

# LECTURE 5: THREAT MODELING

# INTRODUCTION

A Threat Model in software security is a structured approach used to identify, evaluate, and address potential threats that could harm a software system.

The purpose of threat modeling is to understand the security risks to a system, prioritize those risks, and develop strategies to mitigate them. This process is very important in building secure software, as it helps in anticipating and countering potential attacks before they occur.

Threat modeling is the process of systematically identifying security threats and vulnerabilities, assessing their potential impact, and planning mitigations to protect the system.

The primary goal is to improve the security posture of a system by proactively identifying and addressing potential threats, ensuring that security is integrated throughout the software development lifecycle.

# COMPONENTS OF THREAT MODELING

## Key Components of Threat Modeling:

- **Assets:** The valuable components of the system that need protection, such as data.
- **Threats:** Potential actions or events that could compromise the confidentiality, integrity, or availability of assets.
- **Vulnerabilities:** Weaknesses or flaws in the system that could be exploited by a threat to cause harm.
- **Attack Vectors:** The paths or methods that attackers use to exploit vulnerabilities and carry out threats.
- **Mitigations:** The security controls or countermeasures implemented to reduce or eliminate the risks posed by threats.

In software security, mitigations are techniques and practices used to reduce or eliminate the risk of vulnerabilities and attacks.

# BUG VERSUS VULNERABILITY

In software security, "bug" and "vulnerability" are terms that refer to different aspects of software flaws. While they are related, they have distinct meanings and implications.

## 1. Bug

- **Definition:** A bug is a flaw, mistake, or unintended behavior in the software's code that causes the software to operate incorrectly or produce unexpected results. Bugs can arise from errors in logic, incorrect assumptions, or programming mistakes.
- **Scope:** Bugs can affect the functionality, performance, or usability of software. They are not necessarily security-related and may not have any impact on the security of the system.
- **Examples:**
  - o A calculation error in a financial application that leads to incorrect results.
  - o A typo in a user interface string that displays incorrect information to the user.
  - o A crash in a software program due to improper memory management.

# BUG VERSUS VULNERABILITY

## 2. Vulnerability

- **Definition:** A vulnerability is a specific type of bug or flaw in software that can be exploited by an attacker to compromise the security of the system. Vulnerabilities create opportunities for unauthorized access, data breaches, or other malicious activities.
- **Scope:** Vulnerabilities directly impact the confidentiality, integrity, or availability of a system. They can be exploited to perform actions that should not be possible, such as gaining unauthorized access, executing arbitrary code, or causing a denial of service.
- **Examples:**
  - o A buffer overflow vulnerability that allows an attacker to execute arbitrary code on a system.
  - o An SQL injection vulnerability that enables an attacker to manipulate a database.
  - o An authentication bypass vulnerability that lets an attacker access a system without proper credentials

# ATTACK VECTOR

An attack vector, or threat vector, is a way for attackers to enter a network or system. Common attack vectors include social engineering attacks, credential theft, vulnerability exploits, and insufficient protection against insider threats.

Suppose a security firm is tasked with guarding a rare painting that hangs in a museum. There are several ways that a thief could enter and exit the museum — front doors, back doors, elevators, and windows. A thief could enter the museum in some other way too, perhaps by posing as a member of the museum's staff. All of these methods represent attack vectors, and the security firm may try to eliminate them by placing security guards at all doors, putting locks on windows, and regularly screening museum staff to confirm their identity.

Similarly, digital systems all have areas attackers can use as entry points. Because modern computing systems and application environments are so complex, closing off all attack vectors is typically not possible. But strong security practices and safeguards can eliminate most attack vectors, making it far more difficult for attackers to find and use them.

# THREAT MODELING METHODOLOGIES AND TECHNIQUES

When performing threat modeling, there are multiple methodologies security teams can use. The right model for the organization’s needs depends on what types of threats they are trying to model and for what purpose.

## 1. STRIDE threat modeling

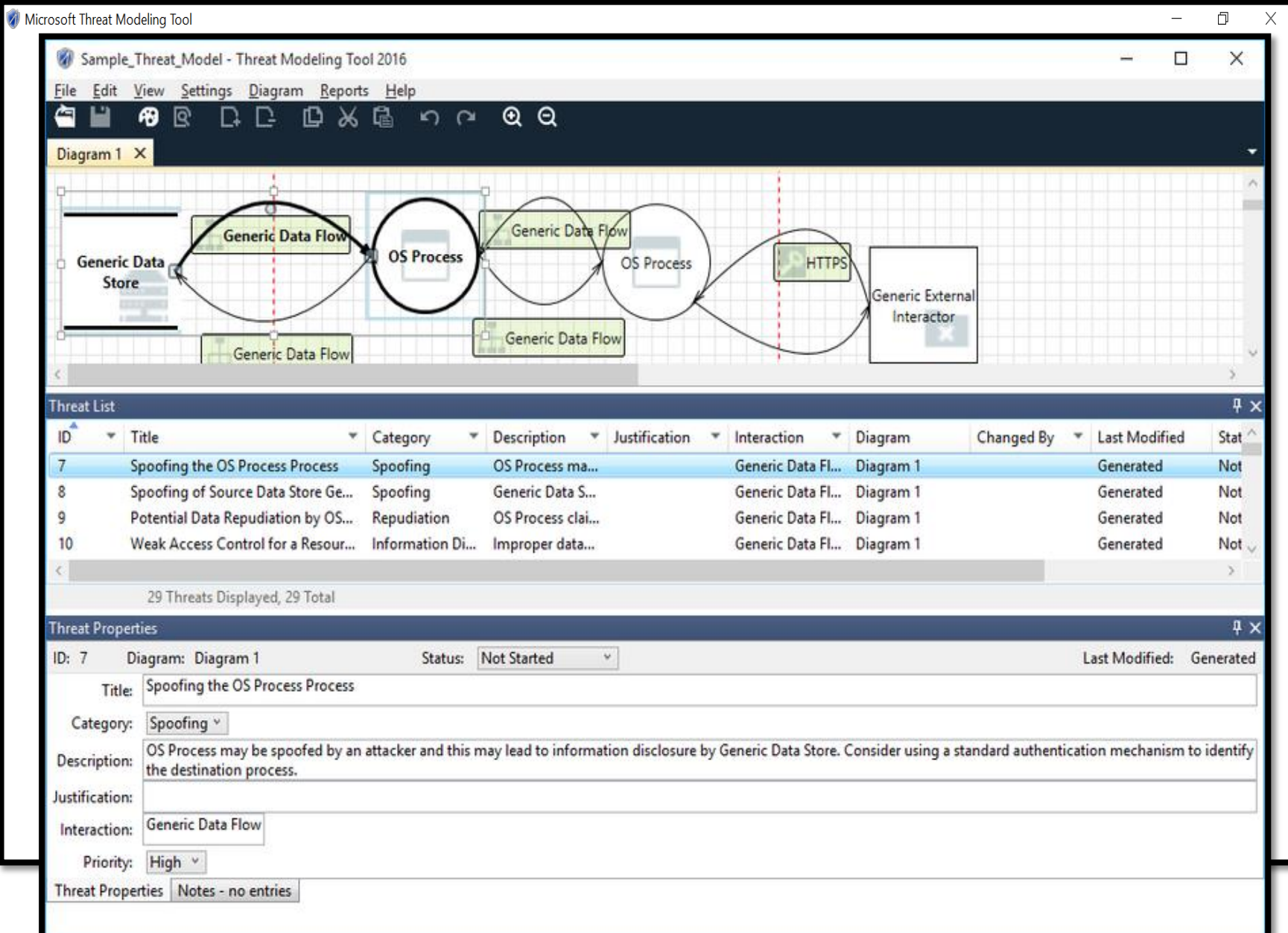
STRIDE is a threat model, created by Microsoft engineers, which is meant to guide the discovery of threats in a system. It is used along with a model of the target system. This makes it most effective for evaluating individual systems.

STRIDE is an acronym for the types of threats it covers, which are:

	Type of Threat	What Was Violated	How Was It Violated?
S	Spoofing	Authentication	Impersonating something or someone known and trusted.
T	Tampering	Integrity	Modifying data on disk, memory, network, etc.,
R	Repudiation	Non-repudiation	Claim to not be responsible for an action
I	Information Disclosure	Confidentiality	Providing information to someone who is not authorized
D	Denial of Service (DoS)	Availability	Denying or obstructing access to resources required to provide service
E	Elevation of Privilege	Authorization	Allowing access to someone without proper authorization

# THREAT MODELING METHODOLOGIES AND TECHNIQUES

. **Microsoft Threat Modeling Tool (MSTMT)** is a free tool created by Microsoft to help developers and architects identify and mitigate potential security threats in their applications early in the development process.



The screenshot displays the Microsoft Threat Modeling Tool (MSTMT) interface. The top window, titled "Sample\_Threat\_Model - Threat Modeling Tool 2016", shows a diagram of a threat model. The diagram includes a "Generic Data Store" connected to an "OS Process" via a "Generic Data Flow". The "OS Process" is further connected to another "OS Process" and a "Generic External Interactor" via "Generic Data Flow" and "HTTPS" respectively. Below the diagram, the "Threat List" pane displays a table of generated threats.

ID	Title	Category	Description	Justification	Interaction	Diagram	Changed By	Last Modified	Status
7	Spoofing the OS Process Process	Spoofing	OS Process ma...		Generic Data Fl...	Diagram 1		Generated	Not
8	Spoofing of Source Data Store Ge...	Spoofing	Generic Data S...		Generic Data Fl...	Diagram 1		Generated	Not
9	Potential Data Repudiation by OS...	Repudiation	OS Process clai...		Generic Data Fl...	Diagram 1		Generated	Not
10	Weak Access Control for a Resour...	Information Di...	Improper data...		Generic Data Fl...	Diagram 1		Generated	Not

29 Threats Displayed, 29 Total

The "Threat Properties" pane at the bottom shows details for the selected threat (ID: 7):

- ID: 7
- Diagram: Diagram 1
- Status: Not Started
- Last Modified: Generated
- Title: Spoofing the OS Process Process
- Category: Spoofing
- Description: OS Process may be spoofed by an attacker and this may lead to information disclosure by Generic Data Store. Consider using a standard authentication mechanism to identify the destination process.
- Justification:
- Interaction: Generic Data Flow
- Priority: High

Threat Properties | Notes - no entries

# THREAT MODELING METHODOLOGIES AND TECHNIQUES

## 2. Process for Attack Simulation and Threat Analysis (PASTA)

PASTA is an attacker-centric methodology with seven steps. It is designed to correlate business objectives with technical requirements. PASTA's steps guide teams to dynamically identify, count, and prioritize threats.

The steps of a PASTA threat model are:

1. Define business objectives
2. Define the technical scope of assets and components
3. Application decomposition and identify application controls
4. Threat analysis based on threat intelligence
5. Vulnerability detection
6. Attack enumeration and modeling
7. Risk analysis and development of countermeasures

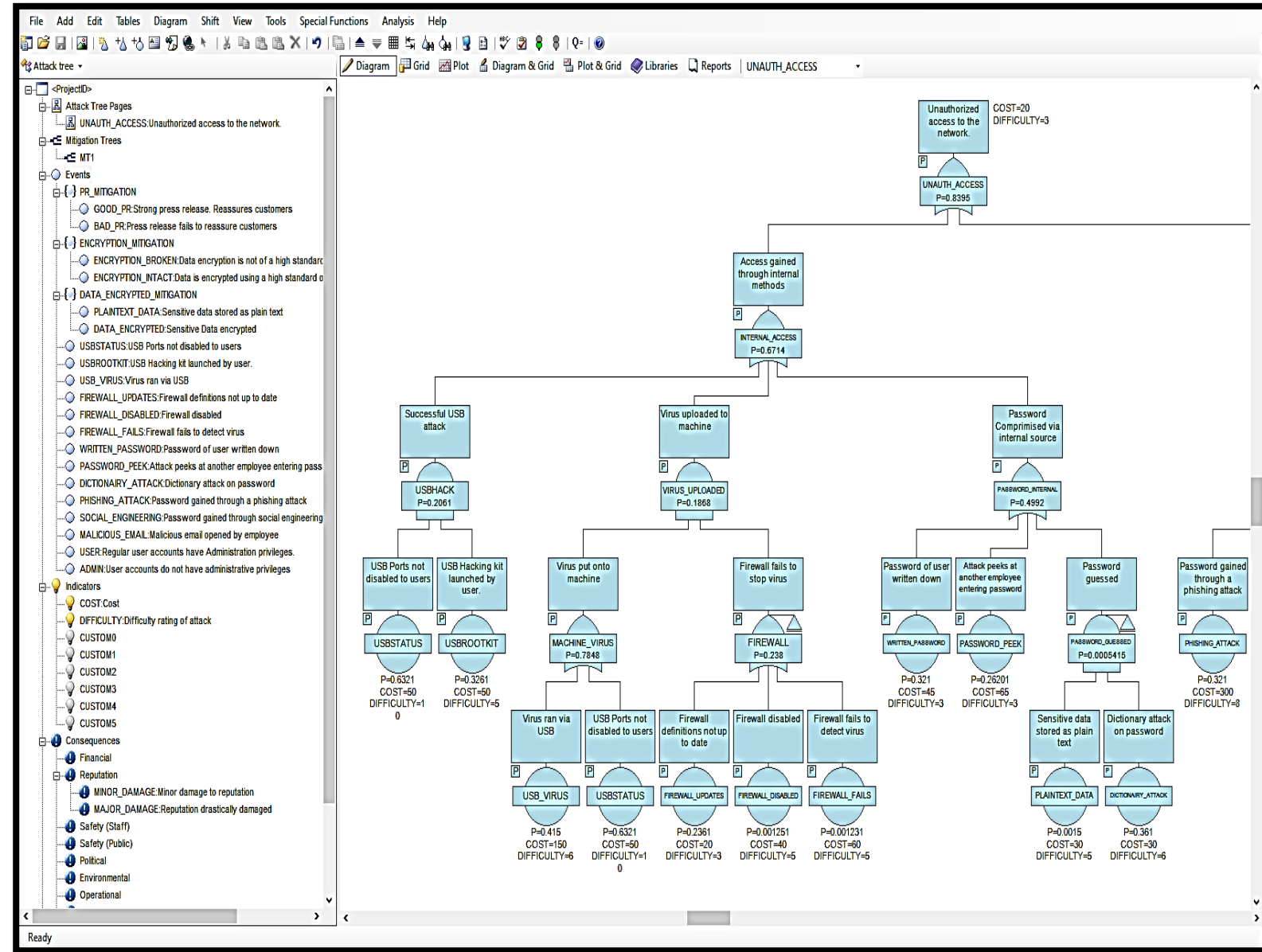


# THREAT MODELING METHODOLOGIES AND TECHNIQUES

## 3. Attack Trees

Attack trees are one of the oldest and most widely used threat modeling techniques.

it is charts that display the paths that attacks can take in a system. These charts display attack goals as a root with possible paths as branches. When creating trees for threat modeling, multiple trees are created for a single system, one for each attacker goal. While once used alone, it is now frequently combined with other methodologies, including PASTA, and STRIDE.



Attack tree tool