aspx) (SDL) with the [Threat Modeling Tool](https://www.microsoft.com/en-us/securityengineering/sdl/threatmodeling), which is still available. Microsoft also developed a similar method called [DREAD](https://en.wikipedia.org/wiki/DREAD_(risk_assessment_model)), which is also a mnemonic (damage potential, reproducibility, exploitability, affected users, discoverability) with a different approach for assessing threats.

**PASTA**

[The Process for Attack Simulation and Threat Analysis (PASTA)](http://securesoftware.blogspot.com/2012/09/rebooting-software-security.html) is a risk-centric threat-modeling framework developed in 2012. It contains seven stages, each with multiple activities, which are illustrated in Figure 1 below:

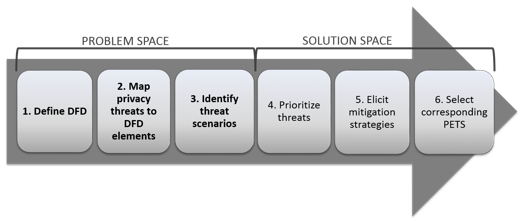
[](https://insights.sei.cmu.edu/media/images/Figure_1.TMM.original.png)

**Figure 1: Adapted from** [***Threat Modeling w/PASTA: Risk Centric Threat Modeling Case Studies***](https://www.slideshare.net/marco_morana/owasp-app-seceu2011version1)

PASTA aims to bring business objectives and technical requirements together. It uses a variety of design and elicitation tools in different stages. 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. Widely regarded as a risk-centric framework, PASTA employs an attacker-centric perspective to produce an asset-centric output in the form of threat enumeration and scoring.

**LINDDUN**

[LINDDUN](https://linddun.org/) (linkability, identifiability, nonrepudiation, detectability, disclosure of information, unawareness, noncompliance) focuses on privacy concerns and can be used for data security. Consisting of six steps, (see Figure 2), LINDDUN provides a systematic approach to privacy assessment.

[](https://insights.sei.cmu.edu/media/images/Figure_2.TMM.original.png)

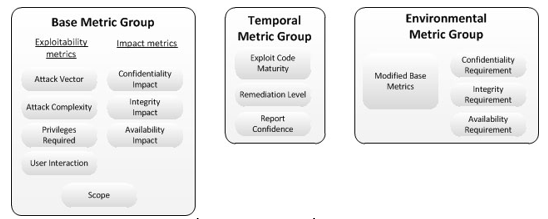
**Figure 2: LINDDUN Steps**

LINDDUN starts with a DFD of the system that defines the system's data flows, data stores, processes, and external entities. By systematically iterating over all model elements and analyzing them from the point of view of threat categories, LINDDUN users identify a threat's applicability to the system and build threat trees.

**CVSS**

[The Common Vulnerability Scoring System (CVSS)](https://www.first.org/cvss/specification-document) captures the principal characteristics of a vulnerability and produces a numerical severity score. CVSS was developed by [NIST](https://www.nist.gov/) and is maintained by the [Forum of Incident Response and Security Teams (FIRST)](https://www.first.org/) with support and contributions from the [CVSS Special Interest Group](https://www.first.org/cvss/v2/meetings). The CVSS provides users a common and standardized scoring system within different cyber and cyber-physical platforms. A CVSS score can be computed by a calculator that is [available online](https://nvd.nist.gov/vuln-metrics/cvss).

As shown in Figure 3, the CVSS consists of three metric groups (Base, Temporal, and Environmental) with a set of metrics in each.

[](https://insights.sei.cmu.edu/media/images/Figure_3.TMM.original.png)

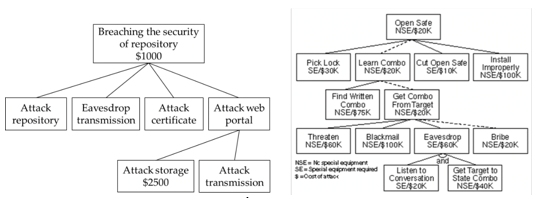
**Figure 3: CVSS v3.0 Metric Groups**

A CVSS score is derived from values assigned by an analyst for each metric. The metrics are explained extensively in the documentation. 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, cyber-physical systems, and purely physical systems](https://dl.acm.org/citation.cfm?doid=2898375.2898390). Attack trees were initially applied as a stand-alone method and [has since been combined with other methods and frameworks](https://www.schneier.com/academic/archives/1999/12/attack_trees.html).

Attack trees are 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. Each goal is represented as a separate tree. Thus, the system threat analysis produces a set of attack trees. See examples in Figure 4.

[](https://insights.sei.cmu.edu/media/images/Figure_4.TMM.original.png)

**Figure 4: Attack Tree Examples**

In the case of a complex system, attack trees can be built for each component instead of for the whole system. Administrators can build attack trees and use them to inform security decisions, to determine whether the systems are vulnerable to an attack, and to evaluate a specific type of attack.

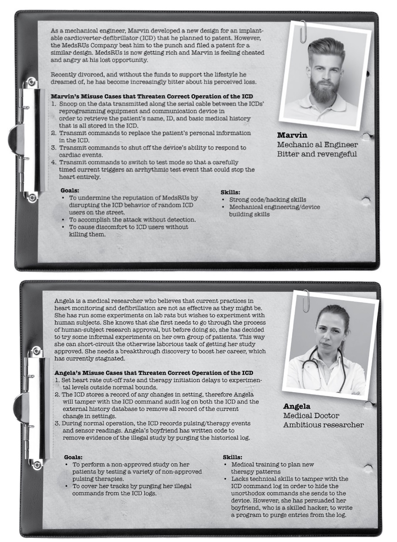
In recent years, this method has often been used in combination with other techniques and within frameworks such as STRIDE, CVSS, and PASTA.

**Persona non Grata**

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. See examples in Figure 5.

[PnG can help visualize threats from the counterpart side, which can be helpful in the early stages of the threat modeling](http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=516617). The idea is to introduce a technical expert to a potential attacker of the system and examine the attacker's skills, motivations, and goals. This analysis helps the expert understand the system's vulnerabilities from the point of view of an attacker.

PnG fits well into the [Agile](https://insights.sei.cmu.edu/blog?tag=agile) approach, which uses [personas](http://www.agilemodeling.com/artifacts/personas.htm).

[](https://insights.sei.cmu.edu/media/images/Figure_5.TMM.original.png)

**Figure 5: Examples of Personae non Grata**

**Security Cards**

[Security Cards](http://securitycards.cs.washington.edu/index.html) identify unusual and complex attacks. They are not a formal method but, rather, a kind of brainstorming technique. With help from a deck of cards (see an example in Figure 6), analysts can answer questions about an attack, such as

* Who might attack?
* Why might the system be attacked?
* What assets are of interest?
* How can these attacks be implemented?"

[](https://insights.sei.cmu.edu/media/images/Figure_6.TMM.original.png)

**Figure 6: Security Card Example**

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.

[](https://insights.sei.cmu.edu/media/images/Table_2.original.png)

**Table 2: Security-Card Dimensions**

**hTMM**

The [Hybrid Threat Modeling Method (hTMM)](https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=516617) was developed by the SEI in 2018. It consists of a combination of [SQUARE (Security Quality Requirements Engineering Method)](https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=7657), 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.

The main steps of the method are

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.

**Quantitative Threat Modeling Method**

This hybrid method consists of attack trees, STRIDE, and CVSS methods applied in synergy. It aims to address a few pressing issues with threat modeling for cyber-physical systems that had complex interdependences among their components.

The first step of the [Quantitative Threat Modeling Method (Quantitative TMM)](https://dl.acm.org/citation.cfm?doid=2898375.2898390) 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. After that, the CVSS method is applied and scores are calculated for the components in the tree.

**Trike**

[Trike](http://www.octotrike.org/home.shtml) was created as a security audit framework that uses threat modeling as a technique. It looks at threat modeling from a risk-management and defensive perspective.

As with many other methods, Trike starts with defining a system. The analyst builds a requirement model by enumerating and understanding the system's actors, assets, intended actions, and rules. This step creates an actor-asset-action matrix in which the columns represent assets and the rows represent actors.

Each cell of the matrix is divided into four parts, one for each action of [CRUD](https://en.wikipedia.org/wiki/Create,_read,_update_and_delete) (creating, reading, updating, and deleting). In these cells, the analyst assigns one of three values: allowed action, disallowed action, or action with rules. A [rule tree](https://en.wikipedia.org/wiki/Decision_tree) is attached to each cell.

After defining requirements, a [data flow diagram](https://en.wikipedia.org/wiki/Data_flow_diagram) (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](https://en.wikipedia.org/wiki/Privilege_escalation) or [denials of service](https://en.wikipedia.org/wiki/Denial-of-service_attack). Each discovered threat becomes a root node in an attack tree.

To assess the risk of attacks that may affect 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.

**VAST Modeling**

The [Visual, Agile, and Simple Threat (VAST) Modeling](https://en.wikipedia.org/wiki/Threat_model#VAST) method is based on [ThreatModeler](https://threatmodeler.com/), an automated threat-modeling platform. Its scalability and usability allow it to be adopted in large organizations throughout the entire infrastructure to produce actionable and reliable results for different stakeholders.

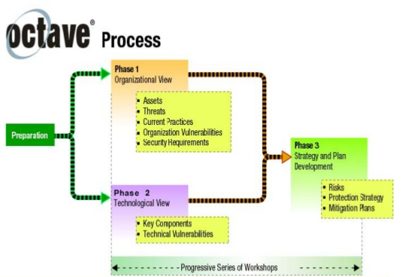
Recognizing differences in operations and concerns among development and infrastructure teams, VAST requires creating two types of models: application threat models and operational threat models. Application threat models use [process-flow diagrams](https://en.wikipedia.org/wiki/Process_Flow_diagram), representing the architectural point of view. Operational threat models are created from an attacker point of view based on DFDs. This approach allows for the integration of VAST into the organization's development and [DevOps](https://insights.sei.cmu.edu/category/devops/) lifecycles.

**OCTAVE**

The [Operationally Critical Threat, Asset, and Vulnerability Evaluation (OCTAVE)](https://resources.sei.cmu.edu/library/Asset-view.cfm?assetid=51546) method is a risk-based strategic assessment and planning method for cybersecurity. It was created by the [CERT Division](https://www.sei.cmu.edu/about/divisions/cert/index.cfm) of the SEI 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.

As shown in Figure 7, OCTAVE has three phases.

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.)

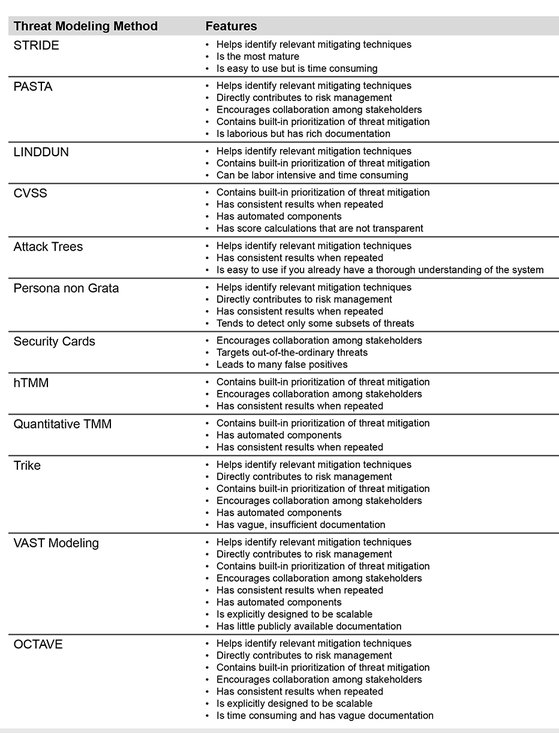
[](https://insights.sei.cmu.edu/media/images/Figure_7.TMM.original.png)

**Figure 7: OCTAVE Phases**

**Wrapping Up and Looking Ahead**

Threat modeling can help make your product more secure and trustworthy. This post presented 12 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. A future SEI blog post will provide guidance on how to evaluate these models for use in specific contexts.

To choose what method is best for your project, you need to think about 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, etc. Table 3 summarizes features of each threat modeling method. These methods can all be used within an [Agile](https://insights.sei.cmu.edu/blog?tag=agile) environment, depending on the timeframe of the sprint and how often the modeling is repeated.

[](https://insights.sei.cmu.edu/media/images/Threat20Modeling20Table.original.jpg)

**Table 3: Features of Threat-Modeling Methods**

**Additional Resources**

Read the SEI White Paper, [Threat Modeling: A Summary of Available Methods](https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=524448), on which this post is based.

Read the SEI Technical Note, [A Hybrid Threat Modeling Method](http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=516617) by Nancy Mead and colleagues.

Read [Evaluation of Threat Modeling Methodologies](http://resources.sei.cmu.edu/library/asset-view.cfm?assetID=474197) by Forrest Shull.

Read the [SEI blog post The Hybrid Threat Modeling Method](https://insights.sei.cmu.edu/blog/the-hybrid-threat-modeling-method/) by Nancy Mead and Forrest Shull.

Browse SEI [OCTAVE-related information](https://resources.sei.cmu.edu/library/asset-view.cfm?assetID=309051).

Read an SEI Technical Report about [Security Quality Requirements Engineering](https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=7657) (SQUARE).