



Guide to the **RISK** **ASSESSMENT** **PROCESS**

ALEXANDRA INSTITUTE, 2022
ZARUHI ASLANYAN AND SIMON DIETERLE

TABLE OF CONTENTS

| | | |
|----|-----------------------|----|
| 01 | INTRODUCTION | 3 |
| 02 | GETTING STARTED | 5 |
| 03 | THE PROCESS | 7 |
| | Step 1 System | 7 |
| | Step 2 Data | 11 |
| | Step 3 Data flow | 16 |
| | Step 4 Threats | 21 |
| | Step 5 Prioritisation | 26 |
| | Step 6: Mitigation | 33 |
| 04 | CONCLUSION | 38 |

01 INTRODUCTION

WHY IS RISK ASSESSMENT IMPORTANT?

The increasing penetration of IT in every aspect of our personal and professional lives is leading to a continuous increase in attack surface. A great many organisations, irrespective of their domain, are exposed to attacks and face a vast amount of risks related to internet fraud, identity theft, spam, phishing attacks, and so on. This makes the security of systems and information of paramount importance.

An organisation's assets, sensitive data, and brand reputation needs to be protected. The way these are protected depends on the cyber-physical systems in use, the risk appetite, as well as the type of attackers they are exposed to. A risk assessment process facilitates reasoning about these dimensions, raising awareness of actual risks, and discussing potential countermeasures.

- The purpose of risk assessment is to identify and evaluate vulnerabilities as well as implement the corresponding mitigation as necessary in order to create a more secure product and/or organisation.
- Risk assessment helps create awareness of threats and the related risks, and discover whether the existing countermeasures are adequate or if more should be done.
- Moreover, risk assessment may be mandated by compliance/regulatory requirements where applicable.

THIS GUIDE HELPS YOU

In this guide, we present a risk assessment process in 6 key steps (see [FIGURE 1](#) on page 4), supported by a graphical tool named

[DRAW.IO](#). The tool makes it practical to revise the outcome of a risk assessment when some elements in the landscape change, allowing to focus only on the changes and thereby fast tracking the analysis. Besides flexibility, the tool adds some formality to the process, as risk assessments can now be stored and shared using a common format.

The guide is intended for a reader without prior knowledge of risk assessment.

The guide is inspired by another risk assessment kit for IoT products developed at the Alexandra Institute, which is available [HERE](#) (in Danish only).

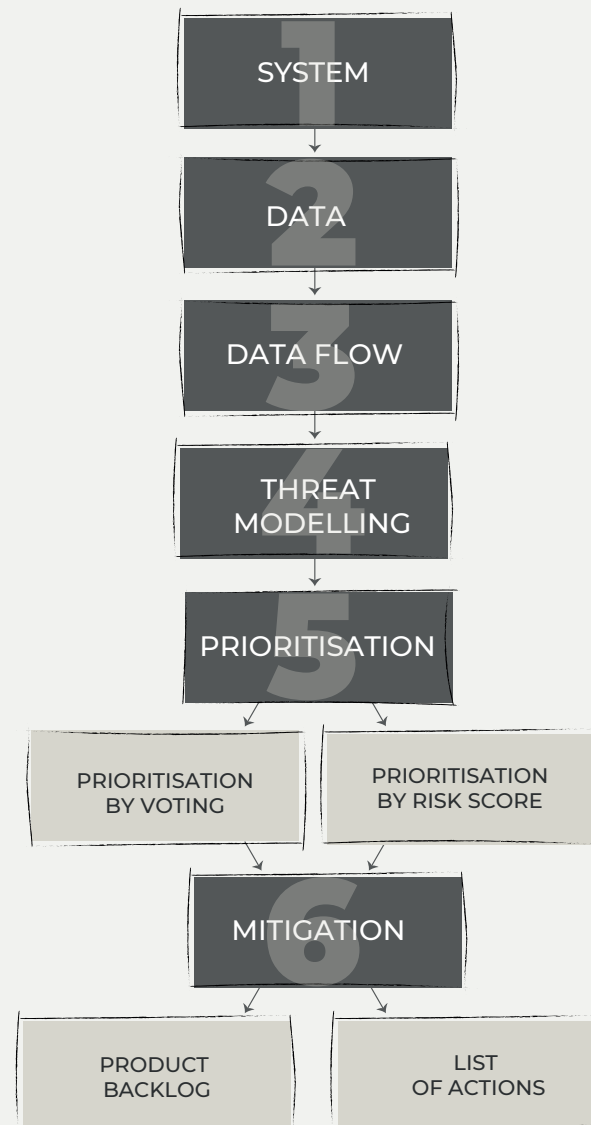


FIGURE 1: THE SIX STEPS IN THE RISK ASSESSMENT PROCESS

02 GETTING STARTED

HOW TO FACILITATE THE RISK ASSESSMENT PROCESS

SCOPING

Before you start your risk assessment, make sure to define the scope of the assessment. It could encompass a product that you want to sell, the operation of infrastructure related to this product, both combined or something in between. To be comprehensive, we recommend to include a product in an operational environment, its users, relevant physical devices, external parties and processes, including inputs and outputs. When starting out with a complex system, doing incremental assessments can be a good starting point.

Clear scoping is necessary to gather the appropriate stakeholders for the sessions. Product related assessments should be done with people from development/product management while when considering

the product in operation, it is beneficial to include people from this field.

Make it clear why the chosen scope reflects a business objective and how it provides value. Figure out if your scope is complete enough to not neglect potential externalised systems that still may contribute to the overall risk picture.

DIGITAL TOOLS

To enable remote or hybrid teams to carry out a risk assessment, the templates and tools for conducting it are provided as a pre-set using the free platform [DRAW.IO](https://draw.io).

Using this link to [DRAW.IO](https://draw.io), you can start a new risk assessment session with all necessary visuals, examples and instructions. The file can be saved in different cloud storage

systems and shared with stakeholders. This allows for collaborative editing.

EXAMPLE SCENARIO: WEBSHOP

The Example Scenario that we use throughout this guide shows a simplified view of the systems involved in ordering from a webshop. This includes the interaction from customer to the shipping provider to deliver the ordered goods. For it to serve as an easy and graspable example, surrounding systems are ignored, and only those that are mandatory for the shopping process are modelled.



Define scope

Use drawing tool:
DRAW.IO

**Follow the 6-step
risk assessment
process**



03 THE PROCESS

A 6-STEP PROCESS TO INCREASE SYSTEM SECURITY

STEP 1 SYSTEM

A requirement for performing any meaningful risk assessment is to have a clear picture of what we want to analyse, i.e., the full overview of the system architecture.

The system architecture is a model that represents the structure and behaviour of a system. It provides an overall view of the system components and their connections. For example, a system model of a webshop

could contain the webshop front end, product database, customers etc.

The architecture can be described at various levels of detail, depending on the task at hand. The goal of this step is to identify the elements that form the system and their connections. This step lays the foundation for achieving a common understanding of the system under evaluation and en-

ables to reason about data flows between components.

VISUAL ELEMENTS

FIGURE 2 on page 8 presents the most common elements for drawing different system components. These elements are pre-defined in **DRAW.IO** but users are free to define their own.




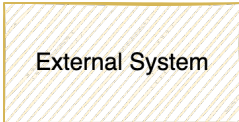

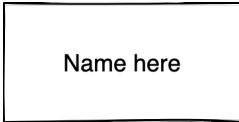
| VISUAL ELEMENTS | NAME | DESCRIPTION |
|---|-----------------|--|
|  | Applications | A program or a group of programs |
|  | Microservice | An example for a special and common type of application. Conveys more concrete meaning than just Application |
|  | Device | Input devices, output devices, storage devices, etc. |
|  | External system | Users or systems not controlled by you |
|  | Data | Any information processed or recorded by the system under assessment |
|  | Uncategorized | Any new category other than the pre-defined ones |

FIGURE 2: PRE-DEFINED ELEMENTS FOR SYSTEM DIAGRAM



INSTRUCTION

In order to produce a model of the system under assessment, we:

1. brainstorm about different system components
2. identify their interfaces and connections

Each component is assigned to a category (see **FIGURE 2** on page 8).

OUR WEBSHOP EXAMPLE

Let us consider the online shopping system described above. We start by brainstorming about the system, its components and their connections (see **FIGURE 3** on page 10).

External components: people and companies that are involved in the system run. In our webshop example, we have customer using the system, and the logistics provider.

Interfaces: we identify a shop front end and a shop administration.

Applications: we identify the following microservices:

- product microservice to manage product information,
- cart microservice to manage shopping cart,
- order microservice to manage the order placed by customer. It connects to the

logistic provider that needs to handle the shipping.

Data: we identify the following:

- stock quantity and price
- order data
- shipping information

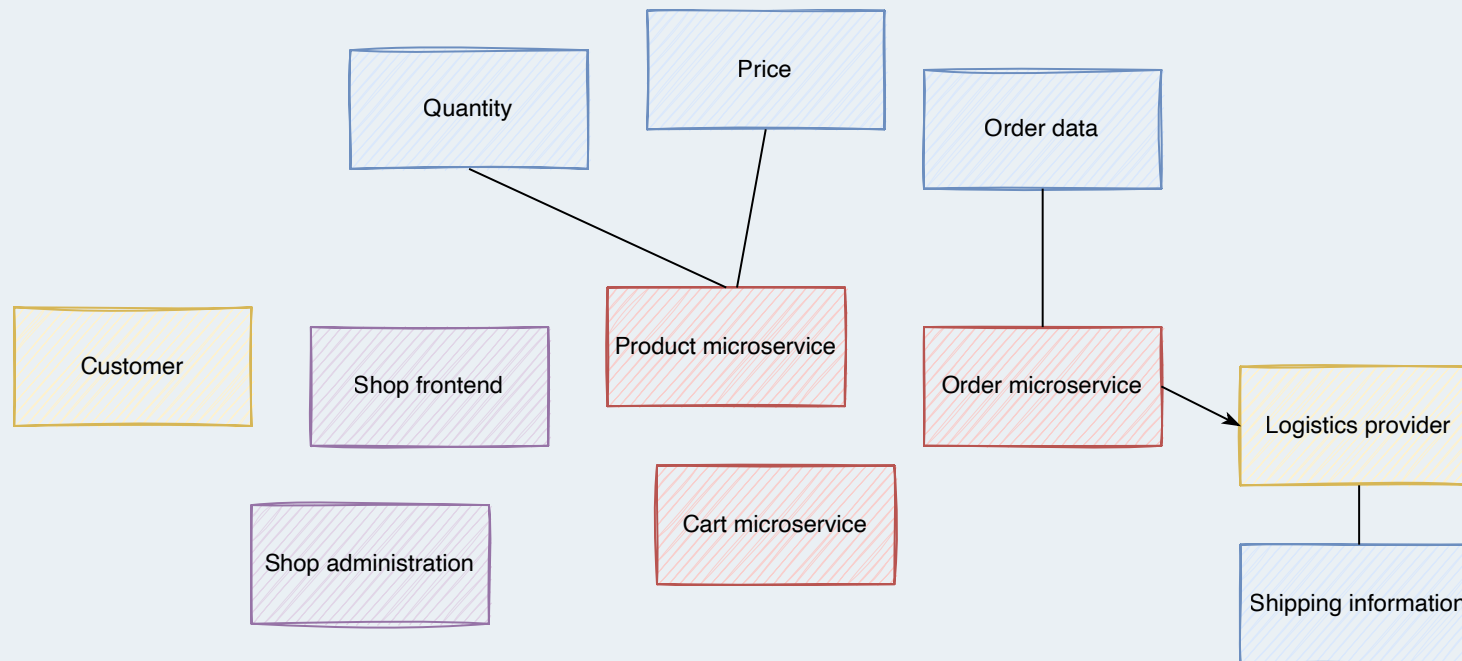


FIGURE 3: SYSTEM DIAGRAM FOR THE WEBSHOP EXAMPLE

STEP 2 DATA

To perform a risk assessment, we need to know what components are potentially under threat and why they need to be protected. In the following, we focus on the protection of data.

The huge amount of data created, stored, and manipulated every day make data critical for a great many systems. For instance, compromised data can affect the ability to operate a system, or the privacy of an individual. Moreover, data breaches can be economically and reputationally expensive.

Furthermore, legal and regulatory requirements, such as the General Data Protection Regulation (GDPR) in Europe, add extra responsibilities on safeguarding collected data. Hence, it is necessary to protect data from unauthorised access, modification, loss, or corruption.

By data we mean any information that is persisted or processed, irrespective of the purpose (operate the business, comply with regulations, perform statistical analysis, etc.).

Classic examples include customer information and product information, but also source code or configuration information that is critical for the purpose and scope of the risk assessment can be regarded and treated as data to be protected.

The goal of this step is to identify data that are important to operate your system and business, and to think about why they are important. This step provides an overview of the data types and how they are handled across the system. Moreover, it creates awareness around the impact of data being compromised.

VISUAL ELEMENTS

For representing data in the system, we use data types. A data type is defined by a name and a set of data elements representing information about the data type.

Examples of data types include product data or order data, and the corresponding elements could be the name of a product, stock quantity, total price of an order or payment information.

In the tool, a data type is represented with a yellow oval shape, while data elements are represented either with red or green rectangles depending on whether data are personal or not (see **FIGURE 4** on page 12).

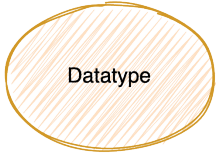


| VISUAL ELEMENTS | NAME | DESCRIPTION |
|--|---------------|--|
|  | Data types | Order data |
|  | Data elements | Payment information, stock quantity, total price |
|  | | |

FIGURE 4: PRE-DEFINED DATA SHAPES IN THE TOOL

**INSTRUCTION**

In order to identify data that are relevant for this risk analysis, we

1. brainstorm (based on the system architecture) about data required for the system to operate,
2. identify the hierarchy of data by defining data types and the corresponding elements,
3. look into all the data elements found in the previous step and find the corresponding data type. In case they are data type, then find a related element for that type.

Once we have identified all the relevant data, we discuss how important each data type is and why. For each data type, we describe its requirements in terms of Confidentiality, Integrity and Availability – CIA (see **CUE CARDS** on page 23). In other words, we reason about the following questions:

- **Confidentiality:** do we need to protect this data type from unauthorised access? Who should be allowed to access it?
- **Integrity:** do we need to prevent this data type from being modified? Who should be allowed to modify it?
- **Availability:** do we need to access the data all the time? How long can we operate without it?

OUR WEBSHOP EXAMPLE

Looking into our webshop example, we consider data identified in **STEP 1 SYSTEM** and expand it.

In step 1 we identify the following data: stock quantity and price, order data, shipping information. Let us look into each of them separately. Stock quantity and price are both attributes of a given product. Therefore, we introduce the data type product data and group these two elements under it. Moreover, we look for the additional elements, such as name of a product.

Order data is a data type, hence we look for the corresponding elements, such as Item

quantity, total price and payment information. Note that payment information is sensitive data. Shipping data is also a data type, with corresponding elements, including shipping label metadata and shipping address, the latter being sensitive data. Now that we have identified the relevant data, we describe the requirements in terms of CIA. For simplicity, we discuss in details order data. The full description can be found in **FIGURE 5** on page 15.

- **Confidentiality:** as we mentioned above, the payment information (a data element in order data type) is sensitive data, hence

no unauthorised party should be allowed to access it.

- **Integrity:** in order to perform the order, the payment information is needed, and that information should not be modified or changed. Similarly, the total price of the order should not be changed during the purchase.
- **Availability:** the order information is needed once for processing the purchase, we do not need order data at all times. On the other hand, we do need product data at all times, as without it no one can make a purchase.

| | Confidentiality | Integrity | Availability |
|---------------|---|--|--|
| Data | Requirements | | |
| Product data | Most of it is public already | Should not be changed | Without it no one can buy |
| Order data | Payment information needs to be protected | Price and payment information can never be changed | We only need it once for processing |
| Shipping data | Not relevant – is on a packet slip | Should not be changed | We only need it once to send to logistics provider |

FIGURE 5: CIA REQUIREMENTS FOR THE WEBSHOP EXAMPLE

STEP 3 DATA FLOW

Understanding how data flows across different systems and data stores is essential for understanding where problems can arise – be it reliability or security. This knowledge is encompassed in an abstraction called data flow diagram (DFD).

The data flow diagram can be used to describe a single application, a complex distributed systems or even business processes. Your data flow diagram will leverage the work done in the previous steps

(**STEP 1 SYSTEM** and **STEP 2 DATA**) by combining elements from both steps and enhancing them with information about how they interact and trust each other.

Thus, the data flow diagram contains a common understanding of the system among all the involved stakeholders and enables a structured approach to facilitating threat modelling.

VISUAL ELEMENTS

To create a data flow diagram, use the 5 different elements introduced in **FIGURE 6** on page 17.

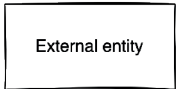



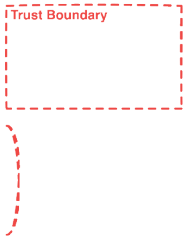
| VISUAL ELEMENTS | NAME | DESCRIPTION | USAGE QUESTIONS AND EXAMPLES |
|---|-----------------|---|--|
|  | External entity | The external entity shape is used to represent any entity outside the application that interacts with the application via an entry point, i.e., the points where data enters the system. | Can we control the process? Is it a human? Examples: Stock updates system, service provider, shopper, customer |
|  | Process | The process shape represents a task that handles data within the application. The task may process the data or perform an action based on the data. | Is data processed? Examples: Web service, DBMS |
|  | Data store | The data store shape is used to represent locations where data is stored. Data stores do not modify the data, they only store data. | What data is stored? Examples: Database, file, memory, external drive |
|  | Flow | The data flow shape represents data movement within the system. The direction of the data movement is represented by the arrow. Happens between external entity, process and data store. | What is the mechanism the data is transferred? Does the sender know the exact receiver of the data? Does the receiver confirm/acknowledge that they received the data? Examples: HTTP or gRPC, message buses or APIs, TCP or UDP |
|  | Trust boundary | The trust boundary shape is used to represent the change of trust levels as the data flows through the application. Boundaries show any location where the level of trust changes. | What is external vs internal to the system? What is on the same network segment? What is protected by a firewall? What runs on the same physical or virtual machine? What runs under the same user account? Examples: Service provider systems, admin VLAN, Host 232, SYSTEM user |

FIGURE 6: ELEMENTS IN THE DATA FLOW DIAGRAM



INSTRUCTION

To go from the work done in the previous steps to a data flow diagram, we recommend the following approach:

Look at the results from the system step; there you have already identified what in a data flow diagram would be a process: a component that processes data, such as an application, a microservice or something similar.

- Take the components from the system step and add them as process components to the data flow diagram.

Elements in the system step that are humans or third parties are marked as external entities to the system.

- Add these components as well.

Take a look at the results of the data step. You already created a collection of different data types which will help you add the next component to the data flow diagram: the data store.

- Identify where all the data types are stored and add the storage location as data store component to the data flow diagram. It is likely that multiple data types are stored in the same data store – e.g., in a database.

Now that we have transferred the prerequisites to our diagram, we focus on where interaction between the components happens. This is represented by the data flow component.

- Connect components with the data flow component where communication and interaction happens.
- Identify if the data flow is just going in one direction, where we don't care about an answer, or if data is going in both directions. Mark it by adding arrows on the ends accordingly
- Specify the kind of communication using a label on the data flow component by e.g. adding the communication protocol

TIP: If communication happens using multiple means, e.g. protocols, draw multiple data flow components.

To complete the data flow diagram, we need to mark where a trust relationship between the components exists. A trust boundary component is used to denote an existing trust relationship.

- Analyse where trust relationships exist within the system and add them to the data flow diagram. Leverage the questions and examples in the component description.

TIP: Choose the rectangle trust boundary when grouping things together, e.g. when it runs on the same server. Use the line trust boundary when isolating single components, e.g. an external entity.

- Add labels to trust boundaries reflecting what type of boundary it is, e.g. user account, virtual machine.

OUR WEBSHOP EXAMPLE

Looking through our system diagram, we can clearly identify components that process data as we already have coloured and grouped some of them as microservices. In addition, we have what we identified as applications.

The five elements are: product, order and cart microservice as well as shop front end and shop administration applications.

Keeping our scope of the simple ordering process in mind, we can group together S shop administration and product microservice as their actions are related to reading and updating product data. We will call it the product management process.

An additional grouping we can make is combining shop front end and the cart microservice, as in our scenario the customer will only interact with the checkout related functionality of the webshop, i.e. the cart. We call this combination the shopping cart process. This leaves us with the order microservice that we add to the diagram as is. The two identified external elements to our

system are the customer and the logistics company which can be copied into the diagram as is. We took the liberty of renaming the 'customer' to 'shopper' in this step. We think that reflects a better persona and also shows that the naming of components is not set in stone as long as there is a common understanding of component names.

Each of our three identified data types is stored in its own data store based on an architectural decision to segment data in a similar fashion. Therefore, we use them as is and note them down as data store components.

To connect our components in the data flow diagram, we can take a look at what was already identified in the system step or just go element by element.

In our case let us discard all the interactions we identified before and do it from scratch with the components that we came up with in this step starting with external entities.

Shopper: they only interact with the shopping cart but also get feedback from it – e.g.,

when the quantity changes from 3 shirts to just one.

- ↔ A bidirectional arrow between those elements
- No further interaction from the shopper

Logistics company: they get information from us when the order management process decides the order goes through and we get tracking information back.

- ↔ A bidirectional arrow between those elements
- No further interaction from the logistics company

Let us continue with our own processes.

Shopping cart: to display information about the products, it needs to read product information like e.g., descriptions. Yet it has no need to add or modify this information. When the shopper decides to check out the shopping cart, the information of the contents of the cart is transferred to the order management to be fulfilled. When the order is successful, the order management reports back and clears the cart.

- ↔ A bidirectional arrow between shopping cart and order management
- → A directional arrow from product information to the shopping cart that reads it
- No further interaction from the shopping cart

Product management: to manage the catalog of available products, product information has to be read, updated and deleted. When orders are fulfilled, the stock quantity needs to be checked and updated as well.

- ↔ A bidirectional arrow between product management and product information
- ↔ A bidirectional arrow between product management and order management

Order management: orders are stored for e.g., billing and fulfilment purposes. For package tracking capabilities, the shipping information is kept for a period of 2 weeks after package delivery.

- ↔ A bidirectional arrow between order management and order information

- ↔ A bidirectional arrow between order management and shipping information

Finally, to complete our data flow diagram, we identified the most important of our trust boundaries between all of our systems and the external entities. This boundary exists in all data flows where externals are involved and is a good place to start. Depending on how thorough you want or need to be, you can dive deeper into the architecture to identify the less obvious boundaries.

In our case this could be e.g., system sharing access to product information because they need to have credentials to the same data store. But in this small example with a model on this abstraction level, we won't gain too much from it.

STEP 4 THREATS

Assessing potential threats to a system helps to align resources to address risk to business operation in a structured way. Regardless of the approaches taken, thinking about negative effects on the system in scope supports the goal of understanding and managing risk. The process of structured assessment of threats is called threat modeling and can

take different forms depending on how much experience stakeholders have and how thorough the approach needs to be. In assessing your threats you create an overview of relevant issues based on the common understanding created in the previous step. Having come up with threats helps

you to prioritise and decide on potential mitigation options and strategies.

VISUAL ELEMENTS

To guide the threat modelling process and keep track of threats that were found, the threat table component shown in **FIGURE 7** can be used.

| VISUAL ELEMENT | | | | | NAME | DESCRIPTION |
|-------------------------|-------------|-----------------|-----------|--------------|--------------|--|
| Threats in interactions | | Confidentiality | Integrity | Availability | Threat table | Is used for keeping track of threats across CIA categories |
| Element | Interaction | Threat | Threat | Threat | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

FIGURE 7: THREAT TABLE COMPONENT



INSTRUCTION

For the process of threat modelling, we leverage work done in the previous step where we created a data flow diagram.

Security vulnerabilities usually exist on the boundary between different systems due to them having different trust relationships. That's the reason why this should be the focus of the assessment.

Start by identifying processes where data flows across trust boundaries. As different data flows to and from one component, e.g. a process, carry different operations, data and therefore potential threats, we look at them individually.

Add pairs of components and associated data flows as rows to the threat table in the first and second columns. E.g, while looking

at process A, a data flow from process B to process A would be noted to process A as element and to process B as interaction.

Once this has been done, start identifying the threats for these elements and interactions by asking what can go wrong for each of those interactions.

Start discussing what can go wrong for a process/external entity when it has an interaction coming *FROM* another component via a data flow. This will reveal important and critical threats.

For interactions going *TO* another component, the impact on the sending component is usually limited but can also reveal problems. Put the found threat in the column according to which security requirement is violated (see **STEP 2 DATA**). Try to

find 1 or 2 threats for all incoming (FROM) connections and 1 for outgoing (TO) connections per column.

TIP: Use the provided cue cards on page 23 to get inspiration on potential threats to each of the security goals of CIA.

Repeat this step for all pairs in the threat table.

When finished with processes with data flows across trust boundaries, high risk threats are covered. You are now able to continue with the next step. Yet, you are also free to continue with processes where the data flow only stays within a trust boundary if time allows.

FIGURE 8: CUE CARDS

OUR WEBSHOP EXAMPLE

In our webshop example, we have two interactions outside our system's trust boundary: between shopper and shopping cart as well as order management and logistics company. Let us start and follow the steps of common online shopping.

For the shopper ↔ shopping cart interaction we should look at both directions when we want to be thorough in modelling threats. As we focus on threats that we can mitigate, we pick the component we can control: the shopping cart process.

Shopper → Shopping cart: we need to think about how the confidentiality, integrity and availability of our shopping cart process can be compromised due to an accidental or malicious action by the shopper.

"How can a shopper threaten the confidentiality, integrity and availability of our shopping cart process."

Confidentiality: a threat that leaks confidential data handled by our shopping cart must be caused by something in our process that changes what our code does. That could be due to user receiving data that we didn't expect and that wasn't rejected before being processed.

Integrity: we expect our shopping cart to have some items in it and a quantity per item. Based on that, we can create orders. Breaking the integrity of how our systems work could be due to someone being able to change the quantity of one item to a negative amount which in turn with the calculation of price = quantity * item price would

lead to a negative cost of the order that could allow someone to get items for free.

Availability: the availability of the shopping cart process might be compromised if certain processing steps take substantially longer due to unexpected calculations. E.g., a user having thousands of items each with a quantity of 1 million in a cart and recalculating their prices slows down other customers.

It might not be obvious that the inverse interaction from shopping cart → shopper is something that threatens your system but in our case, it is pretty obvious that we weren't here without our customers so let's consider how our shopping cart could threaten the shopper.

"How can our shopping cart threaten the confidentiality, integrity and availability of a shopper or the data."

Confidentiality: it might be possible that the wrong shopping cart content is sent to the wrong shopper. E.g., simply changing a shopping cart id in a cookie or in the URL might do the trick.

One can argue that this is exactly the same as if a shopper sends malicious data to the shopping cart as in the previous section. In many cases, that might be the case but we argue here that taking the different viewpoints can also lead to different causes for the threat to happen.

Integrity: as our shopping cart directly displays information it gets from other systems,

we are displaying data we have no control over. Even though these systems are inside of our trust boundary, it could happen that we are forced to display bogus data to the shopper. Our goal should be to prevent that also.

Availability: this is a complex topic and might not be as relevant and obvious as the other topics. The availability of external is mostly relevant when we want something from them, and in this case they want something from us. That is why we are declaring it as irrelevant.

Now we can continue with other data flows across the trust boundaries followed by the internal data flows.

NOTE

Keep in mind that the goal is not to fill each and every box all the time but cover the common potential attack surface of the system in operation.

So if you cannot find a threat at this moment in time, mark the corresponding cell and maybe revisit it later.

It is better to look at more interactions than to be through with only a few.

STEP 5 PRIORITISATION

The previous **STEP 4 THREATS** will produce a huge list of potential threats to the system. Given limitations on resources and budget, it might not be possible to mitigate all the identified vulnerabilities. Hence, the prioritisation helps focus on a selected number of threats first.

We propose two alternative ways of prioritising the identified threats:

1. prioritisation by voting and
2. prioritisation by risk score

In both cases we consider the risk for each threat. By risk we mean the damage that attacks against systems can cause. Moreover, the risk can be estimated on the basis of its impact and likelihood.

STEP 5.1 PRIORITISATION BY VOTING

A first method of sorting the threats is by voting. This is an intuitive approach to prioritising threats and usually well-suited to organisations with limited background knowledge on threat modelling. The goal is to prioritise threats based on a holistic, intuitive assessment of their combined likelihood and impact. This step determines importance and relevance of the threats.

Observe that the system might already have some countermeasures in place. These countermeasures might reduce the overall risk, hence, we should consider them during the voting process.

Moreover, the following questions can help reflect on the risk for each threat:

- Do we have evidence of a threat?
- Did we see it before?
- How exposed is the potential vulnerability?
- What is the worst-case scenario?
- Could the vulnerability be combined with others to make it worse?
- What is the impact?

Before the voting, each participant is assigned a number of votes that they can use. The number depends on the total number of threats to prioritise.

VISUAL ELEMENTS

For performing the voting, we need the list of threats and a voting marker, pre-defined as a red circle in the tool (see **FIGURE 9** on page 27).


| VISUAL ELEMENTS | | | | | NAME | DESCRIPTION |
|---|-------------|-----------------|-----------|--------------|---------------|---|
| Threats in interactions | | Confidentiality | Integrity | Availability | Threat table | Is used for keeping track of threats across CIA categories |
| Element | Interaction | Threat | Threat | Threat | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
|  | | | | | Voting marker | Is used by participants for voting for their preferred prioritisation |

FIGURE 9: PRE-DEFINED SHAPES FOR VOTING IN THE TOOL



INSTRUCTION

In the tool, the voting is done by copying the table of threats from **STEP 4 THREATS** into the voting diagram by placing red circles on a threat (see example in **FIGURE 10** on page 29). Participants cannot vote for the same threat more than once.

The highest-scored threats will be evaluated further for corresponding actions in the next steps.

OUR WEBSHOP EXAMPLE

Let us consider the identified 14 threats in **STEP 4 THREATS** and assume that we have two participants in the workshop. Before the voting, each participant is assigned four votes. Note that the number of assigned votes depends on the total number of threats, hence with an increase in the total number of threats, the assigned votes will increase as well.

Each participant votes independently. **FIGURE 10** on page 29 illustrates a possible outcome. From the figure we can identify the threats with the highest score and evaluate them further in the next step.







| | | Confidentiality | Integrity | Availability |
|--------------------|--------------------------|---|--|---|
| Element | Interaction | Threat | Threat | Threat |
| Shopping cart | From shopper | Malicious input data breaks code flow  | Shopper adds a negative amount of to their cart and prices get negative | Getting input data crashes the system |
| Shopping cart | To shopper | Shopper receives confidential data about other buyers  | Browser displays bogus data | Not relevant |
| Order management | From logistics company | Too much information is exposed for shipping change callbacks  | Shipping change events sent with wrong customer association  | They send too much data |
| Order management | To logistics company | We send more information than necessary as part of shipping contract creation  | ... | ... |
| Shopping cart | From product information | Products we might not want public yet are accessible via the shopping cart  | No data is modified. Let's look at potential risks of modified product information below | Unavailable product information might prevent cart page from rendering and therefore sale |
| Product management | To product information | ... | Price information is set incorrectly (e.g., way too low) | ... |

FIGURE 10: AN EXAMPLE OF VOTING FOR THE WEBSHOP EXAMPLE

STEP 5.2 PRIORITISATION BY RISK SCORE

A second method of sorting the threats is by risk score. It follows a more structured and qualitative approach and is suitable for organisations with previous knowledge on threat modelling.

The goal of this method is to prioritise the threats on the basis of explicit risk factors,

such as likelihood and impact. This step determines importance and relevance of the threats.

VISUAL ELEMENTS

Before scoring, we list the threats identified in **STEP 4 THREATS** into the confidentiality, integrity and availability tables (see **FIGURE 11**).

| Confidentiality | | | | Integrity | | | | Availability | | | |
|-----------------|--------|------------|-------|-----------|--------|------------|-------|--------------|--------|------------|-------|
| Threat | Impact | Likelihood | Score | Threat | Impact | Likelihood | Score | Threat | Impact | Likelihood | Score |
| | | | | | | | | | | | |
| | | | | | | | | | | | |

FIGURE 11: PRIORITISATION TABLE IN THE TOOL



INSTRUCTION

In order to prioritise the identified threats, we:

1. estimate the likelihood of each threat, i.e., how likely it is that an attacker can exploit the threat
2. estimate the impact of each threat, i.e., what is the impact on the system if the threat were to be exploited
3. based on the likelihood estimate and the impact estimate, calculate the overall risk score for each threat using the matrix in **FIGURE 12**

Note that likelihood and impact range over the following values:

- Very Low (VL)
- Low (L)
- Medium (M), High (H) and
- Very High (VH)

There are various ways to estimate likelihood and impact. For example, by raising hands, having an open discussion to identify the values, or voting in private and then discuss the results. During the scoring, we should consider if countermeasures are already in place, which might reduce the overall risk. The highest-scored threats will be evaluated further for corresponding actions in the next steps.

| | | Impact | | | | |
|------------|-----------|----------|-----|--------|------|-----------|
| | Score | Very Low | Low | Medium | High | Very High |
| Likelihood | Very Low | 2 | 3 | 4 | 5 | 6 |
| | Low | 3 | 4 | 5 | 6 | 7 |
| | Medium | 4 | 5 | 6 | 7 | 8 |
| | High | 5 | 6 | 7 | 8 | 9 |
| | Very high | 6 | 7 | 8 | 9 | 10 |

FIGURE 12: RISK EVALUATION MATRIX

OUR WEBSHOP EXAMPLE

Let us consider the identified 15 threats in **STEP 4 THREATS** and list them into CIA tables.

In this guide we will only look into the confidentiality table. For each of the threats in the table, we will estimate the likelihood and impact. For example, the likelihood of

malicious input data breaking code flow is low, while the impact is high, due to privacy reasons. Based on these estimates and the **RISK EVALUATION MATRIX** in **FIGURE 12**, the overall risk is 6.

FIGURE 13 illustrates the estimated likelihood, impact and the corresponding score for each threat.

| Confidentiality | | | | Integrity | | | | Availability | | | |
|---|--------|------------|-------|--|--------|------------|-------|---|-----------|------------|-------|
| Threat | Impact | Likelihood | Score | Threat | Impact | Likelihood | Score | Threat | Impact | Likelihood | Score |
| Malicious input data breaks code flow | High | Low | 6 | Shopper tampers with item amount to reduce price | Medium | Low | 5 | Getting input data crashes the system | Very high | Medium | 8 |
| Shopper receives confidential data about other buyers | High | Low | 6 | Browser displays bogus data | High | Medium | 6 | They send too much data | High | Low | 6 |
| Too much information is exposed for shipping change fallback | Medium | Low | 5 | Shipping change events sent with wrong customer association | High | Low | 6 | Unavailable product information might prevent cart page from rendering and therefore sale | High | Low | 6 |
| We send more information than necessary as part of shipping contract creation | Low | Very low | 3 | No data is modified. Let's look at potential risks of modified product information below | ... | ... | ... | ... | ... | ... | ... |
| Products we might not want public yet are accessible via the shopping cart | Medium | Very low | 4 | Price information is set incorrectly (e.g., way too low) | Medium | Very low | 4 | ... | ... | ... | ... |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |

FIGURE 13: THREAT PRIORITISATION BY RISK SCORE FOR THE WEBSHOP EXAMPLE

STEP 6: MITIGATION

Congratulations!

A lot of work went into creating a risk assessment using threat modelling for your system. Combining what data the system uses with how and where it is processed allowed us to build a data flow diagram that made it substantially easier to systematically identify potential threats and risks.

Regardless if you are still in the process of building the system or if you retroactively looked at a system that already exists, you have established the means of improving security. However, there is still some work to be done: managing those threats and risks. This is best done by avoiding the threats all together and fixing the vulnerable parts of

the system. If fully avoiding is not possible, reducing the risk in any way is the second best choice.

We present two approaches for how to organise the mitigation of these risks depending on how your teams work.

STEP 6.1 AGILE APPROACH: BACKLOG

TIP: Choose this method if you already work with a backlog centered approach.

After identifying and prioritising threats to the components of our system, steps to mitigate those threats need to be taken. This can be carried out in different ways depend-

ing on the implementation of agile methodologies in the team. For the top threats, you need to decide how they should be added to the product backlog. The provided security backlog reference makes a suggestion and gives examples.

VISUAL ELEMENTS

To continue with that task, using the components is optional. If you have easy access to your product backlog tooling, add the backlog items directly there. If that is not the case, use the threat backlog mapping component (**FIGURE 14** on page 34) to keep track of what you want to add to your backlog later.

| VISUAL ELEMENTS | | NAME | DESCRIPTION | USAGE | | | | | | | | | | | | |
|---|--|---|-------------|--------------|---|------------|---|---|---|---|--|--------------------|--|----------------------------|--|--|
| <table><tr><th>Action</th><th>Description</th></tr><tr><td>User stories</td><td>For implementing controls focused on security. E.g., Authenticate User.</td></tr><tr><td>Spikes</td><td>Used to investigate if system is actually vulnerable or which solution is best.</td></tr><tr><td>Acceptance Criteria</td><td>Most common security fix as extra testable scope to existing stories. E.g., Authorisation Checks.</td></tr><tr><td>Epics</td><td>For significant elements of security architecture identified. E.g., Security Logging, Identity Provider.</td></tr><tr><td>Definition of Done</td><td>Security criteria that are required consistently across stories.</td></tr></table> | | Action | Description | User stories | For implementing controls focused on security. E.g., Authenticate User. | Spikes | Used to investigate if system is actually vulnerable or which solution is best. | Acceptance Criteria | Most common security fix as extra testable scope to existing stories. E.g., Authorisation Checks. | Epics | For significant elements of security architecture identified. E.g., Security Logging, Identity Provider. | Definition of Done | Security criteria that are required consistently across stories. | Security Backlog Reference | | Use it as a reference |
| Action | Description | | | | | | | | | | | | | | | |
| User stories | For implementing controls focused on security. E.g., Authenticate User. | | | | | | | | | | | | | | | |
| Spikes | Used to investigate if system is actually vulnerable or which solution is best. | | | | | | | | | | | | | | | |
| Acceptance Criteria | Most common security fix as extra testable scope to existing stories. E.g., Authorisation Checks. | | | | | | | | | | | | | | | |
| Epics | For significant elements of security architecture identified. E.g., Security Logging, Identity Provider. | | | | | | | | | | | | | | | |
| Definition of Done | Security criteria that are required consistently across stories. | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | |
| <table><tr><th>Threat</th><th>Actions</th><th>More details</th></tr><tr><td>Shopper can crash shopping cart</td><td>USER STORY</td><td>Input validation, where, what, how?</td></tr><tr><td>Logistics company gets too much information</td><td>ACCEPTANCE CRITERIA</td><td>We need to know what data is sent to which external party and when before accepting user stories involving externals.</td></tr><tr><td>...</td><td>...</td><td>...</td></tr></table> | | Threat | Actions | More details | Shopper can crash shopping cart | USER STORY | Input validation, where, what, how? | Logistics company gets too much information | ACCEPTANCE CRITERIA | We need to know what data is sent to which external party and when before accepting user stories involving externals. | ... | ... | ... | Threat Backlog Mapping | | Use it to keep track of the decisions if you can't add the items directly to your product backlog. |
| Threat | Actions | More details | | | | | | | | | | | | | | |
| Shopper can crash shopping cart | USER STORY | Input validation, where, what, how? | | | | | | | | | | | | | | |
| Logistics company gets too much information | ACCEPTANCE CRITERIA | We need to know what data is sent to which external party and when before accepting user stories involving externals. | | | | | | | | | | | | | | |
| ... | ... | ... | | | | | | | | | | | | | | |

FIGURE 14: THREAT BACKLOG MAPPING COMPONENT

**INSTRUCTION**

To integrate security work in the normal cycle of software development, mapping the tasks to your agile workflow is essential.

- Discuss threat by threat and decide which action should be taken and put it into the product backlog or add it to existing items
- Try to estimate efforts for stories, spikes, and epics if possible so that no additional backlog grooming is necessary.

OUR WEBSHOP EXAMPLE

Go through the prioritised list of threats and discuss how you want to approach them.

For our example we picked two threats that we added as misuse story to our product backlog: *“As an attacker, I want to send malicious data, so that the shopping cart crashes and becomes unavailable.”*

Our second threat: *“Logistics company gets too much information”*, we want to address more widespread than just the one case with the logistics company. Therefore, we are going to change the acceptance criteria for our stories involving external parties so that we can get a grip on all of the cases.

STEP 6.2 LISTS OF ACTIONS

Once we have identified the highest-scored threats, the next and final step is to discuss possible mitigations. The goal is to agree on who is responsible for each threat, and to identify actions and tasks for mitigating it. This step ensures that the threats are properly addressed and will be followed up on.

VISUAL ELEMENTS

In the tool, we use **FIGURE 15** to record the outcome of the step.

| Threat | Owner | Actions / Comments (optional) |
|----------|----------|-------------------------------|
| Threat 1 | Person 1 | Countermeasure 1, 2, 3 |
| Threat 2 | Person 3 | Evaluate possible options |

FIGURE 15: THREAT ACTION TABLE



INSTRUCTION

In order to assign responsibilities and identify actions, we:

1. discuss and agree on a threat owner
2. decide on actions by considering the potential countermeasures

| Threat | Owner | Actions / Comments (optional) |
|---|---------------------------------------|--|
| Getting input data crashes the system | Person 1 | Evaluate possible options regarding input validation |
| Browser displays bogus data | Person 2 / Developer | Exploratory testing on most relevant pages |
| We send more information than necessary as part of shipping contract creation | Person 3 and data privacy responsible | Evaluate and document |
| ... | ... | ... |

FIGURE 16: LIST OF ACTIONS FOR THE WEBSHOP EXAMPLE

OUR WEBSHOP EXAMPLE

Let us look into the highest-scored threats identified in **STEP 5.2 PRIORITISATION BY RISK SCORE** on page 30. For each threat we will identify a threat owner and a list of actions to mitigate the threat. Consider now the threat with the highest score in our webshop example, *“getting input data crashes the system”*.

First, we assign an owner to the threat, for instance Person 1 in the relevant team. Then we brainstorm on the possible actions and decide which ones to undertake. For instance, we can evaluate possible options regarding input validation.

FIGURE 16 presents the owners and some actions for the remaining threats in our webshop example.

04 CONCLUSION

Risk assessment is the process of identifying and evaluating potential vulnerabilities, as well as considering mitigations provided by security controls planned or in place.

Risk assessment creates awareness of threats and the related risks and helps improve the overall security of a system/organisation.

In this guide, we described a risk assessment process in 6 key steps. The process we presented is supported by an online graphical tool, allowing to increase efficiency of the process and reap the many benefits, such as consistency, repeatability, collaboration and ease of reporting.

Alexandra Institute, 2022

ALEXANDRA INSTITUTE

IT CITY OF KATRINEBJERG

Aabogade 34 · DK-8200 Aarhus N

+45 70 27 70 12

IT UNIVERSITY OF COPENHAGEN

Rued Langgaards Vej 7, 5D · DK-2300 Copenhagen S

+45 70 27 70 91



The Alexandra Institute helps private and public organisations develop innovative solutions, products and services based on state-of-the art IT research. Our mission is to enable Danish companies realise the potential of new technology.