Threat modeling - A comparison between PASTA and STRIDE

Praneet Kala and Ivan Khudur

April 24, 2023

1 Introduction

The concept of Development Operations (DevOps) is being able to automate development, deployment and infrastructure monitoring and is a culture shift towards collaboration between development, quality assurance and operations [5]. This is the main idea of DevOps and makes it possible to rapidly deliver services and software. However, the positive effects may easily be disrupted unless one of the most important parts of development is taken into consideration: security [11]. The amount of cyber attacks are increasing every year, and this threat should not be taken lightly [1]. Failing to take measures when it comes to safety-critical systems have resulted in major financial losses, misuse of sensitive information and in worst case, loss of life [12].

By inserting the security aspect into DevOps the coined term DevSecOps. It is a mindset that integrates the security aspect into the start of the project and makes it a continuous part to think about throughout development [11]. Threat modeling is one security method that can be used to mitigate security risks early on in the software development process [4]. In this essay, we will go through 2 popular threat modeling frameworks and compare them: PASTA (**P**rocess for **A**ttack **S**imulation and **T**hreat **A**nalysis) and STRIDE (**S**poofing, **T**ampering, **R**epudiation, **I**nformation Disclosure, **D**enial of Service, **E**levation of Privilege)

2 Threat modeling methodologies: PASTA and STRIDE

Many, if not all, organisations these days want to ensure that they are safe against cyber attacks and security threats in their data, software, computers, networks, and cloud platforms. Architectural weaknesses can also be revealed if the threat modeling process is started earlier in the development life cycle should there be significant changes that need to be addressed [20]. On the downside, threat modeling is still a growing industry and methods can be time consuming and resource intensive so organisations sometimes avoid it, hence the need for more accessible tools and frameworks [22].

2.1 What is PASTA?

The threat model PASTA is a flexible framework that can be adopted to various scenarios. It was founded in 2015 by VerSprite, and can be used to identify threats regardless of the stage of the development process. It is a linear process that allows collaboration between different parties in the organization, and focuses on risks related to the business context [21]. Furthermore, it is risk-centric, i.e. the focus of PASTA is the security threats that are most relevant to the business and pose the highest risk or has the largest impact [14] [8].

The PASTA framework consists of 7 steps, as illustrated in Figure 1. Each step outputs information that the next step uses [21].

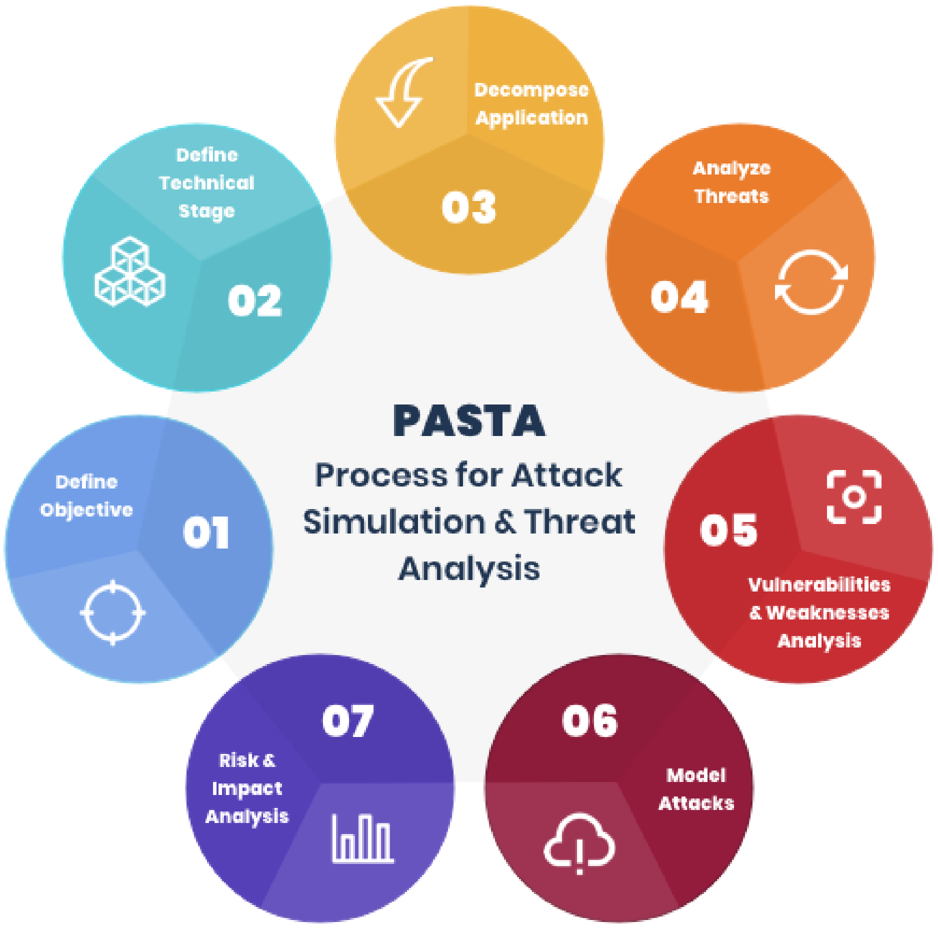

Figure 1: 7 steps of PASTA [23]

2.1.1 The 7 steps of PASTA

The steps can be summarized as follows:

1. **Define objectives** - Identify and define objectives of creating the software. This requires collection of various documents, such as specifying business- and functionality requirements, security policies, standards and guidelines [21].

2. **Define technical scope** - Define how the application should run, i.e. the technology that is used. What is used and how is it setup? What dependencies exists? Should anything be in the cloud, and what third party solutions will be used? [8]

3. **Decomposition of application** - Knowing what should be used, this step focuses on how everything is connected and communicates with each other. A data flow diagram is created, and any implicit trust models are mapped. [21] This is because implicit trust models may be exploited, as the model states that an user/resource in e.g. an internal network may access other resources without authentication, because the model trusts that the user/resource is who it says it is (considering it is in the system) [10] [15].

4. **Analyse threats** - This steps focuses on studying relevant threats to the application based on the gathered information, and build attack scenarios. This is done by

gathering threat information from various resources, in order to fully understand them and map actual threats (i.e. threats that are likely to occur) [8] [21].

5. **Analyse vulnerabilities** - Vulnerabilities in the code and design are analysed, in order to better understand potential security risks that will affect the objectives (that were defined in step 1 of PASTA). The severity of the vulnerabilities can be evaluated in this step using different standards, such as CVSS (Common Vulnerability Scoring System) [8] [21].

6. **Attack analysis** - Knowing what vulnerabilities exists and their severity, and knowing of potential threats, this steps maps them together to determine how likely the weaknesses are to be exploited. This is done using an attack tree of either the entire application, or some part of it, where vulnerabilities are mapped to the different nodes. The nodes of the attack tree corresponds to a component of the application (or part of the application) [21].

7. **Risk and impact analysis** - The final step is to combine all the information that has been gathered so far and create a risk profile. This is then used to create strategies to mitiage the risks [21].

2.2 What is STRIDE?

STRIDE is used to build a secure system and identifies security requirements by considering security aspects such as potential threats to target systems and services [7]. STRIDE is a mnemonic of 6 types of security threats and it can tell us what can go wrong in your software. Each letter of the word in STRIDE is related to a security property and can be associated with specific security attributes [7], as shown in Table 1:

Threat	Security property	Threat definition
Spoofing	Authentication	Impersonate something or someone else
Tampering	Integrity	Modify data or code
Repudiation	Non-repudiation	Claim to have not performed an action
Information disclosure	Confidentiality	Expose information to someone not authorized see it
Denial of service	Availability	Deny or degrade service to users
Elevation of privilege	Authorization	Gain capabilities without proper authorization

Table 1: Correlation between six threats in STRIDE and security properties [7]

If we take a look into each thread, it will allow us to create a good STRIDE model. Microsoft's STRIDE methodology aims to ensure that an application meets the security requirements of Confidentiality, Integrity, and Availability (CIA) [2]. It is a relatively lightweight approach and consists of five steps in it's methodology. There are several methods and approaches that can be used to apply the STRIDE methodology. Five high-level steps consisting of decomposing the system, plotting a Data Flow Diagram (DFD), analysing threats, identifying vulnerabilities and planning mitigation strategies have been proposed by Khan et al. in [6]. Another structured approach that has been presented by Open Web Application Security Project(OWASP) discusses three high level steps that consists of decomposing the application, determining and ranking threats, and finally, determining countermeasures and mitigation steps as described by [3]. In this essay, the second three-step process is discussed.

2.2.1 Decompose the Application

This step is involved with understanding the system and the components involved, excluding physical components. There needs to be an understanding of how the system interacts with external entities and then it can be decomposed into structural and logical components. Although it sounds obvious, it is worth mentioning that information gathering is important. Architectural diagrams in the form of use cases provide insights into the application with an understanding of key components. Requirements are necessary to list so that entry points, assets and trust levels can be identified to evaluate potential interaction points, items and or areas of interest that an attacker would be interested in. Lastly, project documentation to summarise and explain the application, mention of key players, and who is involved in terms of governance and team overview. This sets up the basis for the next step, which is creating a DFD [9].

2.2.2 Determine and Rank Threats

The DFD is an architectural representation that is used to visually showcase the flow of data between various key components involved in a project. If all the components were used to create a DFD, the model would become too complicated to analyse, hence sticking with the key components reduces the difficulty in understanding the system. In this step, one of the important drivers to make this work effectively is to make it an interactive task and engage with the teams involved in the project by receiving their input and cooperation. Non-technical team members need to get involved as well. A good example of how a DFD is constructed for a Hospital Information System (HIS) is show in Figure 2. Being able to capture the necessary elements such external entities, data flows, processes, and data stores in a lightweight manner is a good start among stakeholders [17].

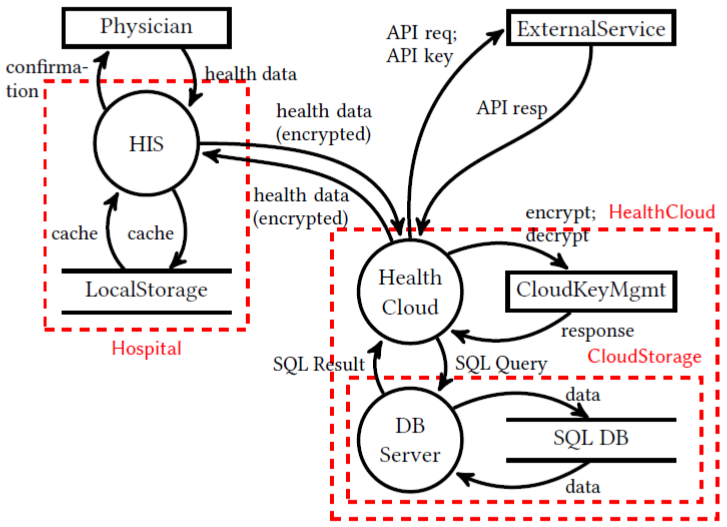

Figure 2: Data Flow Diagram of a Hospital Information System (HIS) [17]

Once a DFD has been created, the threats need to be categorised and in this step, STRIDE is applied in order to systematically identify threats and perform an analysis. This becomes an iterative process after all the possible threats have been drawn up and evaluated. An example of *information disclosure* would be if an attacker gains unauthorised access to read other users' messages through various forms of attacks or if the system could fail in specific ways. The attacker gains advantages of getting through if these vulnerabilities are not protected against. A good way to view and analyse specific situations is to use a threat tree diagram as show in Figure 3. A specific attack is broken down into smaller components describing it's vulnerabilities and the next layer describes how it can be safe-guarded.

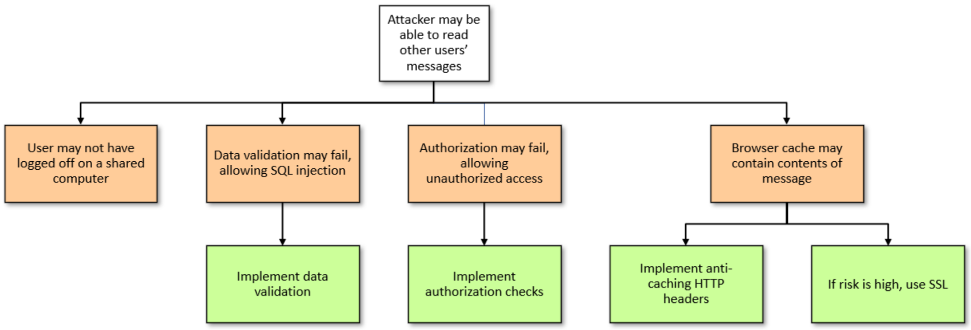
Attacker may be able to read other users' messages
User may not have logged off on a shared computer
Data validation may fail, allowing SQL injection
Authorization may fail, allowing unauthorized access
Browser cache may contain contents of message
Implement data validation
Implement authorization checks
Implement anti-caching HTTP headers
If risk is high, use SSL

Figure 3: Threat Tree Diagram [3]

Lastly, this steps requires the threats to be ranked according to highest priority and using risk factors. Creating this list and organising the threats in a prioritised manner sets up the foundation for the final step - creating a strategy for countermeasures.

2.2.3 Determine Countermeasures and Mitigation

The third and final step is defining and architecting countermeasures to mitigate potential threats that could occur in the system, as determined by the previous steps. In most cases, there are protective measures available (policies, security control, authentication and authorisation protocols) to prevent threats and once these are identified, they are linked to each of the threats that were ranked in step two. If there are not countermeasures for threats listed, they are considered vulnerabilities. After going through all the threats and identifying possible countermeasures, the final step is to categorise them into three different criteria (non-mitigated, partially mitigated and fully mitigated) and added to the documentation. If too much time spent on this process too early, it could take away precious time required to analyse and evaluate potential threats.

3 Comparing STRIDE and PASTA

STRIDE and PASTA are 2 popular threat modeling methodologies that we have explained in the previous section [16]. STRIDE comes from Microsoft and has been highly influenced by them, making it a more developer focused threat modeling process. PASTA on the other hand comes from a cyber security consulting organisation that have a focus from an attacker perspective [18]. PASTA also incorporates business impact analysis and the strategic objectives of an organisation when going through the process, and it is a collaborative process that involves IT departments in the process [18]. STRIDE has been around for a longer time than PASTA and has the benefit of being rather easy to use and understand. However, the simplicity of STRIDE also make it less reliable [21].

Following the methodology it is evident that STRIDE focuses on identifying vulnerabilities with regards to the 6 different threats that compose its name. It is therefore rather easy to miss out on other cyber security threats [13]. PASTA on the other hand is more encompassing when it comes to identifying threats. This is because part of the PASTA process is to build a threat library, in which relevant threats to the application are added as mentioned in section 2.1.4. STRIDE is very traditional and more categorical, and creates an association of risks, wheres PASTA considers relevant threats, simulates attacks on assets, and leverages other processes for security and can find vulnerabilities such as static code scanning, vulnerability management, risk management and maps threats and attacks [19]. Although this makes it more encompassing than STRIDE, it also requires a lot of effort.

Though there are differences between PASTA and STRIDE, both methodologies make use of decomposing the application/software that you use your threat model on. In both methodologies data flow diagrams are used to understand how data flows between different parts of the application and if there are any implicit trust model implemented that could be exploited [8].

4 Conclusion and Reflection

It is clear that both methodologies follow a step-by-step process to reach an end goal, which is to create a risk profile that is then used to formulate a strategy to mitigate risks. PASTA is a more elaborate methodology that involves a lot of different parties in the business and is able to provide a more comprehensive list of relevant threats. This makes it suitable to identify gaps in security that should be fixed, however due to its scope it becomes rather time consuming. STRIDE on the other hand is more focused on a developer-driven approach and can be implemented rather quick. The quickness comes however with a trade-off becomes apparent as STRIDE is limited in identifying relevant threats. That said, both methodologies fulfill a purpose: to identify threats and help make software more secure.

The research that was performed for threat modeling methodologies and processes is very relevant in DevSecOps. It is a vital component to consider for risk management in DevOps teams. Throughout the research of the 2 chosen methodologies we encountered a lot of other threat model methodologies. We chose STRIDE because it is old but still relevant, while PASTA is rather new and becoming ubiquitous across organizations and businesses. This makes PASTA relevant DevSecOps considering it built upon collaboration and you get strategic input from a variety of stakeholders.

To wrap it up here is a quote to chew on: *"If PASTA can't fix it, it's a serious problem"*.

References

[1] Moatsum Alawida et al. *A deeper look into cybersecurity issues in the wake of Covid-19: A survey*. Vol. 34. 10. 2022. DOI: 10.1016/j.jksuci.2022.08.003. URL: https://www.sciencedirect.com/science/article/pii/S1319157822002762.

[2] Archer Charles. *What is STRIDE Methodology in Threat Modelling?* May 2022. URL: https://www.koenig-solutions.com/blog/stride-methodology-in-threat-modelling.

[3] Larry Conklin, Victoria Drake, and Sven strittmatter. *Threat Modeling Process*. Sept. 2022. URL: https://owasp.org/www-community/Threat_Modeling_Process.

[4] Simone Curzi et al. *Integrating threat modeling with DevOps*. July 2022. URL: https://learn.microsoft.com/en-us/security/engineering/threat-modeling-with-dev-ops.

[5] Christof Ebert et al. "DevOps". In: *IEEE Software* 33.3 (2016). ISSN: 19374194. DOI: 10.1109/MS.2016.68. URL: https://ieeexplore.ieee.org/document/7458761.

[6] Rafiullah Khan et al. "STRIDE-based threat modeling for cyber-physical systems". In: *2017 IEEE PES Innovative Smart Grid Technologies Conference Europe, ISGT-Europe 2017 - Proceedings*. Vol. 2018-January. 2017. DOI: 10.1109/ISGTEurope.2017.8260283. URL: https://ieeexplore.ieee.org/abstract/document/8260283.

[7] Kyoung Ho Kim, Kyounggon Kim, and Huy Kang Kim. "STRIDE-based threat modeling and DREAD evaluation for the distributed control system in the oil refinery". In: *ETRI Journal* 44.6 (2022). ISSN: 22337326. DOI: 10.4218/etrij.2021-0181. URL: https://onlinelibrary.wiley.com/doi/10.4218/etrij.2021-0181.

[8] Nick Kirtley. *PASTA Threat Modeling*. July 2022. URL: https://threat-modeling.com/pasta-threat-modeling/.

[9] Nick Kirtley. *STRIDE Threat Modeling Example for Better Understanding and Learning*. Aug. 2022. URL: https://threat-modeling.com/stride-threat-modeling-example/.

[10] Biplob Paul and Muzaffar Rao. "Zero-Trust Model for Smart Manufacturing Industry". In: *Applied Sciences (Switzerland)* 13.1 (2023). ISSN: 20763417. DOI: 10.3390/app13010221. URL: https://www.mdpi.com/2076-3417/13/1/221.

[11] Redhat. *What is DevSecOps*. Mar. 2023. URL: https://www.redhat.com/en/topics/devops/what-is-devsecops.

[12] Mary Sánchez-Gordón and Ricardo Colomo-Palacios. "Security as Culture: A Systematic Literature Review of DevSecOps". In: *Proceedings - 2020 IEEE/ACM 42nd International Conference on Software Engineering Workshops, ICSEW 2020*. 2020. DOI: 10.1145/3387940.3392233. URL: https://dl.acm.org/doi/10.1145/3387940.3392233.

[13] Riccardo Scandariato, Kim Wuyts, and Wouter Joosen. "A descriptive study of Microsoft's threat modeling technique". In: *Requirements Engineering* 20.2 (2015). ISSN: 1432010X. DOI: 10.1007/s00766-013-0195-2. URL: https://link.springer.com/article/10.1007/s00766-013-0195-2.

[14] Robin Schulman and Joseph Longo. *Threat Modeling*. Dec. 2022. URL: https://about.gitlab.com/handbook/security/threat_modeling/.

[15] Malcolm Shore, Sherali Zeadally, and Astha Keshariya. *Zero Trust: The What, How, Why, and When*. 2021. DOI: 10.1109/MC.2021.3090018. URL: https://ieeexplore.ieee.org/document/9585170.

[16] Simplilearn. *What is Threat Modeling: Process and Methodologies*. 2020. URL: https://www.simplilearn.com/what-is-threat-modeling-article.

[17] Laurens Sion et al. "Security Threat Modeling: Are Data Flow Diagrams Enough?" In: *Proceedings - 2020 IEEE/ACM 42nd International Conference on Software Engineering Workshops, ICSEW 2020*. 2020. DOI: 10.1145/3387940.3392221. URL: https://dl.acm.org/doi/10.1145/3387940.3392221.

[18] ThreatModeler. *STRIDE, VAST, TRIKE, & More: Which threat modeling methodology is right for your organisation?* Aug. 2019. URL: https://threatmodeler.com/threat-modeling-methodologies-overview-for-your-business/.

[19] Toreon. *Threat Modeling can be considered as fun as cooking a good PASTA meal.* Aug. 2022. URL: https://www.toreon.com/tmi-newsletter-18-threat-modeling-can-be-considered-as-fun-as-cooking-a-good-pasta-meal-part-2/.

[20] Peter Torr. *Demystifying the threat-modeling process*. Vol. 3. 5. 2005. DOI: 10.1109/MSP.2005.119. URL: https://www.researchgate.net/publication/3437733_Demystifying_the_Threat-Modeling_Process.

[21] Tony UcedaVélez. *What is PASTA Threat Modeling?* Nov. 2021. URL: https://versprite.com/blog/what-is-pasta-threat-modeling/.

[22] Margus Välja et al. "Automating threat modeling using an ontology framework: Validated with data from critical infrastructures". In: *Cybersecurity* 3.1 (2020). ISSN: 25233246. DOI: 10.1186/s42400-020-00060-8. URL: https://link.springer.com/article/10.1186/s42400-020-00060-8.

[23] VerSprite. *The Process for Attack Simulation Threat Analysis*. 2021. URL: https://versprite.com/ebooks/leveraging-risk-centric-threat-models-for-integrated-risk-management/.