Carnegie Mellon University
Software Engineering Institute

THREAT MODELING: A SUMMARY OF AVAILABLE METHODS

Nataliya Shevchenko, Timothy A. Chick, Paige O'Riordan, Thomas Patrick Scanlon, PhD, & Carol Woody, PhD
July 2018

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

List of Figures

List of Tables

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].

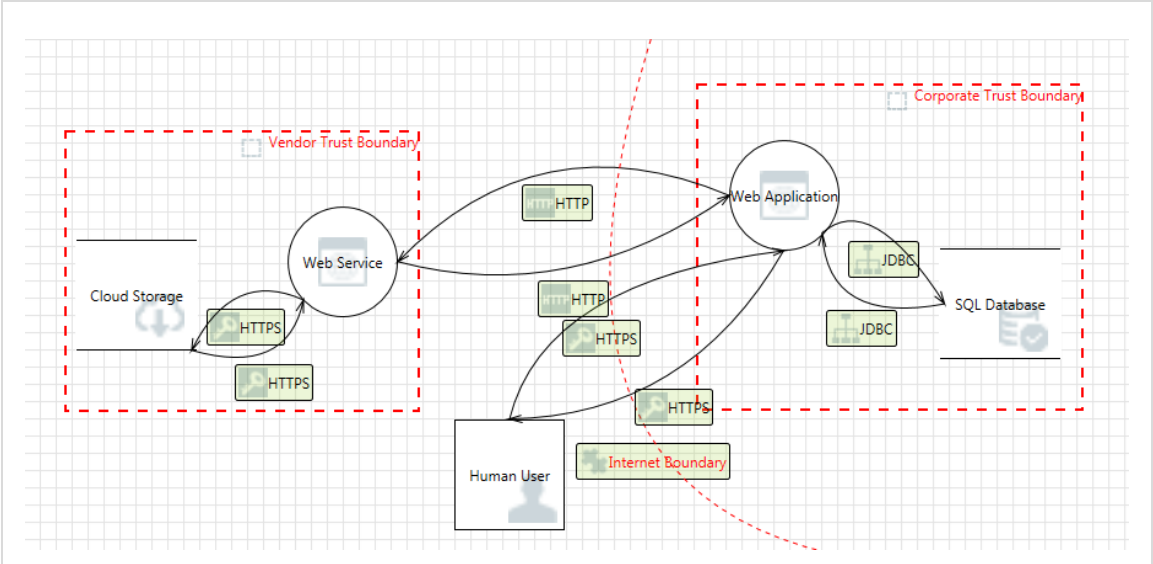

Figure 1: Data Flow Diagram with System Boundaries

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

	Threat	Property Violated	Threat Definition
S	Spoofing identify	Authentication	Pretending to be something or someone other than yourself
T	Tampering with data	Integrity	Modifying something on disk, network, memory, or elsewhere
R	Repudiation	Non-repudiation	Claiming that you didn't do something or were not responsible; can be honest or false
I	Information disclosure	Confidentiality	Providing information to someone not authorized to access it
D	Denial of service	Availability	Exhausting resources needed to provide service
E	Elevation of privilege	Authorization	Allowing someone to do something they are not authorized to do

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].

PASTA

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].

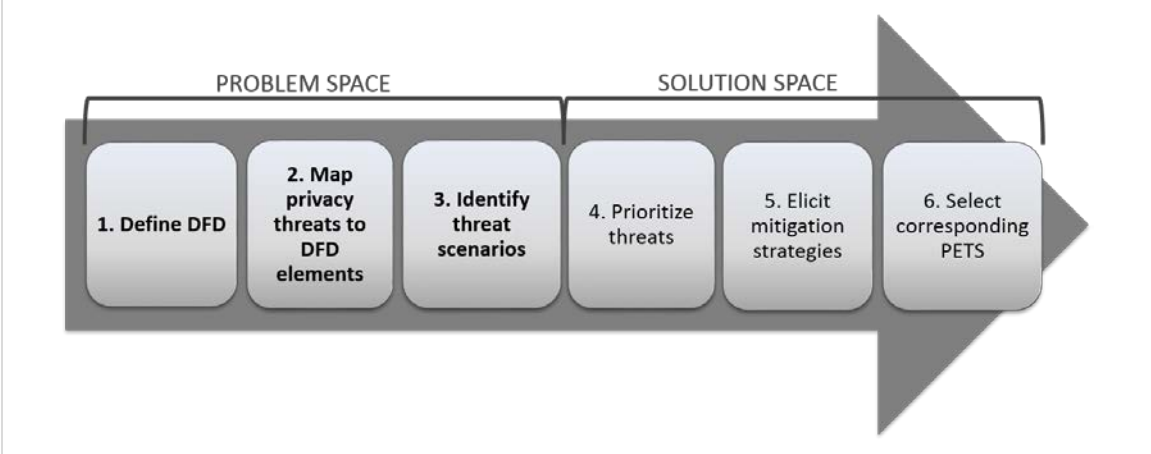

Figure 3: LINDDUN Methodology Steps [34]

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].

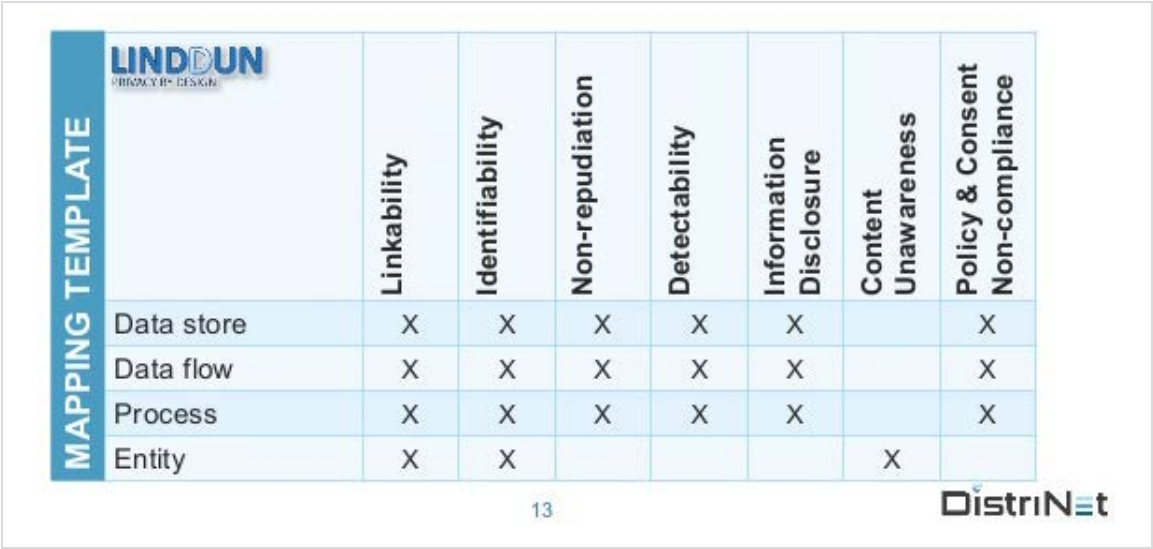

Figure 4: LINDDUN Mapping Step [12]

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].

CVSS

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].

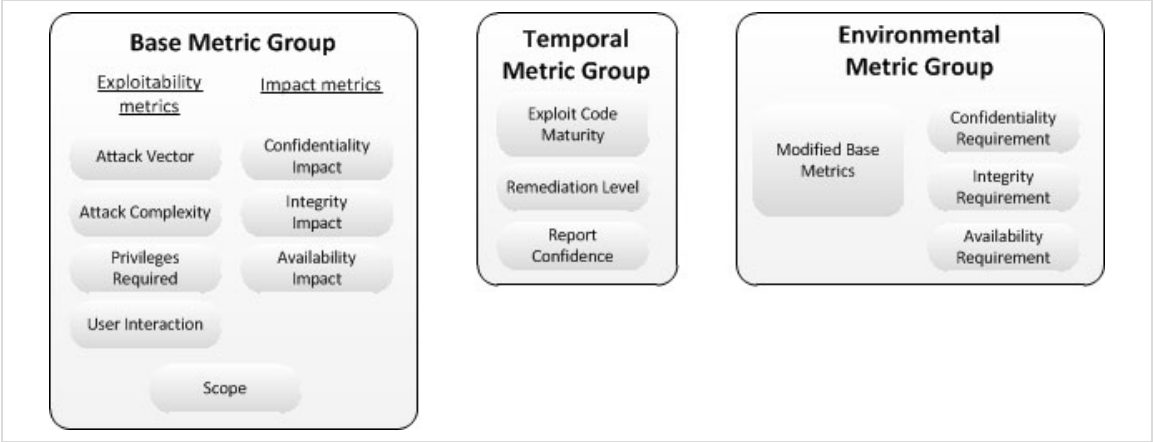

Figure 5: CVSS v3.0 Metric Groups [37]

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.

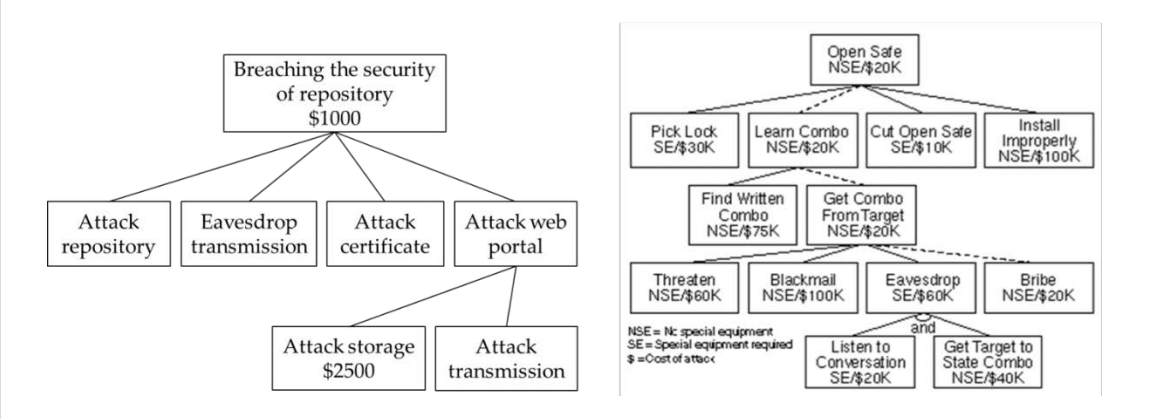

Figure 6: Attack Tree Examples [2]

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].

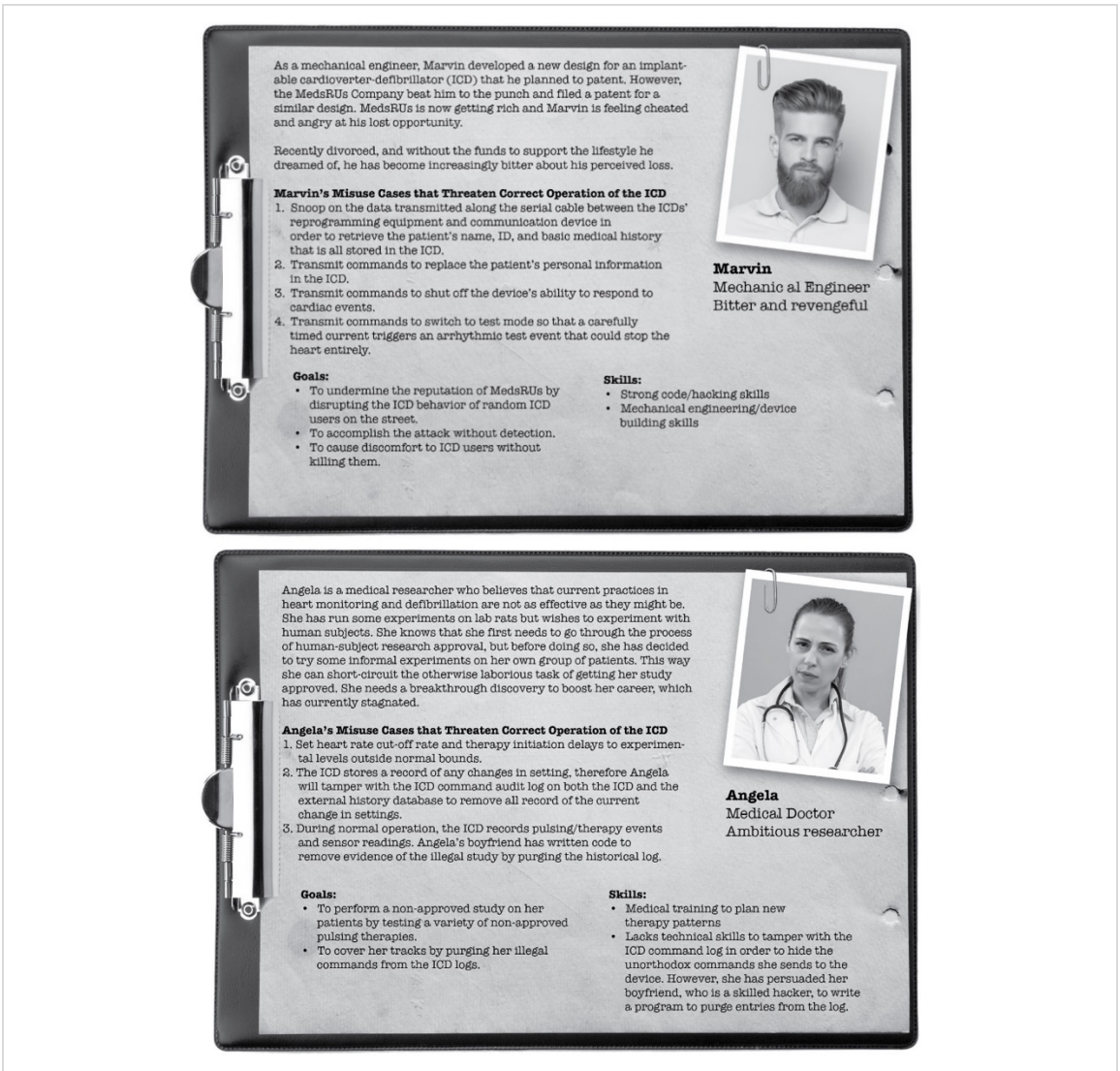

Figure 7: Examples of Personae non Grata [15]

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]

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].

Figure 8: Security Card Example [15]

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 2: Security Cards Dimensions

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

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 HotSoS[2] 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.)

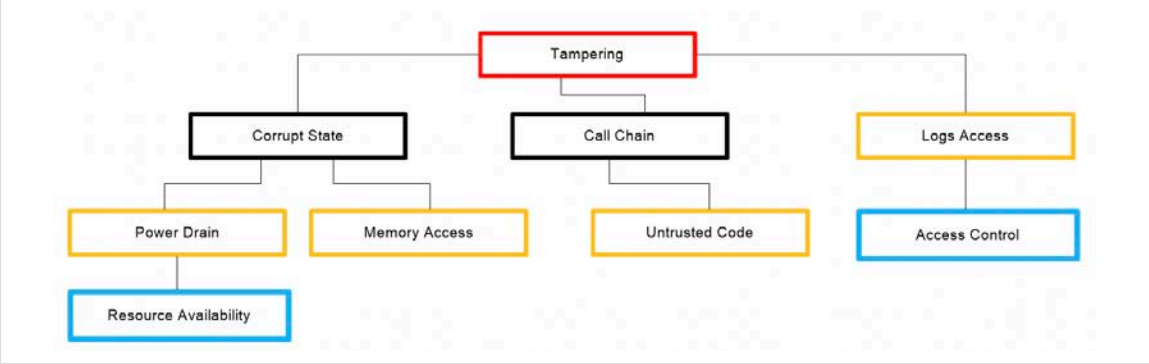

Figure 9: Component Attack Tree [3]

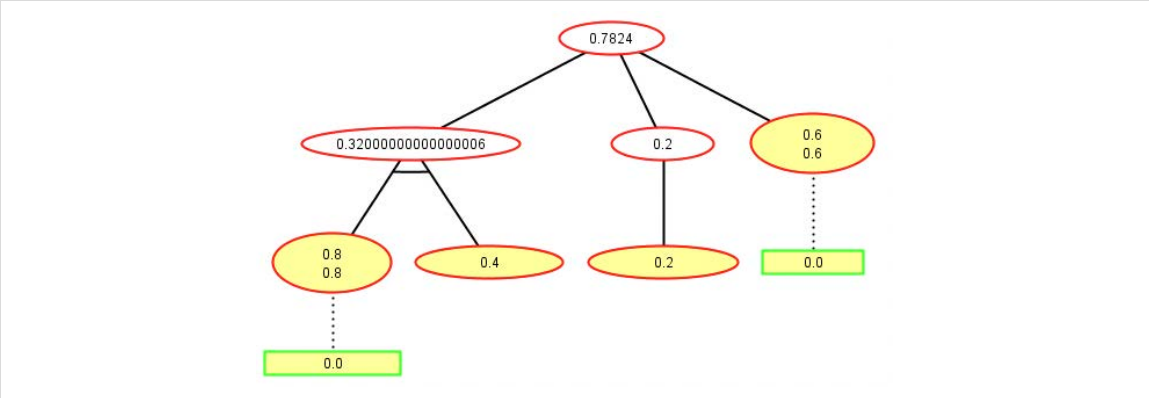

Figure 10: CVSS Scoring for Tampering Attack Tree [3]

2 Hot Topics in Science of Security

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

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.)

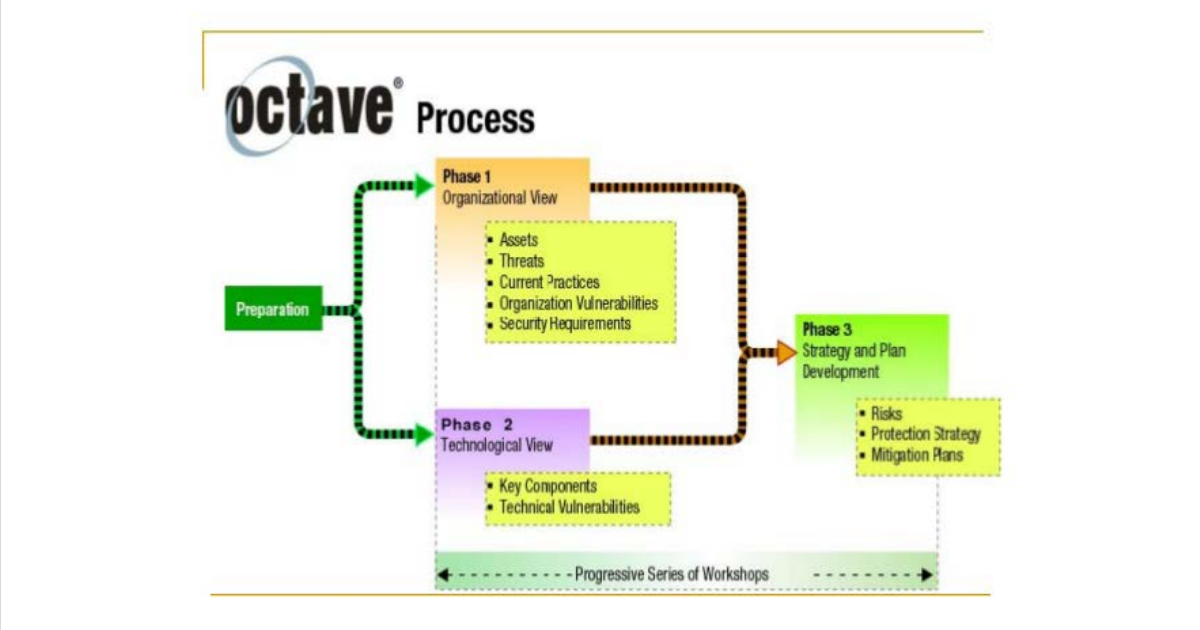

Figure 11: OCTAVE Phases [36]

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

Threat Modeling Method	Features
STRIDE	• Helps identify relevant mitigating techniques • Is the most mature • Is easy to use but is time consuming
PASTA	• Helps identify relevant mitigating techniques • Directly contributes to risk management • Encourages collaboration among stakeholders • Contains built-in prioritization of threat mitigation • Is laborious but has rich documentation
LINDDUN	• Helps identify relevant mitigation techniques • Contains built-in prioritization of threat mitigation • Can be labor intensive and time consuming
CVSS	• Contains built-in prioritization of threat mitigation • Has consistent results when repeated • Automated components • Has score calculations that are not transparent
Attack Trees	• Helps identify relevant mitigation techniques • Has consistent results when repeated • Is easy to use if you already have a thorough understanding of the system
Persona non Grata	• Helps identify relevant mitigation techniques • Directly contributes to risk management • Has consistent results when repeated • Tends to detect only some subsets of threats
Security Cards	• Encourages collaboration among stakeholders • Targets out-of-the-ordinary threats • Leads to many false positives

Threat Modeling Method	Features
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.

[1] David, N.; David, A.; Hansen, R. R.; Larsen, K. G.; Legay, A.; Olesen, M. C.; & Probst, C. W. Modelling Social-Technical Attacks with Timed Automata. Pages 21-28. In Proceedings of the 7th ACM CCS International Workshop on Managing Insider Security Threats. Conference on Computer and Communications Security. October 2015. DOI 10.1145/2808783.2808787.

[2] Cheung, C. Y. Threat Modeling Techniques [Master's Thesis]. Delft University of Technology. November 2016. http://www.safety-and-security.nl/uploads/cfsas/attachments/SPM5440%20%26%20WM0804TU%20-%20Threat%20modeling%20techniques%20-%20CY%20Cheung.pdf

[3] Potteiger, B.; Martins, G.; & Koutsoukos, X. Software and attack centric integrated threat modeling for quantitative risk assessment. Pages 99-108. In *Proceedings of the Symposium and Bootcamp on the Science of Security*. April 2016. DOI 10.1145/2898375.2898390.

[4] Agrawal, A.; Ahmed, C. M.; & Chang, E. Poster: Physics-Based Attack Detection for an Insider Threat Model in a Cyber-Physical System. Pages 821-823. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*. June 2018. DOI 10.1145/3196494.3201587.

[5] Datta, A. & Rahman, M. A. Cyber Threat Analysis Framework for the Wind Energy Based Power System. Pages 81-92. In *Proceedings of the 2017 Workshop on Cyber-Physical Systems Security and PrivaCy*. Conference on Computer and Communications Security. November 2017. DOI 10.1145/3140241.3140247.

[6] Humayed, A. & Luo, B. Cyber-physical security for smart cars: taxonomy of vulnerabilities, threats, and attacks. Pages 252-253. In *Proceedings of the ACM/IEEE Sixth International Conference on Cyber-Physical Systems.* April 2015. DOI 10.1145/2735960.2735992.

[7] Hasan, K.; Shetty, S.; Sokolowski, J.; & Tosh, D. K. Security Game for Cyber Physical Systems. Article 12. In *Proceedings of the Communications and Networking Symposium*. April 2018. https://dl.acm.org/citation.cfm?id=3213212

[8] Allodi, L. & Etalle, S. Towards Realistic Threat Modeling: Attack Commodification, Irrelevant Vulnerabilities, and Unrealistic Assumptions. Pages 23-26. In *Proceedings of the 2017 Workshop on Automated Decision Making for Active Cyber Defense*. Conference on Computer and Communications Security. November 2017. DOI 10.1145/3140368.3140372.

[9] Kreimel, P.; Eigner, O.; & Tavolato, P. Anomaly-Based Detection and Classification of Attacks in Cyber-Physical Systems. Article 40. In *Proceedings of the 12th International Conference on Availability, Reliability and Security.* September 2017. DOI 10.1145/3098954.3103155.

[10] Agadakos, I.; Chen, C.; Campanelli, M.; Anantharaman, P.; Hasan, M.; Copos, B.; Lepoint, T.; Locasto, M.; Ciocarlie, G. F.; & Lindqvist, U. Jumping the Air Gap: Modeling Cyber-Physical Attack Paths in the Internet-of-Things. Pages 37-48. In *Proceedings of the 2017 Workshop on Cyber-Physical Systems Security and PrivaCy*. Conference on Computer and Communications Security. November 2017. DOI 10.1145/3140241.3140252.

[11] Ding, J.; Atif, Y.; Andler, S. F.; Lindstrom, B.; & Jesufeld, M. CPS-based Threat Modeling for Critical Infrastructure Protection. *ACM SIGMETRICS Performance Evaluation Review*. Volume 45. Issue 2. September 2017. Pages 129-132. DOI 10.1145/3152042.3152080

[12] Wuyts, K.; Van Landuyt, D.; Hovsepyan, A.; & Joosen, W. Effective and efficient privacy threat modeling through domain refinements. Pages 1175-1178. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*. April 2018. DOI 10.1145/3167132.3167414.

[13] Sion, L.; Yskout, K.; Van Landuyt, D.; & Joosen, W. Solution-aware data flow diagrams for security threat modeling. Pages 1425-1432. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*. April 2018. DOI 10.1145/3167132.3167285.

[14] Shostack, A. *Threat Modeling: Designing for Security*. Wiley, 2014. ISBN 978-1118809990.

[15] Mead, N.; Shull, F.; Vemuru, K.; & Villadsen, O. *A Hybrid Threat Modeling Method*. CMU/SEI-2018-TN-002. Software Engineering Institute, Carnegie Mellon University. 2018. http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=516617

[16] Shull, F. *Evaluation of Threat Modeling Methodologies*. Software Engineering Institute, Carnegie Mellon University. 2016. http://resources.sei.cmu.edu/library/asset-view.cfm?assetID=474197

[17] Schneier, B. Attack Trees. *Dr. Dobb's Journal.* July 22, 2001. http://web.cs.du.edu/~ramki/papers/attackGraphs/SchneierAttackTrees.pdf

[18] Shostack, A. Threat Modeling Thursdays. July 23, 2018 [accessed]. https://adam.shostack.org/blog/category/threat-modeling/threat-model-thursdays/

[19] UcedaVélez, T. *Threat Modeling w/PASTA: Risk Centric Threat Modeling Case Studies*. Technical Report. Open Web Application Security Project (OWASP). 2017.

[20] Karahasanovic, A.; Kleberger, P.; & Almgren, M. Adapting Threat Modeling Methods for the Automotive Industry. In *Proceedings of the 15th ESCAR Conference.* 2017. https://research.chalmers.se/en/publication/502671

[21] Simeonova, S. Threat Modeling in the Enterprise, Part 2: Understanding the Process. *Security Intelligence.* August 15, 2016. https://securityintelligence.com/threat-modeling-in-the-enterprise-part-2-understanding-the-process/

[22] Beyst, B. Which Threat Modeling Method. *ThreatModeler.* April 15, 2016. https://threatmodeler.com/2016/04/15/threat-modeling-method/

[23] Cleland-Huang, J. How Well Do You Know Your Personae Non Gratae? *IEEE Software*. Volume 31. Number 4. July 2014. Pages 28–31. http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6834694

[24] Mead, N. & Shull, F. The Hybrid Threat Modeling Method [blog post]. *SEI Blog.* April 23, 2018. https://insights.sei.cmu.edu/sei_blog/2018/04/the-hybrid-threat-modeling-method.html

[25] Lawson, C. & Pratap, K. Market Guide for Security Threat Intelligence Products and Services. 2017. https://www.gartner.com/doc/3765965/market-guide-security-threat-intelligence

[26] Hernan, S.; Lambert, S.; Shostack, A.; & Ostwald, T. Uncover Security Design Flaws Using the STRIDE Approach. *MSDN Magazine.* November 2006.

[27] Howard, M. & Lipner, S. *The Security Development Lifecycle.* Microsoft Press. 2006.

[28] Scandariato, R.; Wuyts, K.; & Joosen, W. A descriptive study of Microsoft's threat modeling technique. *Requirements Engineering*. Volume 20. Number 2. June 2015. Pages 163–180. DOI 10.1007/s00766-013-0195-2

[29] Microsoft Corporation. SDL Threat Modeling Tool. *Security Development Lifecycle*. July 20, 2018 [accessed]. https://www.microsoft.com/en-us/sdl/adopt/threatmodeling.aspx

[30] Martins, G.; Bhatia, S.; Koutsoukos, X.; Stouffer, K.; Tang, C.; & Candell, R. Towards a systematic threat modeling approach for cyber-physical systems. Pages 1-6. In *Proceedings of the 2015 Resilience Week*. August 2015. DOI 10.1109/RWEEK.2015.7287428.

[31] UcedaVélez, T. *Real World Threat Modeling Using the PASTA Methodology*. Technical report. Open Web Application Security Project (OWASP). 2012. https://www.owasp.org/images/a/aa/AppSecEU2012_PASTA.pdf

[32] UcedaVélez, T. & Morana, M. M. Risk Centric Threat Modeling: Process for Attack Simulation and Threat Analysis. Wiley. 2015. ISBN 978-0-470-50096-5.

[33] Leblanc, D. DREADful [blog post]. *David LeBlanc's Web Log*. August 14, 2007. https://blogs.msdn.microsoft.com/david_leblanc/2007/08/14/dreadful/

[34] Downloads. *LINDDUN: Privacy Threat Modeling.* July 23, 2018 [accessed]. https://distrinet.cs.kuleuven.be/software/linddun/download.php#

[35] Deng, M.; Wuyts, K.; Scandariato, R.; Preneel, B.; & Joosen, W. A Privacy threat analysis framework: supporting the elicitation and fulfillment of privacy requirements. *Requirements Engineering.* Volume 16. Issue 1. March 2011. Pages 3-32. https://link.springer.com/article/10.1007/s00766-010-0115-7

[36**]** Alberts, C.; Dorofee, A.; Stevens, J; & Woody, C. *Introduction to the OCTAVE Approach*. Software Engineering Institute, Carnegie Mellon University. August 2003. https://resources.sei.cmu.edu/library/Asset-view.cfm?assetid=51546

[37] Common Vulnerability Scoring System v3.0: Specification Document. *Forum of Incident Response and Security Teams.* July 23, 2018 [accessed]. https://www.first.org/cvss/specification-document

[38] National Vulnerability Database. *National Institute of Standards and Technology.* July 23, 2018 [accessed]. https://nvd.nist.gov/vuln-metrics/cvss#

[39] Hanic, D. & Surkovic, A. *An Attack Model of Autonomous Systems of Systems* [Master's Thesis]. Mälardalen University. 2018. http://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1218262&dswid=-7458

[40] Khan, R.; McLaughlin, K.; Laverty, D.; & Sezer, Sakir. STRIDE-based Threat Modeling for Cyber-Physical Systems. In *Proceedings of the 2017 IEEE PES Innovative Smart Grid Technologies Conference Europe.* September 2017. DOI 10.1109/ISGTEurope.2017.8260283.

[41] Denning, T. A.; Friedman, B.; & Kohno, T. Home. *Security Cards: A security threat brainstorming toolkit*. 2013. http://securitycards.cs.washington.edu/index.html

[42] Saitta, P.; Larcom, B.; & Eddington M. Trike v.1 Methodology Document [Draft]. *OctoTrike.* July 13, 2005. http://www.octotrike.org/papers/Trike_v1_Methodology_Document-draft.pdf

[43] Stanganelli, J. Selecting a Threat Risk Model for Your Organization, Part Two. *eSecurity Planet.* September 27, 2016. https://www.esecurityplanet.com/network-security/selecting-a-threat-risk-model-for-your-organization-part-two.html

[44] Common Vulnerability Scoring System SIG. *Forum of Incident Response and Security Teams*. July 30, 2018 [accessed]. https://www.first.org/cvss/

[45] Mead, N.; Hough, E.; & Stehney, T., II. *Security Quality Requirements Engineering Technical Report*. CMU/SEI-2005-TR-009. Software Engineering Institute, Carnegie Mellon University. 2005. https://resources.sei.cmu.edu/library/asset-view.cfm?assetID=7657

Contact Us

Software Engineering Institute
4500 Fifth Avenue, Pittsburgh, PA 15213-2612

Phone: 412/268.5800 | 888.201.4479
Web: www.sei.cmu.edu
Email: info@sei.cmu.edu

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

DM18-0894