STRIDE

Welcome back. In this video, we'll look at exploring threat modelling through the STRIDE framework. In developing computer systems the security is often left as an afterthought. It can be difficult for developers to try and consider the types of attacks which might happen.

As a result, STRIDE was developed to help developers consider common types of attack against a system. STRIDE makes use of an acronym to represent common threats against a system. S stands for spoofing, T stands for tampering, R for repudiation, I is for information disclosure, D is for denial of service, and E is for elevation or escalation of privilege.

We will examine each of these in turn. We will consider the threat, the property it relates to, such as confidentiality, integrity, availability. Provide a definition, and look at an example of such a threat. First up is spoofing. If we look at how STRIDE is structured it's a fairly simple structure. It represents a range of common threats, one for each letter of STRIDE. These are Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, and Escalation of privileges.

We will address each of these in turn. First up is spoofing. This relates to the security property of authentication and how an individual or entity may masquerade as someone or something else. A good example of this is phishing emails and also phishing websites. Moving on. Next up we have tampering. This relates to the property of data integrity.

Tampering can be defined as unauthorised modification of data. An example of this could be unauthorised modification of a salary within an HR database. In computer security terms, repudiation is the rejection of responsibility for an action. Non-repudiation is a security property which provides assurance that an individual cannot deny an action.

For example, providing assurance of authenticity through a digital signature. If we didn't do something about repudiation, this would mean a limited ability to demonstrate an individual has completed an action to some degree of confidence, which would mean a lack of confidence in the transactions and actions between individuals or entities within the computer system. One example could be accessing an inappropriate website when one is not logged

into a computer.

It would be difficult to identify with any certainty that that individual who normally uses a computer was using it at that time. Another example might be claiming that an email from an individual's address was not sent by them. Next up we have information disclosure.

This relates specifically to confidentiality of information and the disclosure of information to parties who should not have the authorization to see that information. One example of this is password leaks, which happens regularly and is often represented in the news. Turning now to denial of service.

This relates to the availability. In particular, services are unavailable to legitimate users at a time when it should be available. An example of this is through service request floods, such as HTTPS. And finally we have escalation of privilege. This relates to authorization and an individual achieving increased privileges without appropriate authorization. An example of this might be a user with read only permissions for a document being able to write to that document. Having now considered each of these elements in turn, we can look at how we might apply this framework.

As mentioned, this framework is structured to help assist developers to consider common threats so it can be limited. It tends to be a proactive consideration of potential threats rather than analysis of suspected attacks. In considering the STRIDE model, each element should be considered in turn.

How can that be applied to the system component or application under consideration? For example, how could a threat actor spoof this component of the system or the program? As you progress through this, you should record details of each threat as you progress, noting any assumptions. For example, assuming that the attacker is able to perform reconnaissance, which would allow them to identify a web application which is used internally, could then potentially spoof by trying common username and password combinations within that application. As you can see, this can be helpful in identifying mitigation techniques to implement within a system, as it helps developers identify any potential common threats.

Within the previous example, this might be implementing a password policy and providing training to staff to help them deploy appropriate passwords. In general, the mitigation approaches for each potential threat can be considered as follows. For spoofing, ensure appropriate authentication.

For tampering, ensure data protection and integrity. Repudiation, ensure appropriate non-repudiation mechanisms. Information disclosure, ensure confidentiality, such as through the use of encryption or cryptographic hashes, or indeed air-gapped machines which aren't

connected to the network. For denial of service, ensure availability through implementation techniques such as firewalls, intrusion detection and prevention systems to prevent such an attack happening.

For privilege escalation, ensure appropriate authorization mechanisms. Hopefully you can see how this process can assist developers and others identify any potential threats when constructing a system or exploring processes. STRIDE is a framework to help consider possible threats, but it is worth noting that this may not be comprehensive.

It tends to be more of a proactive framework to help individuals identify possible threats. If you're exploring a set of security incidents within a larger campaign from attackers, for example, then it may result in ignoring important components such as novel attacks like day-to-day vulnerabilities.

That's it for this video where we've examined the STRIDE framework and how that can help software developers build security into their systems. I hope you enjoyed the video, and I'll see you next time.