Threat modeling - A comparison between PASTA and STRIDE

Praneet Kala and Ivan Khudur

May 1, 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]. DevOps 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 may and 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, you get the coined term DevSecOps. It is a mindset that integrates the security aspect into the start of the project, making it a continuous part throughout development [11]. Threat modeling is one security method that can be used to mitigate security risks earlier in the 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 want to ensure they are safe against cyber-attacks and security threats. Architectural weaknesses can also be revealed if the threat modeling process is started earlier in development life cycles [15]. We will compare the similar steps between STRIDE and PASTA and how the different steps make them better or worse and the trade-off for choosing one over the other.

2.1 What is PASTA?

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 stakeholders, and focuses on risks related to the business context [16]. Furthermore, it is risk-centric, i.e. the focus lies on the security threats that are most relevant to the business and pose the highest risk or has the largest impact [8].

The PASTA framework consists of 7 steps, as illustrated in Figure 1. Each step outputs information that the next step uses [16].

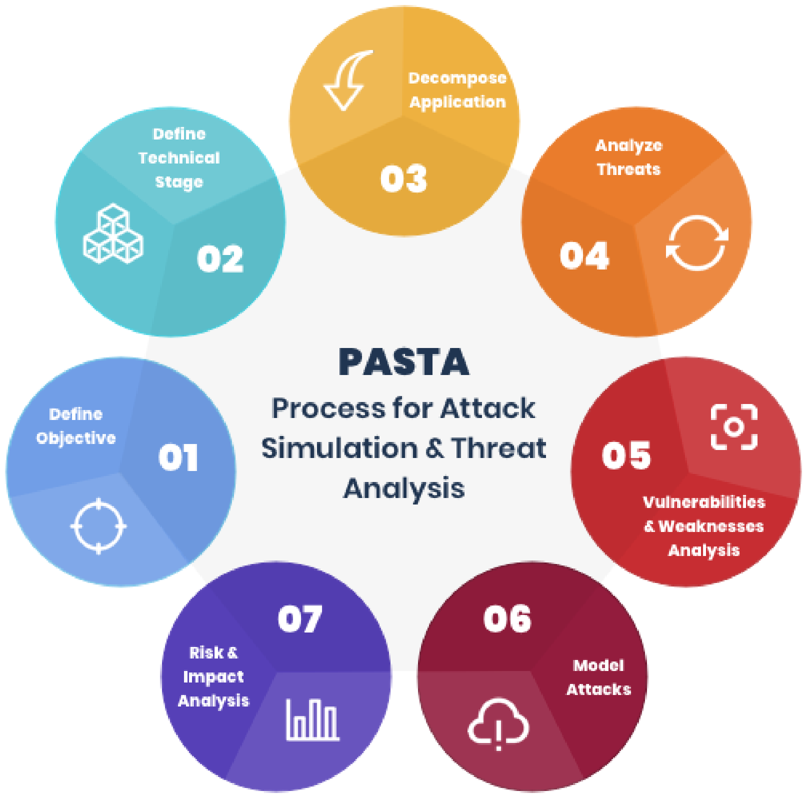


Figure 1: 7 steps of PASTA [17]

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 [16].

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** - Focuses on how everything is connected and communicates with each other. A data flow diagram (DFD) is created, and any implicit trust models are mapped [16]. This is because implicit trust models may be exploited, as written in [10].

4. **Analyse threats** - This step focuses on studying relevant threats to the application based on the gathered information, and build attack scenarios. It is done by gathering threat information from various resources, to fully understand them and map actual threats (i.e. threats that are likely to occur) [8] [16].

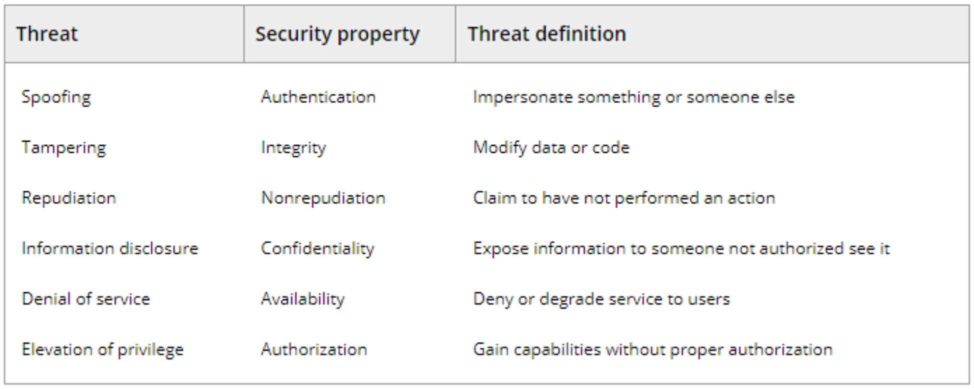
5. **Analyse vulnerabilities** - Vulnerabilities in the code and design are analysed, to better understand potential security risks that will affect the objectives. The severity of the vulnerabilities can be evaluated in this step using different standards, such as CVSS (Common Vulnerability Scoring System) [8] [16].

6. **Attack analysis** - The vulnerabilities and threats that have been documented are mapped 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) [16].

7. **Risk and impact analysis** - The final step is to combine all the information that has been gathered so far to create a risk profile. This is then used to create strategies to mitigate the risks [16].

2.2 What is STRIDE?

Microsoft's 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]. It aims to ensure that an application meets the security requirements of Confidentiality, Integrity, and Availability (CIA) [2]. Each letter of the word is related to a security property and can be associated with specific security attributes [7], as shown in Figure 2:



Threat	Security property	Threat definition
Spoofing	Authentication	Impersonate something or someone else
Tampering	Integrity	Modify data or code
Repudiation	Nonrepudiation	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

Figure 2: Correlation between STRIDE and security properties [7]

The structured approach that we will focus on in this essay has has been presented by Open Web Application Security Project (OWASP) and discusses three high-level steps as described by [3]. Other sources propose 5 steps as described by [6].

2.2.1 The 3 steps of STRIDE

The steps can be summarized as follows:

1. **Decompose the application** - Understanding the system and the components involved (excluding physical components), and how it is all connected and communicates with each other, by gathering information and putting together documentation [3]. This makes it possible to evaluate different items and create a DFD, which is a visual representation of the flow of data between various key components. An example of a DFD for a Hospital Information System (HIS) is shown in Figure 3.

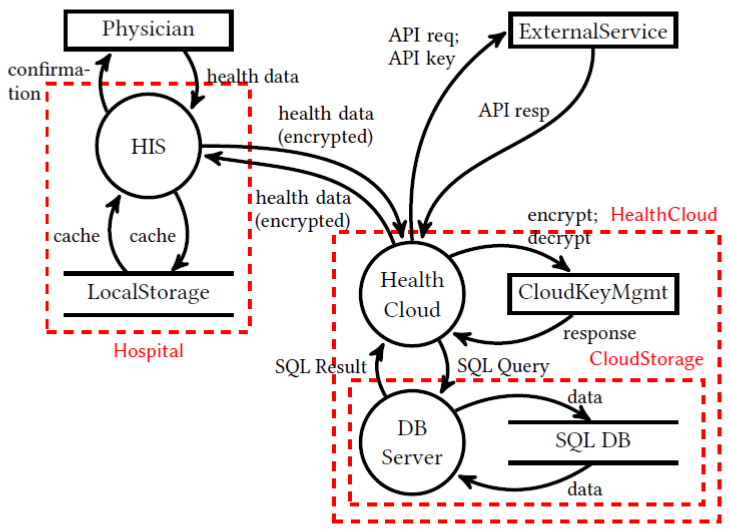


Figure 3: DFD of a HIS [14]

2. **Determine and Rank Threats -** Categorising threats. STRIDE is applied to systematically identify threats. This is an iterative process after all the possible threats have been drawn up and evaluated. A good way to view and analyse specific situations, for example *information disclosure*, is to use a threat tree diagram as show in Figure 4.

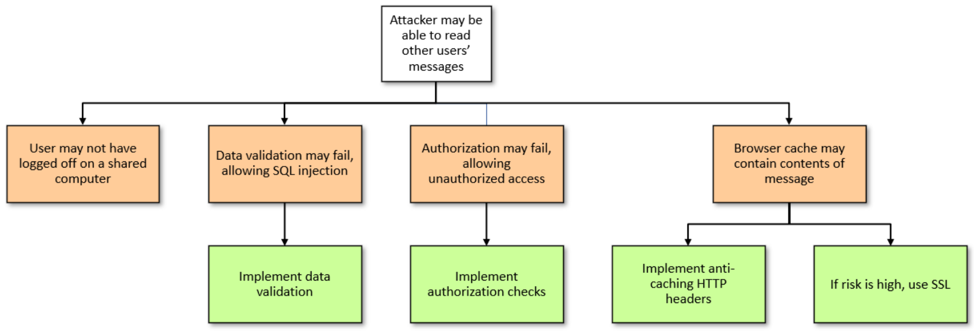

Attacker may be able to read other users' messages
User may not have logged off on a shared computer
Data validation may fail, allowing SQL injection
Authorization may fail, allowing unauthorized access
Browser cache may contain contents of message
Implement data validation
Implement authorization checks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

Figure 4: Threat Tree Diagram [3]

The threats are then ranked according to highest priority and using risk factors for creating a strategy for countermeasures.

3. **Determine Countermeasures and Mitigation -** There are protective measures available (policies, security controls, authentication, authorisation protocols) to prevent threats and once these are identified, they are linked to each of the threats. If there are no countermeasures for threats listed, they are considered vulnerabilities. The final list is then categorised using three criteria: non-mitigated threats, partially mitigated threats and fully mitigated threats [3].

3 Comparing STRIDE and PASTA

The methodologies we have talked about approach threat modeling in different ways, but work towards the same end goal. We are going to compare the overall structure of these 2 different methodologies, by focusing on these areas:

- Understanding the application

- Performing a threat analysis
- Creating a risk mitigation profile

3.1 Understanding the application

Both of the methodologies decomposes the application that you do the threat modeling on. These steps are however performed at different intervals, because there are more steps in PASTA. PASTA is more comprehensive and allowing an organisation to incorporate business aspects by defining objectives and technical scope before starting the decomposition of the application. Advantages of this include governance and compliance into the threat model. It is also a more collaborative effort and involves different teams, which provides better insight to the application and makes teams more aligned [8]. Depending on the business needs, this makes the PASTA approach more prepared when starting the decomposition step, compared to following STRIDE. The decomposition step is where the steps in PASTA and STRIDE correlate. This is where both methodologies focus on identifying all assets, users, data flows etc and create an architectural DFD and implicit trust models.

Although the PASTA approach is thorough, with all the steps leading up to the decomposition part, the thoroughness might also become an obstacle. This is because it is quite easy to get off track when many teams/persons with different backgrounds are involved and try to cover everything from what the business objectives are, to producing an architectural view of the application. Everyone in the project team will probably be able to contribute with something, but chances are they would need to understand other parts. Time could therefore be spent on less valuable things that do not contribute to the threat modelling, such as going into why the objectives are ”this but not that”. This is probably where STRIDE is a little stronger, as the focus is on the application and thus requiring less time to complete the decomposition.

3.2 Performing a threat analysis

Analysing threats are done in both methodologies, but how it is performed differs between them. STRIDE focuses on identifying vulnerabilities with regards to the 6 different threat categories that compose its name. It is a more traditional process that defines threat categories from an attacker and defensive perspective [3]. In STRIDE, all threats, vulnerabilities and attacks are related to each of the categories and they are generally quickly identified as high-level threats. PASTA on the other hand identifies threats, vulnerabilities and attacks that will affect the objectives of creating the application and are not connected to any particular categories [16].

Another distinct difference between STRIDE and PASTA is how and when the threat analysis is performed. STRIDE has no common practice to perform a threat analysis, however, there are various approaches used such as STRIDE-per-element or STRIDE-per-interaction. The former approach does not work well for interactions between components: it is more detailed and time consuming. The latter approach is ideal for cyber security threats and is less exhaustive [9]. PASTA takes a different approach, by creating a library of the most relevant threats based on e.g. threat intelligence reports and security incident reports. The most relevant threats are chosen by considering the impact it has on the business [16]. This is a time-consuming process that adds cost and complexity, which is an important trade-off that an organisation needs to make. However it ensures that they are less susceptible to missing out on potential risks.

3.3 Creating a risk mitigation profile

The final step in the threat modeling process is to bring together and finalise all the information gathered from the prior steps. STRIDE has a very high level and simplified threat profile based on three criteria as mentions in section 2.2.1. PASTA's risk-centric and business driven approach, creates a more informative report that provides a qualitative and quantitative business impact analysis. There are also residual benefits with the PASTA approach that might not occur in STRIDE. The risk mitigation strategy of PASTA could include e.g. changes to a vulnerable, non-technical component that is used by other systems. That would mean that the other systems would also become more secure, even though they were not considered in the PASTA process [13]. With STRIDE it would be difficult finding something similar, as the focus is more on the application itself.

4 Conclusion and reflection

It is well known that threat modeling is an effective method that is used to ensure security in applications. In this essay we have compared two popular threat modeling methodologies. PASTA is a more elaborate and time-consuming methodology that involves many different parties in the business and is able to provide a 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, thus presenting a trade-off; it is limited in identifying relevant threats.

To reflect on the two methodologies, it is clear that PASTA is well-structured and has predefined steps. A thing that we noticed throughout the research of PASTA was that many of the sources talk about the same 7 steps. It sets up a good foundation for performing threat modeling and there is a golden thread that is followed throughout the process. We believe this is one of the attractions of PASTA, as the flow does not "leave any stones unturned". STRIDE on the other hand is an open-ended methodology and does not have predefined steps like PASTA. It is up to the organisation or individual to decide how the process can be broken down into smaller steps to suit the application that needs security.

In conclusion, when choosing a threat methodology you have to make sure you know what you want to achieve and how fast you would want to do it. Regardless which one is chosen, doing threat modeling is better than not doing it.

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] Alissa Knight. *Threat Modeling of Connected Cars using STRIDE*. Sept. 2018. URL: https://alissaknight.medium.com/threat-modeling-of-connected-cars-using-stride-e8184764eb0a.

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

[14] 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.

[15] 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.

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

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