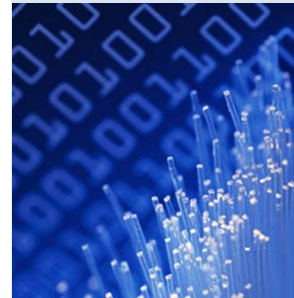




The Importance of Threat Modelling

An IRM Research White Paper by
Varun Uppal



IRM Research

Information technology constantly changes and advances. IRM is dedicated to keeping pace with new technology and continuing to innovate in the field of information security. This ensures that we are well informed of new issues and technologies, expanding our knowledge and providing world class services to our clients.

The Importance of Threat Modelling

Threat modelling is a strategic way of building security into applications by enumerating threats relevant to the solution. Also known as application risk assessment, it helps application designers to think in both an offensive and defensive manner, thereby allowing timely identification of potential threats, vulnerabilities, attacks and corresponding countermeasures. Development and testing activities become more effective once threat modelling processes have been adopted as it allows for easier prioritisation of application components and threats to test for.

Advantages of Threat Modelling

Some of the main advantages of threat modelling are as follows:

Heals Question Design Level Security: Ideally security processes should be integrated at the application design phase. This helps solution architects identify potential problem areas at an early phase, against which the development team can construct adequate countermeasures.

Helps in Formulating Security Tests: As will be detailed later, one of the most significant advantages of adopting threat modelling is that it streamlines security testing and makes it more effective in identifying vulnerabilities. Depending on threats identified, security test cases can be derived which are further prioritised based on the component to be tested. Since this allows easy mapping of tests executed to each of the threats identified, it provides maximum test coverage ensuring fewer issues are missed as a part of security testing, which is one of the shortcomings of less structured security testing.

Repeatable Test Model Aids in Identifying Security Implication of System Changes: The deliverable of threat modelling is a repeatable model that provides a holistic view of system security. One of the biggest advantages is that it helps in creating a repository of security tests indigenous to each application. The test repository can be used in the future to mount attacks for each system change that occurs. Any system changes that take place can be quickly represented in the final output which in turn would reflect any security implications the change may have. Finally, the set of security test cases can be upgraded to include new tests which would validate any issues which may have arisen as a result of the change.

Reduction of Ongoing Software Support Costs: It is a known fact that patching applications in production has cost implication and can often result in significant financial losses and downtime in situations where the patch results in functional or compatibility issues. If adopted at an early phase in the SDLC (Software Development Lifecycle), threat modelling helps in cutting down production patch times by proactive identification of vulnerabilities during design and development. By the time the application goes into production, security defects are considerably reduced, thereby reducing the requirement for future patches.

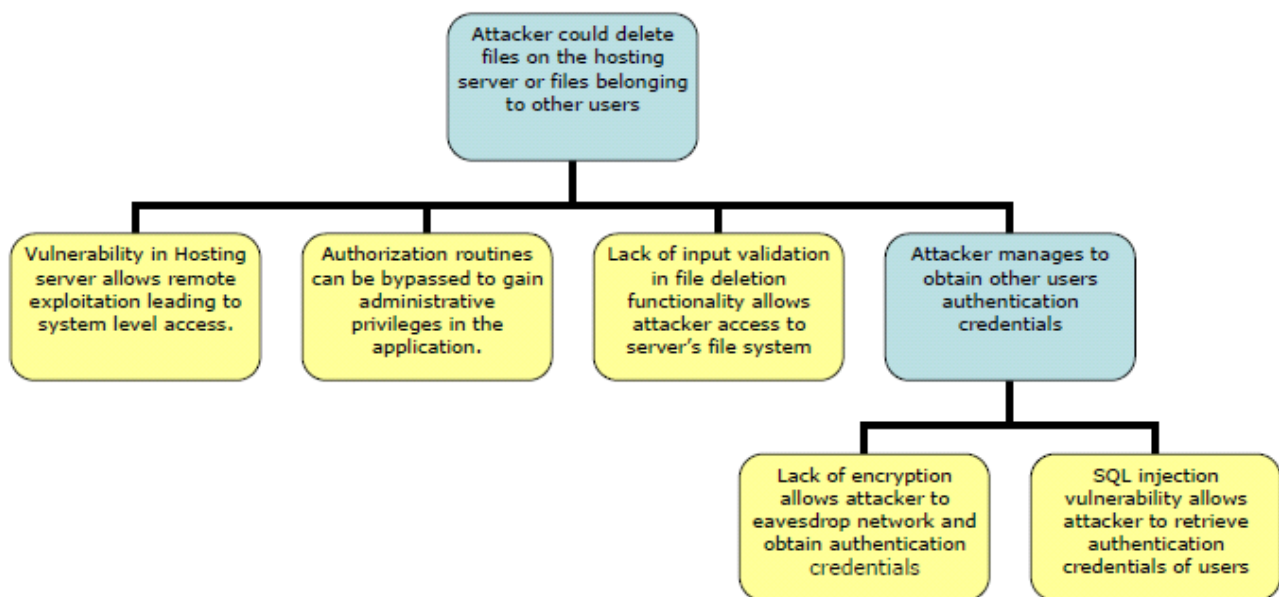
Creating a Threat Model

In order to create an accurate threat model, there are a series of stages that must be completed; these stages are detailed below:

Objective and Application Walkthrough: The first step to developing a threat model is to acquire as much information as possible about the target application. This includes application architecture, roles and access privileges. It should also capture data sources which will be utilised by the solution. Application walkthrough sessions should be performed where the development or design team provides an overview of what the application is intended to do. This phase should ensure that the security objectives of the application have been adequately captured i.e. it is well understood what elements the application needs to defend.

Application Decomposition: This phase lists various components which constitute the entire application, their interactions with users and other components and the data they process. Components could be data sources such as database servers, authentication and authorisation modules etc. Decomposing the application makes it easier to identify threats which may arise due to specific application logic or technology being employed in that component, thereby allowing designers and developers to take a decision regarding the layer at which countermeasures need to be integrated. For example, a query processing component which handles user supplied data can be vulnerable to injection attacks. The application designers can rely on a threat model to identify all instances where features of this component would be invoked and correspondingly take a decision to implement countermeasures, in the form of input validation, at component level or in the form of a centralised scheme.

Threat and Vulnerability Identification: This step involves listing all possible known threats applicable to the application being reviewed. Threats can be derived from the application's functionality and business logic, the development platform used or supporting infrastructure and data which the application processes and wishes to protect. One of the most widely adopted models for representing threats visually is "Threat Trees". Threat trees compromise threats as the parent nodes with child nodes being vulnerabilities required for those threats to actually manifest. A list of comprehensive threat trees also make it easy to derive security test cases corresponding to each vulnerability (i.e. child node). This can be made clearer by the example of an application which implements multiple roles/access privileges and allows users to upload and delete files. It consists of an admin role which has the privilege to delete any user file and also schedule system level commands. A sample threat tree is shown below, where the threat nodes are in blue and the vulnerabilities in yellow.

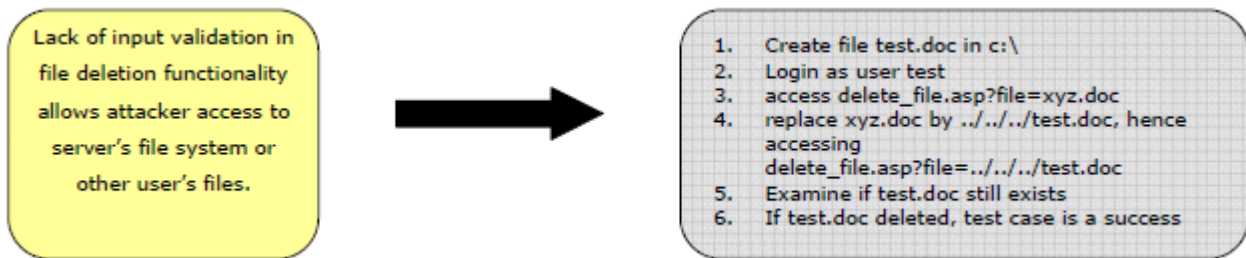


Notice how one threat can lead to another threat scenarios. In this case, the attacker's ability to obtain authentication credentials can also lead to the attacker being able to delete files belonging to other users. This is just a simple example and therefore, the threat tree presented here is not comprehensive. Further vulnerabilities can be added to this threat description depending on the extent of information available about the application.

Prioritising Threats and Vulnerabilities: Threat prioritisation is instrumental in developing an effective threat management strategy. Although a threat modelling exercise may yield several points of interest, not all of them would require the same degree of attention. Hence the implementation of countermeasures and test case development requires the prioritisation of issues identified so that the most critical findings can be resolved first. There are numerous ways of prioritising threats and vulnerabilities; however, the Microsoft

DREAD (Damage, Reproducibility, Exploitability, Affected Users, and Discoverability) model happens to be a popular one.

Developing Countermeasures and Security Test cases: Depending on the phase of SDLC in which threat modelling is being performed, the information built from previous phases can be used to design and implement countermeasures or develop security test cases if the objective of threat modelling is to drive security testing. As mentioned earlier, test cases are developed corresponding to each vulnerability in the threat tree. Considering one of the vulnerabilities identified in the example, a test case can be developed as shown below:

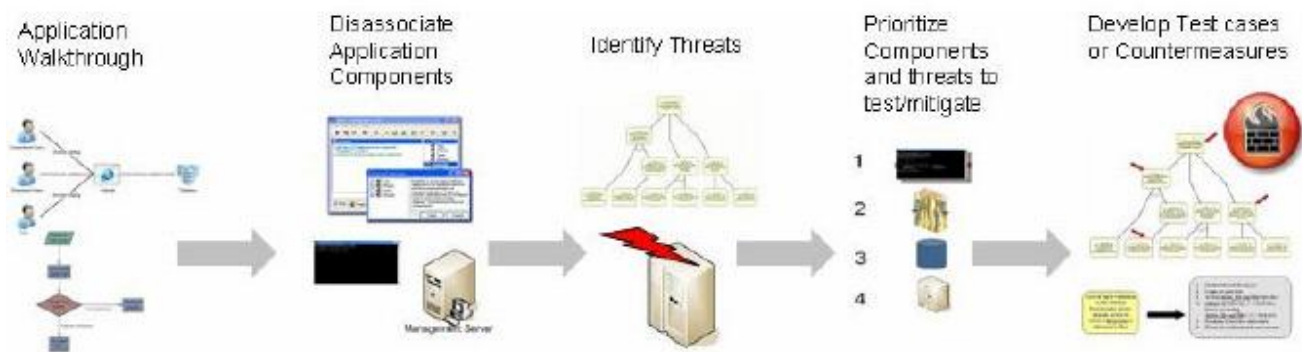


Although this procedure initially requires more effort, as mentioned earlier it helps establishing test repositories which are re-usable. Vulnerabilities corresponding to new developments in the application can easily be plugged into the threat model and test cases can then be derived.

Revisiting the threat model: The threat model is a living document which is to be continually updated as new attack techniques and vulnerability classes are discovered. The threat model is usually followed by a design change, code review or testing exercise; the findings from which are accompanied by changes or the implementation of new countermeasures. All these changes should be reflected in the threat model output so as to keep it up to date.

Threat Modelling Process Overview

An overview of the complete threat modelling process can be summarised as shown below:



Integrating Threat Modelling into the SDLC

Threat modelling can be applied at any phase of the SDLC although ideally it should be integrated at the requirements stage itself during which it helps capture overall security requirements of the project. At the design phase, application architects can analyse security requirements enabling them to take decisions regarding development of countermeasures into the solution. During application development and testing,

the threats identified can be translated to security test cases or code review guidelines specific to the solution.

Although deploying patches in a production environment would not be the most ideal situation, threat modelling attempts to identify the maximum number of defects in a single pass thereby preventing repeated down times. Using application architecture and specifications, security consultants can formulate a threat model for an application in production and derive security test cases for it, which can be more comprehensive than a conventional Black Box penetration test.

Further Information

If you would like further information about IRM's Threat Modelling services, please contact research@irmplc.com to talk to one of our security consultants.

About the Author

Varun Uppal is a Senior Security Consultant at Information Risk Management Plc (IRM) where he heads up the Threat Modelling consultancy service and is based in IRM's European Technical Centre, Cheltenham, England. Varun has five years of security consultancy experience in the UK, US and Asia including threat modelling, vulnerability research, reverse engineering and source code audit. He has also produced Data Validation Testing sections of the OWASP testing guide and has identified security vulnerabilities in commercial products such as Apple QuickTime.

About IRM

Information Risk Management Plc (IRM) is a vendor independent information risk consultancy, founded in 1998. IRM has become a leader in client side risk assessment, technical level auditing and in the research and development of security vulnerabilities and tools. IRM is headquartered in London with Technical Centres in Europe and Asia as well as Regional Offices in the Far East and North America. Please visit our website at www.irmplc.com for further information.