Til hovedinnhold

Datatilsynet

Software development with Data Protection by Design and by Default

The Norwegian Data Protection Authority has developed these guidelines to help organisations understand and comply with the requirement of data protection by design and by default in article 25 of the General Data Protection Regulation. We have cooperated with security professionals and software developers in public and private sector among others.

About the guidelines

These guidelines are primary intended for developers, software architects, project managers, testers, data protection officers and security advisors. Everybody who develops and contributes to the development of software containing or processing personal data.

Software development begins with an idea of creating a product that will help to simplify or improve the quality of a process or task. There are functional requirements to how the software should solve the task.

Background

Software development should follow a methodology with key activities to ensure that the final product is robust. There is some technical literature that focuses on security by design as part of developing software. However, there is less about data protection by design and by default as part of developing software. While working on this guide, we have used Microsoft Security Development Lifecycle (SDL), Secure Software Development LifeCycle (S-SDLC) and ENISA; Privacy and Data Protection by Design – from policy to engineering, as a starting point, and explored how to incorporate data protection principles, subject rights, and the requirements of the GDPR into every step of the process.

Seven activities in a continuous process

The circular diagram at the start of this guide contains the seven key activities in the process of developing software with data protection by design and by default. Each activity is depicted as a jigsaw piece or a step leading to the next activity in the circle. We chose a circle to illustrate because both software development and data protection are continuous processes.

This guide describes each activity in the process, alongside our recommendations on how to carry it out, and what measures we consider important to successfully comply with the requirement to build data protection by design and by default into software. However, this does not mean that each organisation must follow this process to the letter when building data protection into software. It is up to each organisation to decide which methodology to use, which measures, areas, and activities should be emphasised, and where to increase your efforts. The choice of methodology is typically influenced by the services provided, the industry, the type of software being developed, and considerations/perceptions relating to the organisation's own level of risk.

This guide should not dictate a strict, sequential, and rigid process for each activity in the development process. It should rather be adapted into the specific methodology the organisation has chosen to use.

Organisations that develop and release software at a fast pace, should consider how the guide best can be used to ensure that basic data protection and security parameters are in place, and how, for example, data protection and security tests can be included in fully automated testing regimes.

Summary of the activities

The seven activities in this guide start with **training**. In the section on training, we cover the most important topics on which to provide training, why, how to do this, and which tools to use.

The section on **requirements** describe the measures needed to ensure data protection and security, the tolerance levels the organisation should set for data protection and security, and the need to assess both security risks and data protection implications.

The next section takes these requirements further, to **design**, by dividing them into data oriented and process oriented design requirements. During this activity, it is important that the organisation carries out both threat modelling and an analysis of the attack surfaces.

The **coding** activity underlines the importance of developers using approved tools and frameworks, disabling unsafe functions and modules, and regularly carrying out static code analysis and code review.

The section on **testing** involves a recommendation to test whether data protection and security requirements are implemented properly, a description of what sort of security testing should be carried out, and an explanation of the importance of threat modelling and analysing the attack surface.

Before **release**, an incident response plan should be established, and a full security review of the software should be carried out. Release is then approved and all relevant data from the entire development process are archived.

The final activity involves the **maintenance** of the software and the response to incidents. The organisation should be prepared to respond to incidents, personal data breaches, faults and attacks, and be capable of issuing updates, guidelines, and information to users and those affected by the software.

These guidelines are developed in cooperation with security experts and software developers from both the private and public sectors.

Special thanks go to:
Dagfinn Bergsager (University of Oslo), Johannes Brodwall (Sopra Steria), Andreas Hegna (Storebrand), Karoline Klever (Microsoft), Rita Nortug (Nets), Eirik Saltkjel (Directorate of eHealth), Eskild Storvik (Capgemini).

What is data protection by design?

Profiling, automated decision-making, and personalised services have become part of our day-to-day lives. These trends often involve processing of personal data on a large scale. Users expect services to both be secure and safeguard their privacy in an effective manner. Businesses that take data protection issues seriously, build trust. Thus, strong data protection measures can be a competitive advantage.

Data protection legislation contains basic principles for safeguarding the privacy of data subjects. Data protection by design and by default helps ensure that the information systems we use fulfil these data protection principles, and that the systems safeguard the rights of data subjects.

Data protection by design, and data protection by default, are central requirements in the General Data Protection Regulation (GDPR) that apply from May 2018. This guide describes how to comply with these requirements. The data controller must comply with the requirements governing data protection by design during software development, and when ordering systems, solutions, and services. The requirements must accordingly also be included when entering into agreements with suppliers, and when using consultants.

Transparency is a principle in the new regulation, and it is crucial when building data protection into software. Transparency about the use of personal data involves providing information about what is being processed, by whom, why, how, and for how long it is kept. In order for data subjects to exercise their rights, organisations must be open about their processing of personal data. That way, the data subjects can make informed decisions about whether or not to use a software, and this ensures the legitimacy and effectiveness of the data controller.

Management commitment is crucial for making the decision to apply the principles of use data protection by design in the organisation's procurements and software development. Management must also ensure to provide sufficient resources for this task. Taking data protection into account throughout the development process is both cost-effective and more efficient than making changes to an existing piece of software. Enterprises that do not comply with the GDPR risk significant costs, in the form of both fines for breaking the law, liability to the data subjects, and loss of revenue resulting from damage to their reputations.

Training

During this activity, the specific types of training that should be given are determined. To ensure that everyone in the organisation understands both the need for, and the risks associated with, data protection and security, the training needs to be structured.

The target group for this activity is management and employees in the organisation.

What is important to learn

An understanding of data protection and information security is a prerequisite for developing software with data protection by design and by default. Employees should know what requirements are applicable, what they should look out for, and which tools enables them to convert knowledge of data protection and information security into software that safeguards it.

Employees must also know which methodology and routines should be followed. The organisation itself must decide what is relevant, and what type of training is required for individual employees. A training plan should be drawn up.

Which requirements apply for the organisation?

The employees should receive training in the relevant internal and external requirements. Internal requirements may relate to data protection, information security, internal control, and resource management. This includes routines for risk assessment and requirements for documentation. External requirements include data protection law in general, the significance of the data protection principles in particular, and rights of the data subjects.

Other external requirements might include regulatory and mandatory requirements related to the subject area, sector, or industry for which the software is to be developed. There may also be a requirement to follow best practices, standards, code of conducts for the chosen technology. Examples of these include the Freedom of Information Act, the Patient Records Act, the pending ePrivacy Regulation, the Regulations on the Use of Information and Communications Technology (ICT), the framework for information security (for example ISO27001
[https://www.iso.org/isoiec-27001-information-security.html]
, and the ISF Standard of Good Practice for Information Security (SoGP)
[https://www.securityforum.org/tool/the-isf-standard-good-practice-information-security-2018/]
.

How to do this in practice Software developers should have an established development methodology, approved by management, that they follow when developing software. When developing software that processes personal data, the methodology should include data protection by design and by default, and security by design. Examples of development frameworks with embedded security are Microsoft Security Development Lifecycle (SDL)
[https://www.microsoft.com/en-us/sdl/]
and OWASP Flagship projects
[https://www.owasp.org/index.php/OWASP_Project_Stages]
.

Which tools can be used? The organisation should prepare an overview of the tools, standards, and best practices that should be used during software development. Employees should be trained in which tools they can use, how to use them, and for what purposes. Examples of tools for tasks including security testing, setting security requirements, measurement, and threat modelling can be found in:

OWASP Application Security Verification Standard Project (OWASP ASVS)
[https://www.owasp.org/index.php/Category:OWASP_Application_Security_Verification_Standard_Project]

OWASP Top 10
[https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project]

OWASP Testing Project
[https://www.owasp.org/index.php/OWASP_Testing_Project]

ISO27k information security
[http://www.iso27001security.com/]

Microsoft SDL
[https://www.microsoft.com/en-us/sdl/]

Download

Example of a checklist for what training activities should include (pdf)
[/globalassets/global/english/guidelines/privacy-by-design/checklist-training.pdf]

Requirements

This activity revolves around setting requirements for data protection and information security for the final product. These requirements must reflect the need for data protection and information security, and should be included as part of the project plan.

The target group comprises everyone involved in development, or with ownership of the software, including those responsible for defining the requirements, the product owners, customers/purchasers, project leaders, developers, architects, and testers. The Data Protection Officer and security advisor should also be involved in this activity.

When requirements for data protection and security, tolerance levels, data protection impacts, and security risks are detected early, the development team will already know which requirements they need to meet, and can therefore mitigate the risks associated with data protection and information security throughout the development process.

Requirements for data protection and information security

To set the correct requirements, it is important to know what categories of personal data will be processed in the software, what conclusions can be drawn about individuals based on the data being processed, who is the user and owner of the data, who is defined as the controller, and if applicable, who is the data processor or recipient of the personal data. This is necessary for determining which laws, rules, guidelines, and codes of conduct are applicable to the software being developed. These provide guidelines for determining which requirements should be set for the software.

Relevant requirements for data protection and security are contained in the data protection regulation, business practices and policies for data protection and information security, various security standards, and codes of conduct or other relevant laws and regulations relating to the sector. The company must decide for itself which requirements are relevant to their business, the software being developed, and the context in which the end product will be used. Requirements for data protection and information security should be formulated in a checklist that should be integrated into the project plan, and monitored throughout the development process.

Meet the data protection principles

The most important requirement applicable to software with data protection by design is that the data protection principles are met. The processing shall be lawful, fair and transparent. Processing of personal data shall be carried out for specified, explicit and legitimate purposes, and only data that is necessary for the software to function shall be collected.

By “necessary”, we mean that you must assess the amount of personal data, extent of their processing, the period of their storage, and their accessibility. For example, you should consider the level of detailed information required, how long the data will need to be stored, whether automatic deletion routines can be implemented, where the data will be stored, and who will have access to the data, and from where.

Concise information and secure the data

Clear and concise information about how the personal data will be used is fundamental to ensure protection of data subject’s rights. The software must make it easy for the data subjects to exercise their rights, such as access, information, rectification, restriction, and data portability. This can, for example, be resolved by using a login portal that provides an overview of the registered data, information on users’ rights, and a help form that can be used to make objections or rectifications.

The company must ensure the security of personal data during e.g. collection, storage, alteration, viewing, communication, and deletion. Encryption and access control are examples of measures that can be used to help ensure security.

The security requirements for the software are determined by identifying which risks the software may be exposed to, and which risks the company is willing to take. This defines the parameters for selecting relevant

and correct measures for the company and the software. We recommend using the Data Protection Authority's checklists and recognised standards for information security when selecting relevant measures.

Generic examples of suggested measures:

OWASP ASVS
[https://www.owasp.org/index.php/Category:OWASP_Application_Security_Verification_Standard_Project]

ISO27001
[https://www.iso.org/standard/54534.html]

The ISF Standard of Good Practice for Information Security (SoGP)
[https://www.securityforum.org/tool/the-isf-standard-good-practice-information-security-2018/]

Defining risk tolerance levels

Risk assessment is about identifying the potential consequences of different incidents or scenarios, and assessing how likely or easy it is that an unwanted incident occurs. It is the company's management that determines the degree of risk the company is willing to take in different scenarios. This is called risk tolerance. This tolerance level provides guidance on what measures and resources need to be put in place to ensure that the software does not exceed the defined level of acceptable risk.

In terms of security, the tolerance level is defined individually for different "security scenarios". Examples of such security scenarios could include accidental alteration of personal data, unauthorised disclosure of personal data, and a lack of accessibility that could significantly affect life and health.

In terms of data protection, the tolerance level is defined individually for different "data protection scenarios". Examples of such data protection scenarios could include the data subject losing control over his/her personal data, the data subject being subjected to discrimination based on profiling carried out by the software, or a person being re-identified from anonymised data.

Some security scenarios and data protection scenarios will have zero tolerance for risk, while for others, the company may be willing to take a certain degree of risk. Management must set acceptable tolerance levels, i.e. risk appetite, for both data protection and security.

The company's risk appetite may be documented by defining tolerance levels for data protection or security in different scenarios, often in a reference table that can be reused for other risk assessments.

Security Risk Assessment

A risk assessment begins with mapping values that should be secured. The data protection regulation defines personal data as a value.

A threat assessment should be carried out to identify which actors could be interested in the values, and which attack vectors different threat actors use. An evaluation is then carried out to determine which values are vulnerable to any given threat. Information security standards can help to detect vulnerabilities, thus also identifying the requirements that need to be established for data protection and security.

The result of the risk assessment should be assessed against the security tolerance level. If the risk level is higher than the pre-determined level of acceptable risk, measures must be implemented to mitigate the risk. It is also necessary to determine who will be responsible for the measure, and to set a deadline for implementation.

Data protection impact assessment

The purpose of a data protection impact assessment is to assess the impact an envisaged software or processing operation may have on the protection of personal data. It is to ensure that the software does not infringe on the data subject's fundamental rights. The processing should, for example, be lawful, fair, and transparent. For certain types of processing of personal data it is required to carry out a data protection impact assessment:

1. In the case of a systematic and extensive evaluation of personal aspects relating to natural persons which is based on automated decision-making, including profiling, and on which decisions are based that produce legal effects concerning the natural person or similarly significantly affect the natural person,
2. when processing sensitive personal data on a large scale, or
3. a systematic monitoring of a publicly accessible area on a large scale.

If you are in any doubt, we recommend carrying out a data protection impact assessment. The assessment shall contain at least:

- A systematic description of the envisaged processing operations and the purposes of the processing, including, where applicable, the legitimate interest pursued by the controller,
- an assessment of the necessity and proportionality of the processing in relation to the purposes,
- an assessment of the risks to the rights and freedoms of the data subjects, and
- the measures envisaged to address the risks, including safeguards, security measures and mechanisms to ensure the protection of personal data and to demonstrate compliance with the data protection regulation taking into account the rights and legitimate interests of data subjects and other persons concerned.

In cases where a data protection impact assessment indicates that the processing would result in a high risk in the absence of measures taken to mitigate the risk, the data protection regulation requires that you contact the Data Protection Authority for a prior consultation.

We recommend The Article 29 Data Protection Working Party's Guidelines on Data Protection Impact Assessment (DPIA)
[http://ec.europa.eu/newsroom/document.cfm?doc_id=47711]

Download

Example of a checklist for setting requirements (pdf)
[/globalassets/global/english/guidelines/privacy-by-design/checklist-requirements.pdf]

Design

During this activity, you must ensure that requirements for data protection and information security are reflected in the design. The requirements identified during the requirements activity must be met, and requirements for the design must be defined.

It is important to take into account the existence of threat actors that may attempt to obtain and gain access to personal data. To reduce the attack surface, it must be analysed, and the software modelled and designed to ensure a robust end product.

The target group for this activity is primarily architects, secondarily data protection officers, security advisors, and developers.

Design requirements accurately and comprehensively describe how the characteristics of each piece of software should be developed. The requirements need to describe how functionality can be implemented and distributed in a safe and secure manner. An example could be an identity and access management system where the user can see what he or she has consented to, his or her security settings and configurations, password management, and how the login process works. We have split the design requirements into two categories, data oriented and process oriented requirements, as recommended by the European Union Agency for Network and Information Security (ENISA) [https://www.enisa.europa.eu/publications/privacy-and-data-protection-by-design] .

Data oriented design requirements

1. Minimise and limit – The amount of personal data collected and processed should be limited to what is lawful and what is strictly necessary. The data shall be deleted when storage is no longer required for the purposes. A good rule of thumb is “Select before you collect”.
2. Hide and protect – Personal data, and their interrelationships, should not be communicated, processed, or stored in plain view. By hiding directly identifiable personal data from plain view, the risk of abuse and the scope of potential incidents is significantly reduced. Examples include pseudonymisation, encryption, and aggregation of personal data.
3. Separate – By separating the processing or storage of several sources of personal data that belong to the same person, the possibility of creating complete profiles of one person is reduced. For example, personal data can, based on their purpose, be stored in separate databases, units, components and areas. Separation is also an effective means to achieve purpose limitation and to avoid linkability between different data sets. A fixed time should be set for automatic deletion in tables containing personal data. This can be done by access control to tables containing personal data, splitting database tables, and distinguishing between components, units and areas with a high level of trust and those with a lower level of trust.
4. Aggregate – In order to safeguard the protection of the data subject’s rights, personal data should be collected and processed with as much aggregation as possible. You should avoid detailed personal data as much as possible, within the limitations of what can still have business value, and what is relevant to the purpose of collection and use. Examples include reducing the detail and sensitivity of personal data concerning individuals, and by removing unnecessary or excessive information whenever possible. To illustrate general trends or values, you can combine statistical data about large numbers of people without identifying individuals.
5. Data protection by default – All settings should, by default, be configured to the most privacy-friendly setting. The user should make a conscious choice to change any setting that would result in a less privacy-friendly configuration. When installing an app, the default configuration should be that the app does not track the user’s location, or share the user’s data with others. If the user wishes to use such features, he or she must actively choose to change the settings.

Process oriented design requirements

1. Inform – The software should be designed and configured so that the data subject is sufficiently informed on how the software works and how personal data is processed. When profiling or automated decision-making of personal data are being conducted the data subject should be informed on how it is

being done. It is important to remember that special requirements apply if the software is aimed at children. For example, the software should, using clear and plain language, provide information about what personal data will be used for. Images, icons, and symbols may be used to make the information clearer and more intelligible. Animation, video, and sound may be good tools for customising information to the user's level of understanding.
2. Control – The data subject has the right to control their own personal data. This includes a right to access, update, and/or delete their own data. Where automated decision-making is taking place, or decisions are being made without human intervention, the data subject can demand manual processing. The software should be designed so that the data subject can exercise these rights as easily as possible. One example might be that the user could use a menu or a separate page within the program to give and withdraw consent, allow to view, block, update, and delete their own personal data.
3. Enforce – The software should be designed so that it may facilitate documenting how it safeguards the data subject's rights. The documentation should cover accountability and how the data protection regulation is enforced. It should be available for audits and inspections of the processing. This also includes artificial intelligence, profiling, and automated decision-making. One example would be to have a data protection or privacy policy describing how the software ensures the enforceability of the data subject's rights, how you comply with data protection regulation, and what technical measures are in place to protect personal data. Technical measures might include access control and encryption.
4. Demonstrate – The controller must be able to document compliance with the data protection regulation and security of processing. The software must be designed and developed so that the controller can document and demonstrate how the requirements of the data protection regulation have been implemented. Examples include documentation demonstrating that the software has been developed using a methodology that ensures data protection by design and information security (SSDLC - Secure Software Development Life Cycle), reports from security audits, vulnerability scanning, security tests such as penetration testing, and reports on exercising data breach management.

Reduce opportunities to exploit vulnerabilities

Analyse the attack surface of the designed software to reduce attack vectors and opportunities to exploit weak points and vulnerabilities. In a design review, both communication and data flow, input and output should be analysed through the eyes of an attacker. You should also always investigate whether the same type of information is being collected or stored in multiple locations (duplicated functionality) and assess whether this functionality can be simplified. By keeping the design simple, and eliminating unnecessary features and complexity, the likelihood of errors will be reduced.

In this sort of analysis, you should use the assessments of both security risks and data protection impacts that were completed during the requirement activity. If a vulnerability is revealed, mitigating measures must be implemented to achieve acceptable risk level for data protection and security. The tolerance/risk level should be set in the requirement activity. Analysis and reduction of the attack surface should be documented.

Threat modelling Threat modelling involves the analysis of components, access points, data flow, and process flow within the software. Those involved should analyse how someone could misuse the software in different scenarios. You should review each scenario to see how the design can be improved to avoid any threats that are identified. This is done by implementing vulnerability-reducing measures, resulting in a more robust end product.

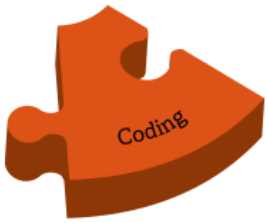
As a follow-up, you should carry out a risk assessment of any vulnerabilities that remain, and which must be mitigated using other measures. These vulnerabilities should be entered into a risk log.

Download

Example of a checklist for content in the design activity (pdf)
[/globalassets/global/english/guidelines/privacy-by-design/checklist-design.pdf]

Coding

This activity will enable developers to write secure code by implementing the requirements for data protection and security.



It is important that the company choose a secure and common methodology, both for coding and for enabling the developers to detect and remove vulnerabilities from the code. Automated code analysis tools should be introduced, and the company must have established procedures for static code analysis and code review.

The target group for this activity is primarily developers, secondarily data protection officers, security advisors, and testers.

Use approved tools and frameworks

To ensure consistent practice, a list of approved and permitted tools, processes, and frameworks for software development and roll-out must be defined and documented. In addition, it must be clear what the different tools may be used for. This entails describing approved tools and associated security features that can help to automate and enforce security procedures in the coding. The list should also include which supporting components, and third-party components and development tools, are permitted for use during development. Tools and supporting components should be risk-assessed and analysed for vulnerabilities. The list should be agreed upon and approved by the security advisor, and must be updated regularly. It should be a goal to use the latest versions of approved tools to take advantage of the opportunities provided by new security features. For example, the list may include the approved encryption technology and cryptographic key length. The list should be updated regularly with the most recognized and up-to-date algorithms and methods, and what key length is deemed sufficient.

Software today is often composed of several collaborating services. This means that a significantly greater number of programming languages, libraries, and frameworks are used in software development now than in the past. Code, libraries, and infrastructure are often merged into more static containers. This is why it is so important for regular scanning for vulnerabilities to be set up in all parts of the code, both in the underlying libraries and in the container setup. Examples of such tools can be found on GitHub [https://github.com/coreos/clair] and Docker Security Scanning [https://docs.docker.com/docker-cloud/builds/image-scan/] .

Disable unsafe functions and modules

Many functions, APIs, third-party libraries and modules can be unsafe to use based on current threat levels. An analysis should be performed on all functions, APIs, third-party libraries and modules used during software development. Those that are unsafe should be forbidden, while those that are outdated, or contain known vulnerabilities, should be updated. When a blacklist is available, the code should be checked (including inherited code) to replace blacklisted features with safer alternatives. This can be done using code scanning tools. In addition, the code should be checked to deactivate unnecessary tracking, logging, and collection of personal data. For example, unsafe functions and modules can in some cases be handled by tools such as OWASP Dependency Check [https://www.owasp.org/index.php/OWASP_Dependency_Check] .

Static code analysis and code review

Static code analysis and code review should be performed on a regular basis. Static code analysis ensures that guidelines for secure coding are being followed and can be measured to ensure controls are working. You should use automated code analysis and code review tools as much as possible. Additionally, the code should be manually reviewed to ensure that any weaknesses that could lead to improper use or leakage of personal data are caught. For example, it may be difficult to identify patterns because data alone does not necessarily constitute personal data, but connections between different types of data can provide personal information.

In order to ensure data protection, it is important to map where in the software personal data is stored. A review of the code should particularly examine where personal data is written. A common weakness is to write personal data in application logs with insufficient security.

Download

Example of a checklist for content in the coding activity (pdf) [/globalassets/global/english/guidelines/privacy-by-design/checklist_coding.pdf]

Testing

During this activity, testers check that the requirements for data protection and information security defined during the Requirements and Design activities have been implemented as planned, as well as verify that the requirements have been properly met.

The software must also be tested for vulnerabilities. This is done using dynamic testing, fuzz testing, and penetration testing. It is important to review the attack surface to verify that attack vectors uncovered during the Design activity, and potential new attack vectors introduced during coding, are handled.

The target audience for this activity is primarily testers. Data protection officers and security advisors, developers, and project leaders may also be involved in testing.

Test that requirements for data protection and security have been implemented

Tests should be carried out to ensure that requirements for data protection and security are met through design and coding, and that the requirements have been correctly implemented within the software. The checklist prepared during the Requirements activity should be used to verify that all components required to meet the requirements are included in the software. The verification should, where applicable, include new components introduced later in the development process, during design and coding. For testing, synthetic personal data should be used.

Security testing

Security testing involves comprehensive testing of the software to discover vulnerabilities and to ensure that the code adequately safeguards security and data protection. We recommend establishing procedures for automatic execution of test sets, which should be run each time software is built.

Dynamic testing

Test functionality in running code by using tools or manual reviews to analyse how the software behaves in relation to different user permissions and in the cases of critical security failures. The testing will ensure that users only have access to the information and functionality that they are authorised for. It is important to verify that attempts to gain unauthorised information are logged as security breaches. Examples of known vulnerabilities that should be investigated while running the software are Cross-site scripting and SQL injection. Session management, cookie use, and access control should also be tested to ensure that personal data cannot be retrieved by users who are not logged in.

Fuzz testing

This sort of testing is conducted by intentionally triggering errors in the software. This can be done using tools that send random and malformed data (incorrect input values) to all possible input fields in the software, either manually or using intelligent tools that analyse vulnerabilities in web applications. If the software has multiple interfaces, efforts should be made to test each of them. Use tools that can analyse output and apply security contexts, such as web proxies with the capacity to scan for vulnerabilities in their own software.

Penetration testing/vulnerability analysis

In order to detect vulnerabilities, penetration testing or vulnerability analysis should be carried out before release, and at regular intervals. Such a security test must be a legal and authorised attempt to find, exploit, and detect vulnerabilities. Following the test, measures to make the software more robust should be implemented. Because knowledge can be personnel-dependent, you should regularly rotate those responsible for testing. Furthermore, it is advisable to use security advisors to analyse the results.

Threat model and attack surface review

As software can differ from the functional and technical specifications defined during the Requirement and Design activities, both the threat model and the attack surface should be reviewed once the software is complete for release. This review should verify that new attack vectors that may have been introduced during coding have been identified and managed, and that the threat model is being reviewed against newly developed software. As the requirements may have changed during the development process, data protection impacts must be reviewed again. For example, verify that all personal data has a legal basis for processing, is necessary and is adequately protected.

Download

Example of a checklist for content during the testing activity (pdf) [/globalassets/global/english/guidelines/privacy-by-design/checklist_testing.pdf]

Release

During this activity, the software is prepared for release. This includes planning for how the organisation effectively can handle incidents that may arise after release, as well as procedures for updating the software. A comprehensive and final security review should be done before the software is released.

In organisations where software releases are frequent, an incident response plan should be established before the initial release, while security reviews should be carried out upon each release. It is important to archive all relevant data from the development process.

The target group for this activity is primarily project leaders, security advisors, and data protection officers. When preparing an incident response plan, those responsible for incident response handling should contribute, while testers should contribute to the security review.

Incident response plan

The organisation should establish a plan for managing incidents relating to the software. The plan should contain defined resources and a contact point or response centre that can manage incidents. The plan should contain relevant contact information for support and escalation, including contact information for the organisation's Data Protection Officer. The plan should also contain an overview of how incidents in code inherited from third-parties should be managed. The response time should be defined based on the importance of the software. For example, software related to emergency care in the health sector will probably require that a 24-hour response centre be available 365 days a year. It is important to consider which communication channels should be used to report incidents. For example, do the channels adequately safeguard the security of the contents of messages, and the privacy of those making the reports?

The plan should define what comprises an incident and its life cycle. One example could be detection, analysis, reporting, handling, and normalising. Furthermore, the plan should describe the configuration and handling of logs, including guidelines imposed by the data protection regulation. The plan should also contain procedures for evaluating incident response ("lessons learned"). It should also briefly describe what consequences these experiences may have for the development process, the business, and those affected.

If the incident includes a breach of confidentiality, integrity, or availability relating to personal data, the controller must be able to notify the Data Protection Authority within 72 hours, and the data subjects should be notified immediately if it likely to result in high risk to the rights and freedoms of the natural person.

The plan should contain a regime for patching the software that includes policy and follow-up. This also applies to third-party code. The plan should be updated regularly, and the company should define which triggers require updating.

Full security review of the software

The review should be based on previous reviews during the development process, and be included in the control gates that must be carried out prior to release. All requirements, analyses, and assessments carried out during the development process must be reviewed again, to uncover any deviations. Different groups of experts should be involved in each security review, to ensure the best possible review of different scenarios and consequences, as well as the best possible quality of any measures implemented as a result.

Release approval Data Protection Officer and the security advisor should verify that all the data protection and security requirements have been implemented and are functioning as intended. The organisation must define who is authorised to make the final approval to release the software.

All relevant data and documentation from the entire development process should be archived. This is important for performing support tasks, and will help to reduce long-term costs related to software management and maintenance. Archiving is also important for customers and supervisory authorities.

Download

Example of a checklist for content during the release activity (pdf) [/globalassets/global/english/guidelines/privacy-by-design/checklist-release.pdf]

Maintenance

The most important element of this activity is that the organisation has implemented a plan for incident response handling (prepared during the release activity) and follows it.

The organisation must be prepared to handle incidents, security breaches, and attacks that can result in breaches of confidentiality, integrity, or availability relating to personal data. It should have a response centre that can handle incidents and deliver updates, guidelines, and information to users and data subjects.

The target group for this activity is primarily the response incident team, security advisors, and data protection officers, as well as maintenance, service, and operation staff.

Handling incidents and data breaches

The organisation should operate an incident response plan. When critical incidents occur, it is important to remain calm and analyse the incident in a comprehensive manner. Note that the nature of the incident may result in changes to how the plan is being executed. The response incident team should know whom to contact when necessary, and who is responsible for building, testing, and installing updates. The response incident team should also know which priorities apply, as well as exactly what they can and should do in the event of a crisis. In order to achieve this, staff requires periodic incident response training. For more information on how to do incident response training, see the Agency for Public Management and eGovernment's (Difi) guidelines for planning and conducting ICT drills.

Incidents should be reported to a defined point of contact or response incident centre that handles internal and external incidents. Incidents should be reported via the defined channels described in the incident response plan developed during the release activity. Users of the software should be encouraged to report errors, vulnerabilities, and data breaches, so that the software can be continually improved and further developed. Incident assessments should also follow this plan.

Maintenance, service and operation of the software

Follow the organisation's procedures for maintenance, service and operation of the software. This includes procedures for a continuous safeguarding of data protection and security. Examples of such procedures are regular security testing, vulnerability analysis, and penetration testing of software, infrastructure and network. The procedures should include error debugging, performance improvements, updates, and patching, of both the software and third-party components. It is important to define what can and what should be logged. The company must also have the capability to regularly secure, monitor, and handle incidents in the logs. Note that personal data being logged one place is often exported to other applications and may become available to more people than those who are authorised (privilege escalation).

Conduct regular external and internal audits to document compliance with relevant regulations.

The company should have a management system for data protection and information security that includes procurement, maintenance, service and operation. The management system should be established in accordance with recognised frameworks, such as:

ISO 27001
[https://www.iso.org/isoiec-27001-information-security.html]

The ISF Standard of Good Practice of Information Security (SoGP)
[https://www.securityforum.org/tool/the-isf-standard-good-practice-information-security-2018/]

Download

Example of a checklist for setting requirements to the maintenance activity (pdf) [/globalassets/global/english/guidelines/privacy-by-design/checklist-maintenance.pdf]

1 Training

The checklist is dynamic, not exhaustive, and will be updated regularly. If you have any suggestions or comments, we would like to hear from you.

What training should be provided?

Training should be given in the following topics:

- The General Data Protection Regulation in general, particularly following themes;
  - principles of privacy, Article 5
  - the lawfulness of processing, Article 6
  - conditions for consent, Articles 7 and 8
  - processing of special categories of personal data, and criminal offences, articles 9 and 10
  - Chapter III concerning data subjects' rights, Articles 12-23
  - Chapter IV on the duties of data controllers and data handlers, Articles 24-43, particularly
    - privacy by design, and privacy by default
    - records of data processing activity
    - security of personal data
    - notification of personal data and information security breaches to the supervisory authority, and notification of data breach to the data subject
    - data protection impact assessment and prior consultation
    - data protection officer, appointment, job descriptions, overview of tasks
    - codes of conduct and certification
- laws and regulations related to the subject area of the software to be developed (e.g. patient record law, Privacy and Electronic Communications Regulation (ePrivacy), ICT regulation)
- mandatory business / sector / industry requirements and code of conduct
- the organisation's own information security requirements and guidelines
- the organisation's own internal security protocols
- roles and organization in the organisation relating to privacy and information security
- Information Security Framework (e.g. ISO27001, Standard of Good Practice (SoGP))
- Framework for software development (e.g. Microsoft Security Development Lifecycle (SDL), ISO27034
- security testing (e.g. OWASP Top 10, OWASP Testing Guide, OWASP ASVS walkthrough)
- threat and risk assessment (e.g. STRIDE, DREAD, Microsoft Threat Modelling Tool) documentation requirements

Who should receive training?

- All employees should have a basic understanding of privacy and information security.

- Management should be competent in how to assess the impact and consequence of privacy implications, risk assessment, the responsibilities of management, and handling of risks relating to privacy and information security.
- Project leaders should be competent in the topics of data protection by design and by default and information security by design.
- Developers should be competent in the topics of secure coding, and privacy and security by design.
- Architects should be competent in the topics of secure architecture, data protection by design and by default and security by design.
- Testers should be competent in the topics of security testing, data protection by design and by default and security by design.
- Suppliers should be competent in the topics of secure maintenance, service and operation, data protection by design and by default and security measures, data processing agreements, incident response handling, and emergency response. The suppliers should have readable, standardized and updated security documentation to be in compliance with GDPR.

When training should be given?

- at the start of employment
- with updates at regular intervals
- at the start of a (development) project

Example on how training should be carried out:

- through differentiated education programs at different detail levels, ranging from the basic skills (minimum mandatory skills) to specialised and/or in-depth knowledge (this can be for dedicated employees)
- using different educational techniques and tools (such as classroom training, course materials, workshops, e-learning, competitions, one-on-one discussion, certifications, courses, metaphors (easy-to-remember cartoon shorts such as "bobby tables" for example https://xkcd.com/327/), one-pagers, movies, rewards, etc.
- by updating employees based on results from internal assessments
- by updating employees based on results from penetration testing and vulnerability assessments
- by ensuring that employees familiarise themselves with the organisation's policies for privacy and information security policy, including the data protection by design and by default requirements
- as a regular fixed point on the agenda in developer forums and industry conferences.

Why training should be carried out?

- The General Data Protection Regulation requires that appropriate technical and organizational measures are taken to safeguard the rights and freedoms of natural persons and particular their right to the protection of personal data. Training is an organizational measure, and is a duty reflected in the General Data Protection Regulation, Articles 24, 25, 28, 32, and 39 (1) b).
- The training must have the support of management, and accordingly the organization.

2 Requirements

The checklist is dynamic, not exhaustive, and will be updated regularly. If you have any suggestions or comments, we would like to hear from you.

Requirements for software, products, applications, systems, solutions, or services must:

- fulfil the data-protection principles
- protect the data protection rights of the data subject
- fulfil the company's obligations
- ensure that that settings are by default set to the most privacy-friendly option
- ensure that the end product is robust, secure, and provides enforceability of the data subjects rights

What needs to be done before the requirements are set?

- Define the processing to be done, and establish an overview of the personal data:
  - Will personal data be processed by the software?
  - Identify the controller, and any processors and subcontractors. Processing contracts must be signed, and subcontractors must be approved by the controller.
  - What is the legal basis for the processing?
  - What is the purpose of the processing?
  - For how long does the legal basis and/or the purpose allow storage of personal data? Is it necessary to plan for automatic erasure?
  - Define the categories of personal data that is *necessary* to be processed to achieve the purpose. The processing of special categories of personal data and personal data relating to criminal convictions and offences (sensitive personal data), is generally prohibited, with some exceptions, so determine if one of these exceptions will apply. Document the full scope of data stored within the software.
  - How is transparency being achieved? Automatic notifications by the system, privacy dashboard, etc.?
  - Is personal data being transferred to a third country, or to an international organisation? Conditions apply to the transfer of personal data to third countries or international organisations, including restrictions on access, operations and storage. Where personal data is to be transferred to a third country or an international organisation, it is important to ensure that all such transfers are lawful.
- In what context will the processing take place? Is it likely that the software could be used in another context?
- Identify all requirements that apply to your business: Are there codes of conducts specific to the industry or sector? Are there any policies and requirements that can help you determine requirements for the software?
- Are there certification schemes you can, and should, aim to follow? What requirements apply in such cases?

- Has the Data Protection Authority made any administrative decisions in this field, relating to your own business or to other comparable businesses, that should be included as requirements in the software?

Requirements for data protection and information security

- If the software is working as intended without identifiable data, no identifying data must be collected.
- Data protection can be designed in using pseudonymisation techniques in the software.
    - Unnecessary identification and redundant personal data in the software result in greater risk to the user or the data subject. It also makes the software more vulnerable and attractive to actors wishing to reuse the data for other purposes.
    - The software must only use personal data as planned, and all data must be deleted when storage is no longer lawful according to the legal basis or no longer necessary to fulfil purpose.
    - Personal data must be available to those authorised to use it when necessary.
    - The software must be developed with default settings that protect the rights of data subjects and safeguards privacy.
    - The software should guide the user to the most privacy-friendly manner of use. For example, location sharing should be disabled by default, with the user able to activate it if required.

Requirements for meeting data-protection principles

The basic requirements for software being used to process personal data are:

- Lawfulness, fairness, and transparency
    - Processing of personal data in the software is only lawful if one of the following applies:
        - the data subject has given her/his consent
        - the processing is necessary for the performance of a written agreement or contract with the data subject
        - the processing is necessary for compliance of a legal obligation
        - the processing is necessary to protect vital interests of the data subject or another natural person
        - the processing is necessary to perform a task carried out in the public interest or in the exercise of official authority
        - the processing is necessary for the purposes of legitimate interests pursued by the controller (balancing of interests)
    - The software must ensure default settings that protect the rights of data subjects and safeguards privacy.
    - Processing of personal data must be predictable by the data subject, and performed with respect for the data subject's interests.
    - The software must be designed in a way that relevant aspects of personal data processing are known to the data subject, so that the persons concerned can make informed decisions or exercise their rights
    - The software must ensure that other rights, such as freedom of expression, freedom of thought, absence of discrimination and freedom of religion, are safeguarded.

    - Clear and comprehensible information must be provided to the data subject regarding the purpose of the processing of personal data, the legal basis, recipients of the information
- purpose limitation
    - The software must only collect personal data for specified, explicit and legitimate purposes.
    - The personal data must not be further processed in a manner that is incompatible with the original purposes.
- data minimisation
    - The software must only process personal data that is adequate, relevant, and limited to what is necessary in relation to the purposes for which they are processed.
- Accuracy
    - The software must ensure that all personal data is accurate and up-to-date. Incorrect data must be deleted or rectified.
- storage limitation
    - The software must ensure that it is not possible to identify the data subject for longer than is strictly necessary for the purposes for which the personal data are processed.
- integrity and confidentiality
    - The software must ensure appropriate security of the personal data.

If the software operates based on **consent:**

- The consent must be explicit (not passive), voluntary (no coercion/pressure), and informed (predictable).
- The processing that is based on consent shall be clearly distinguished from other matters
- A declaration of consent must be written in clear and plain language at the reader's level, and be intelligible and easily accessible to the user. Separate conditions apply to children's use of information society services.
- The users must be able to withdraw consent at any time and as easily as they give it.

Requirements to protect the rights of the data subject

The obligation **to provide information** differs depending on whether the personal data is obtained from the data subject or if it is obtained directly from a system or from persons other than the data subject.

When personal data is obtained from the data subject, he/she must be informed of the following:

- who the data controller is (identity and contact details)
- who the data protection officer is (if relevant)
- why their personal data is being processed (for what purpose)
- the legal basis is for processing (consent, contract, etc.)
- what the legitimate interests are, in cases where the legal justification for the processing is balancing of interests
- who the information will be shared with (recipients), including processors

- whether the information will be transferred to a third country or international organisation

When necessary **to ensure fair and transparent processing**, the data subject must be informed of the following:

- how long the personal data will be stored
- data subject's rights to access, rectify and erase personal data. That it is possible to object to and put restrictions to the processing of personal data and whether there is a right to data portability.
- hat consent can be withdrawn at any time
- whether the processing of personal data is being performed as a result of contractual requirements, or is necessary in order to enter into a contract
- whether the use of the software entails automated decision-making or profiling, in which case they should also be provided with information about the algorithms and the significance/consequences of the processing
- whether the personal data is intended to be used for purposes other than those for which it was collected, and if so, what rights and regulations apply to this processing.

When personal data is collected from <u>persons other than</u> the data subject, **information** must be provided concerning:

- which categories of personal data are being processed
- the source(s) of the personal data and whether it came from publicly accessible sources

The software must make it easy for the **data subject to exercise their rights**, such as

- the right to access their personal data, information about the processing, and other rights
- the right to rectify their personal data as quickly as possible
- the right to delete their personal data as quickly as possible, if the conditions for deletion apply ("the right to be forgotten")
- the right to restriction of processing of their personal data, if the conditions for restriction apply
- the right to data portability for their personal data, if the processing is based on consent or agreement and is carried out by automated means
- the right to object against the processing of their personal data, if the preconditions for exercising the right to object are present
- rights relating to automated individual decision-making, including profiling that may have legal consequences, or similarly significant effect for the person concerned

- If the user has requested that personal data be rectified, deleted, or limited, the data controller must inform each recipient to whom the personal data have been disclosed.
- The software must pseudonymise personal data when there is no longer any need to have identifying personal data and anonymise or delete personal data when the purpose of processing is fulfilled.

- The software must contain safeguards preventing the linking of personal data about the data subject to other personal data in other systems, or to personal data collected for other purposes.

Requirements to fulfil the organisation's obligations
When using processors:

- The controller must only use processors who provide adequate guarantees that they will implement measures ensuring compliance with the data protection regulation and ensure the protection of the rights of the data subject.
- The controller must ensure that any suppliers and subcontractors fulfil all requirements by entering into processing contracts.

To ensure security of processing of personal data, it is necessary to

- ensure **confidentiality (C)**. Personal data must be secured against unauthorised disclosure or access.
- ensure **integrity (I)**. Personal data must be secured against accidental and unlawful destruction, loss, or alteration.
- ensure **accessibility (A)**. Personal data must be available to authorised personnel who require it for their work.

Additionally, the data protection regulation requires ensuring resilience (R), and we also recommend ensuring traceability (T):

- Resilience means that software that is processing personal data must be able to resist e.g. vulnerabilities, attacks, and accidents.
- Traceability is documentation of changes made within the software and to personal data. The purpose of traceability is to manage security breaches.

The OWASP Application Security Verification Standards (ASVS) (https://www.owasp.org/index.php/Category:OWASP_Application_Security_Verification_Standard_Project), like several other security standards, contain several security requirements for use in software development. We will not repeat them here, but advise everyone to evaluate the level of security required for software and software development. We also recommend applying the requirements of a comparable level to the OWASP ASVS standards.

Norway has several regulations setting security requirements within different sectors. Each individual business must ensure that it knows which rules it is subject to when developing software.

Listed below are a number of security requirements that can be implemented, and examples of which security principle it refers to (C, I, A, R, T).

- Access control:
    - The software has access control (authorisation, authentication, and traceability):
        - Users must be identified (T)
        - Which roles should have which rights (principle of least privilege) (A)
        - It is possible to control traceability when auditing logs (T)

    - Users are granted access only to information necessary for the performance of individual tasks (principle of least privilege). (C, T)
    - Administrator privileges are given to a small number of individuals based on principle of least privilege. (C, I, A, R, T)
    - The data subjects have access to their own personal data. (I, A)
    - Passwords are securely handled, and the software requires strong passwords. (C, I, A)
    - The software supports and requires strong authentication (such as two-factor authentication) where necessary (e.g., users may be encouraged to use it, while it is required of administrators and users with access to personal data requiring protection or personal data on multiple subjects). (C, I, T)
    - The software must monitor if and when anyone attempts to gain unauthorised access. (C, I, R, T)
    - The software must restrict access by third parties, and limit what a third party may access (e.g., restrict access to specified IP addresses or provide temporary and limited access). (C, I, R)
- The software must ensure that there is appropriate and sufficient information security during the storage and communication of data. Encryption can help to achieve this. When using encryption, widespread and recognised algorithms and methods must be used at all times, with a sufficient key length. Minimum requirements must be set for administration, specifying how often security algorithms must be reviewed and updated
    - at endpoints (PC, laptop, telephone, tablet) (C, R)
    - upon remote access (C, R)
    - upon transfer and storage via cloud services (C, R)
    - for backup and security copies, and units containing backup data (C, A, R)
- The software must protect the integrity of data and be able to detect changes in files, servers, and networks, by (I)
    - comparing hash values and checksums
    - limiting write access
    - regular integrity checks
    - setting reference values (min/max)
- The software must ensure that personal data is available when necessary through (A)
    - Redundancy
    - contingency plans
    - incident management
    - the software must be able to restore availability and access to personal data in a timely manner in the event of a physical or technical incident
- The software must be resilient. It must (R)
    - be secured against known security holes and vulnerabilities
    - be correctly configured
    - ensure segmentation of stored data, systems, processors, and networks
    - ensure that third party software and patches are kept up-to-date
    - be capable of receiving notifications from users and others about vulnerabilities in the software, and of ensuring that they are managed and taken seriously
    - ensure the secure destruction of media that process personal data
- The software must allow changes to be traced and enable management of security breaches by (T)

- documenting software and procedures
- logging configuration changes, processes, activities, and incidents
- access control to logs based on the principle of least privilege and only when access is specifically required
- deleting or anonymising logs after a given deadline
- not storing logs for longer than necessary

Acceptable level for risk for data protection and information security

- Establish tolerance levels for data protection and for information security. The purpose of setting tolerance levels is to define acceptable levels of risk for security and data protection in the software. These must be based on established and accepted supporting tables and tolerance limits.
- Define individual acceptable levels for risk for data protection and information security. Methodology can be reused.
- Auxiliary tables can categorise criticality levels e.g., Critical, High, Moderate, and Low. Enter variables for each category, and define the acceptable level of risk.
- Examples of categories that must have **acceptable levels for risk for data protection**:
  - The data subject must have control of their personal data.
  - The data subject must not loose his/her rights or freedoms.
  - The data subject must not be profiled or discriminated.
  - The data subject must not be subject to identity theft or fraud.
  - The data subject must not suffer financial loss.
  - The data subject must not suffer loss of reputation.
  - In cases of pseudonymisation, it must not be possible to trace back to the original identity without authorisation.
  - Confidentiality breaches of protected data must not occur.
- Examples of categories that must have **acceptable level for risk for security**:
  - There must be no accidental or unlawful destruction, loss, or alteration of personal data.
  - There must be no unauthorised disclosure of, or access to, personal data.
  - Personal data must be secured with regard to confidentiality, integrity, accessibility, and resilience of the software.
  - Personal data must be pseudonymised as quickly as possible, and encrypted.
  - It must be possible to restore availability and access to personal data in a timely manner in the event of a physical or technical incident.
  - There must be a process for regularly testing, assessing and evaluating the effectiveness of measures for ensuring the security of processing.

EXAMPLE:
The company management has set the acceptable level for risk for the category "Damage to life and health" at 'Low' criticality. The red text (see table below) is beyond the company's acceptable level for risk. If information is compromised, there is a lack of information integrity, or a lack of access to information leading to 'Medium', 'High', or 'Critical' criticality, this will be unacceptable for the company's management, and measures must be implemented. Examples of situations in which information lacks

sufficient integrity resulting in 'High' criticality might include the provision of incorrect information about the brakes on a car, or about a person's blood type.

Criticality level	**Category – example:** Damage to data protection rights
Critical	Death or injury resulting in permanent disability
High	Damadge to reputation or social exclusion
Medium	Personal exposure resulting in sick leave
Low	No exposure

Microsoft SDL (https://www.microsoft.com/en-us/sdl/) and ISO 27034-x (https://www.iso.org/standard/44378.html) also provide examples on how to find acceptable level of risk.

Data protection impact assessment and security risk assessment

- The purpose of a data protection impact assessment is to assess the impact an envisaged software or processing operation may have on the protection of personal data. It is to ensure that the software does not infringe on the data subject's fundamental rights. The processing should, for example, be lawful, fair, and predictable. In certain types of processing of personal data it is required to carry out a data protection impact assessment:
  - In the case of a systematic and extensive evaluation of personal aspects relating to natural persons which is based on automated processing, including profiling, and on which decisions are based that produce legal effects concerning the natural person or similarly significantly affect the natural person,
  - when processing sensitive personal data on a large scale, or
  - a systematic monitoring of a publicly accessible area on a large scale.

  If you are in doubt about your obligation, we recommend you to carry out a data protection impact assessment.

- The assessment shall contain at least:
  - A systematic description of the envisaged processing operations and the purposes of the processing, including, where applicable, the legitimate interest pursued by the controller,
  - an assessment of the necessity and proportionality of the processing in relation to the purposes,
  - an assessment of the risks to the rights and freedoms of the data subjects, and
  - the measures envisaged to address the risks, including safeguards, security measures and mechanisms to ensure the protection of personal data and to demonstrate compliance with the data protection regulation taking into account the rights and legitimate interests of data subjects and other persons concerned.
- In cases where a data protection impact assessment indicates that the processing would result in a high risk in the absence of measures taken to mitigate the risk, the data

protection regulation requires that you contact the Data Protection Authority for a prior consultation.

- Carry out a technical risk assessment of the software's information security:
    - Such an assessment should reveal vulnerabilities and deficiencies in the security of the software, thus contributing to the provision of adequate security requirements.
    - Be sure to test, assess and evaluate the effectiveness of the security measures.
    - Please examine existing standards and examples of how to implement risk analysis, such as ISO27005 (https://www.iso.org/standard/56742.html), the Data Protection Authority guidelines on risk assessment (https://www.datatilsynet.no/regelverk-og-skjema/behandle-personopplysninger/risikovurdering/), and the guidelines for internal control and information security issued by the Agency for Public Management and eGovernment (Difi) (http://internkontroll.infosikkerhet.difi.no/).

3 Design

The checklist is dynamic, not exhaustive, and will be updated regularly. If you have any suggestions or comments, we would like to hear from you.

Data oriented design requirements

Minimise and limit. The amount of personal data collected and processed should be limited to what is lawful and what is strictly necessary. The data shall be deleted when storage is no longer required for the purposes. Follow the principle "*Select before you collect*".

Examples:

- Review the Data Protection Impact Assessment (DPIA).
- Be sure that the need for personal data matches the amount and scope of data collected. Do not collect more information than necessary. Limit the amount of information processed on units or in areas with lower trust and security assurance.
- Avoid, limit, or minimise the need to collect and process sensitive personal data.
- Limit and minimise the exposure of unnecessary functionality and personal data in the user interface. Consider, for example, whether it is necessary to store directly identifying information in the software itself, or if pseudonymised information is sufficient.

Hide and protect. Personal and their interrelationships, should not be communicated, processed, or stored in plain view. By hiding directly identifiable personal data from plain view, the risk of abuse and the scope of potential incidents is significantly reduced. Examples include pseudonymisation, encryption, and aggregation of personal data.

Examples:

- Conduct *threat modelling* and "*attack surface analysis*" when designing software.
- Use encryption mechanisms for secure transfer, communication, processing and storage. This is particularly important if there is a need to transfer personal data over uncontrolled areas and networks.
- Anonymise or pseudonymise personal data wherever possible.
- Avoid unnecessary exposure to communication patterns and connections (such as APIs, feeds, gateways, login interfaces, etc.).

Separate. Personal data should be separated from other data when possible. Data should be stored in separate databases, entities, and areas for each purpose and process. By separating the processing or storage of several sources of personal data that belong to the same person, the possibility of creating complete profiles of one person is reduced. Separation is also an effective means to achieve purpose limitation and to avoid linkability between different data sets. Tables containing personal data should have shorter storage times and a deadline for automatic deletion, while tables without personal data can be stored for longer. Measures to achieve this can include access control to tables, splitting database tables, distinguishing between components, units and areas with a high level of trust and those with a lower level of trust, and dividing access to specific areas according to necessity.

Examples:

- Separate sensitive personal data from less sensitive personal data (in the database, access to sites, for clients and units, etc).
- Separate tables in the database and associated access rights to tables and areas, according to job necessity (the principle of least privilege).
- Keep less highly trusted units and areas separate from more highly trusted units.
- Rows in tables should be hard to link to one another.

Aggregate. Personal data should be gathered and processed in as aggregated manner as possible to ensure the enforceability of the data subject's rights, without prejudice to the business value and purpose of the collection and use.

Examples:

- Reduce the use of detailed and sensitive personal data.
- Remove unnecessary and excessive information. For example,
    - "Raise" the measures of time by using weeks instead of days or hours
    - use counties or regions, rather than street addresses, when allocating people or units
    - use groupings rather than individuals
- Use anonymising techniques whenever possible.

Data protection by default. All settings should, by default, be configured in the most privacy-friendly setting. The user should have to make a conscious choice to change any settings that would result in a less privacy-friendly configuration. For example, it should be the user's choice to share more data with others.

Examples:

- All privacy-friendly configurations need to be on by default.
- Device tracking should be disabled by default.
- Bluetooth should be disabled by default.
- Tracking from one website to another should be disabled by default.
- By default there should be no reuse of information about which websites the data subject has visited.

Process oriented design requirements

Inform. The software should be designed and configured so that the data subject is sufficiently informed, on how the software works and how personal data is processed. When profiling or automated processing of personal data are being conducted the data subject should be informed on how it is being done. It is important to remember that special requirements apply if the software is aimed at children.

Examples:
A website containing information about the software should be established and made available to the data subjects before they begin using the software:

- Provide a point of contact, contact form, or similar, that the data subject can use to request information on
    - the purpose and need for data collection

    - the security of the software (how the data is protected)
    - use of subcontractors, and possible sharing of data with subcontractors
- Use multiple approaches and channels to ensure that you reach all users and user groups.
- Use simple, comprehensible language.
- Use multiple languages if necessary.
- Enrich and vary with photographs, icons, audio, video, etc to target users.

Control. The data subject has the right to control their own personal data. This includes requesting access to view, update, and/or delete their own data. Where automated processing is taking place, or decisions are being made without human intervention, the data subject can demand manual processing. The software should be designed so that the data subject can exercise these rights as easily as possible.

Examples:
Establish and enable the ability to:

- maintain an overview of which elements of data processing are necessary to fulfil the contract, and which are subject to voluntary consent
- allow the data subject to provide consent on an information page with a checkbox before using the software
- withdraw consent via a menu within the software. Keep in mind that collection of personal data must cease if consent is withdrawn
- give access for the rectification, blocking, or deletion of personal data, e.g. by allowing collected data to be viewed directly by the data subject within the software
- ensure the permanent deletion of personal data in the database and wherever else it is stored (e.g., backup). The information can also be exported to a file or paper version for manual review with a corresponding procedure allowing the data subject to correct, block, or delete data (within 30 days)
- terminate a contract/agreement, install, uninstall, enable and disable an application, service, technical component, or system, by using functionality within a menu, on a dedicated page, or manually using a form
- submit questions or complaints relating to data protection and security. Alternatively, the information can be made available on a website with contact information connected to a staffed communications channel for handling enquiries (in this case, a documented procedure must be established)
- object to profiling by enabling users to make a conscious choice to reject profiling and the redistribution of personal data. This can, for example, be done via a menu with a checkbox and a flag to be stored in the database, or carried out manually via a staffed communications channel
- define requirements for openness and information on how automated decisions are made, as well as the ability for data subjects to demand manual processing. This is described in the system documentation, and should be made available on an information page within the software.

Enforce. The software should be designed so that it can document that how it ensures the enforceability of the data subject's rights. The documentation should cover accountability and how the data protection regulation is enforced. It should be available for audits and

inspections of the processing. This also includes artificial intelligence, profiling, and automated processing.

Examples:

- The software must apply the highest privacy settings by default:
    - Settings should be presented in a menu where the data subject must make a conscious choice to actively "change" to less privacy-friendly settings.
    - If the software at a later date has to be changed to a less privacy-friendly setting, the data subject must be clearly informed of the change, and may consent to it by clicking a checkbox after notification of the change has been received/read.
- The software must meet the dataportability requirement:
    - There must be a way for the data subject to request the disclosure of their own personal data, or request that the data be transferred to another service provider, in a standardised and reusable format.
    - This right applies when processing is based on consent or a contract.
    - The data subject may request that personal data be exported and delivered in a secure manner (manually by traditional post, or another secure delivery method) to new service providers.
- Consent must require the active participation of users:
    - Consent, or change of consent, is signalled by filling out a text box or clicking a checkbox that marks a flag in the database.
    - Software aimed at young people and minors must contain features requiring consent from the parent or guardian before access is granted. This feature must ensure or confirm that they are authorised to grant this consent. Alternatively, procedures can be established to request manually documented consent from the parent or guardian.
- The data subject should be given an overview of what access (including access and changes by the data subject) has taken place, when, by whom, and what types of consent has been given. All access should be recorded in logs that can be made available to the data subject.

Demonstrate. The controller must be able to document compliance with the data protection regulation and security of processing. The software must be designed and developed so that the controller can document and demonstrate how the requirements of the data protection regulation have been implemented. Examples include documentation demonstrating that the software has been developed using a methodology that ensures data protection by design and information security (SSDLC - Secure Software Development Life Cycle), reports from security audits, vulnerability scanning, security tests such as penetration testing, and reports on practicing data breach management.

Analyse and reduce the attack surface of the software being developed

- Analyse the attack surface of the pre-designed software to reduce opportunities to exploit weaknesses and vulnerabilities in the software.
- Review designs and analyse where it is possible to receive input data, get output and where data is transported and stored.

- Check if the same type of information is being collected in multiple places (duplicate functionality), and assess whether functionality can be simplified.
- Reduce the likelihood of errors by simplifying the software and eliminating unnecessary functionality.
- Reuse the assessments of security risks and data protection impacts that were completed during the requirements phase.
- Implement vulnerability-reducing measures to achieve acceptable tolerance levels for data protection and security, if the analysis reveals that the existing levels are not satisfactory.
- Reuse the tolerance level developed during the requirements phase.
- Be sure to document the analysis and the reduction of the attack surface.

Threat modelling

- Analyse components, access points, data flow and processing of data in the software.
- Ensure that those involved in the development team analyse how the software could be misused in different scenarios.
- Review each scenario to see how the design can be improved to avoid any threats identified. This can be done by implementing vulnerability-reducing measures, resulting in stronger and more robust software.
- Perform a risk assessment of any vulnerabilities that remain and which must be mitigated using other measures. Ensure that these vulnerabilities are included in a risk log.

Examples of tools

- Design principles for good security, including:
  - The principle of least privilege
  - Defence in depth
  - Fail Securely
- Threat Modelling
- Attack-Surface-Analysis
- Use Case and Misuse-Case Modelling
- Attack-Trees
- DREAD/STRIDE
- The Data Protection Authority's guidelines on anonymisation and anonymisation techniques (https://www.datatilsynet.no/aktuelt/2015/Hvordan-anonymisere-personopplysninger-Ny-veileder/)

Why are design requirements necessary?

- Personal data must be processed lawfully and the design of the software must reflect this, cf. Article 6 of the General Data Protection Regulation. It is therefore important to be aware of the type of personal data being processed, the purpose and terms of the processing, and the compilation of personal data.
- The rights of the data subject must be reflected in the design, cf. the General Data Protection Regulation articles 12-23.

4 Coding

The checklist is dynamic, not static, and will be updated regularly. If you have any suggestions or comments, we would like to hear from you.

Possible measures for secure coding

- Create a list of approved tools and libraries
- Scanning of dependencies for known vulnerabilities or outdated versions
- Manual code review
- Static code analysis with security rules

Use approved tools and frameworks

- Establish a list of
    - approved tools with associated security features that will help to automate and enforce security procedures in the coding
    - approved supporting components
    - permitted third-party components and development tools. Avoid sharing of personal data through third-party libraries – use synthetic data instead
- Describe in the lists what the various tools and supporting components will be used for, including new security analysis, functionality, and protection. Tools and supporting components should undergo risk assessment, and be analysed for privacy and security vulnerabilities.
- Keep the lists updated according to organisation guidelines. This means that new tools and versions must be reviewed, and used wherever possible.
- Use only approved tools and supporting components from the list. Any exceptions should be documented and approved by the security officer.

Examples: Code patterns and templates for widely used/established functionality should be generalised, quality checked, and documented as current patterns/templates. Examples include how database calls should be made, and how they should be structured.

Examples of approved tools and supporting components that must be included in lists:

- Code library
- Programming language
- Version control system
- Testing tools
- Infrastructure
- Monitoring tools
- Logging server
- Third party framework and APIs

Example of this in practice:
The organisation has decided on a coding method for initial and further development. This thus applies to all projects and project managers. They use a task management system, such as Jira or Confluence. All tasks have a dedicated owner who is also responsible for the task's data protection and security. The status of all active tasks is established at a morning

meeting for the team, where challenges that have arisen relating to, e.g. data protection can be discussed. Code review must be carried out by a different developer. This is important for avoiding personnel dependencies in the code, as well as to improve the quality of the code. But, it is equally important to have an additional person review the code to verify that data protection and security is preserved. Versions will be built for testing, and a release owner will be appointed.

Invalidate unsafe functions and modules

- Unsafe functions and modules are handled by tools, such as OWASP Dependency Check.
- Disable unnecessary tracking, logging, and collection of personal data.

Static code analysis and review

- Establish business routines and/or checklists for static analysis and code review.
- Analyse and review the source code regularly, and each time it is built.
- Check data flow, storage and temporary storage of data.
- Static analysis for data protection should primarily be performed manually, as "automated" tools for reviewing code for data protection are limited. It can be difficult to identify patterns because data alone does not necessarily constitute personal data, but connections between different types of data can provide personal data.
- It is important that the person who does the work (the reviewer) has a thorough understanding of both the principles of data protection, the data subject rights and the requirements for data protection by design.
- Have different levels of scanning, such as for developers, for security advisors, and for the person responsible for product release.
- Establish guidelines for what should be scanned, and when.

Examples of tools for static code analysis:

- RIPS PHP Static Code Analysis Tool, OWASP LAPSE+ Static Code Analysis Tool, SonarQube, Checkmarx.

Examples of how to perform static code review:

- Regular (e.g. daily or weekly) scanning.
- Scanning with both Scan Master and Dev Brancher, and performing a full security scan.
- During "on build scanning", minimum security requirements should be checked by Dev Brancher or Scan Master (e.g. SQL-Injection).
- "On Demand scanning" is performed by the user's IDE, and is a full security scan.
- Use checklists for code review:
    - Example A1. Find all points with database access. Are queries with "dirty variables" being concatenated?
    - Example A4. How are users blocked from accessing other users' data?

Qualities to look for in a code analysis tool:

- It should be designed for security.
- It should support multiple levels (different programming languages and platforms).

- It must be possible to expand. The tool should be able to be expanded to include new attack and defence techniques.
- It must be useful to both security analysts and developers.
- It should support existing development processes.
- It should have value for multiple parties with ownership of the development (such as interfaces with measurements that can support decisions on release management, check costs when changes occur, and provide information needed for maintenance).

Why secure coding?

- The General Data Protection Regulation will apply when software is to be developed, cf. Article 25.
- Tools, support systems, and infrastructure should be “state of the art”; i.e., the newest and most updated version of the technology, cf. Article 25.
- It should be a fixed goal for the management to use secure and common methodologies for
    - coding
    - identifying and removing vulnerabilities in the code
    - using automated tools for code analysis
    - static code analysis and review

5 Testing

The checklist is dynamic, not static, and will be updated regularly. If you have any suggestions or comments, we would like to hear from you.

Test that the requirements for data protection and security that were specified in Requirements have in fact been implemented, and that they are correctly implemented:

- Remember that the data protection regulation also apply to development and testing environments.
- Establish a comprehensive understanding of functionality and information.
- Verify that the requirements and design components have fulfilled the security and data protection requirements. This can, for example, be done by creating standard test scenarios based on functional requirements that can be reused throughout the business.
- Compile a checklist on whether all the components needed for compliance are included. This also includes new components that may not originally have been specified when the requirements were determined. Examples:
    - All settings should be set to the most privacy-friendly option by default.
    - It must be possible to export and import the data subject's data (data portability).
    - Is data being saved in the correct place?
    - Is the data being collected necessary for the purpose of the software?
    - The data subject should be able to give consent (this applies also to children and persons subject to guardians).
    - The data subject must be able to refuse or withdraw consent.
    - Is it possible to terminate a contract/agreement, install, uninstall, activate and deactivate a program, service, technical component, or system?
    - Access control
        - for authorised parties only
        - accessible to the data subject
        - gives access to rectify, block, or delete personal data
    - It should be possible to send queries or complaints relating to data protection and security.
    - The data subject can reserve the right not to be profiled.
    - Get information about how automated decisions are made, what is collected and processed, where personal data is stored, transparency, etc.
- Test that notification procedures exist and work, that notifications are acknowledged (for example, develop a draft text that can be used in the event of an incident, including measures to prevent a similar incident from occurring again), as well as what steps have been taken to prevent any impact on the data subject.
- By testing use synthetic personal data to check correctness, lawfulness, fairness, purpose, accuracy, data minimisation, and resilience.
    - Use real data only for quality assurance when the software is implemented in the production environment, and users will be using the software. Tests on real data should be performed by professionals who are authorised to view the data, such as appropriate professionals for child welfare programs in the child welfare

sector, super users for patient journals, or super users for a school system in a school.

- Search for real personal data, and check why they are appearing. This can, for example, be done by regularly scanning for potential leaks of national identity numbers.

Security testing by challenging vulnerabilities in the software

- **Dynamic testing:** test the functionality of the running code by using tools or manual review to analyse how the software behaves with different user privileges and for critical security vulnerabilities. Testing must ensure that users only gain access to the information and functions they are authorised for. Verify that unauthorised attempts to acquire information are logged as security and personal data breaches.
    - Prefer input validation with whitelisting over blacklisting
    - Perform a vulnerability analysis on the application and network layers. The analysis should measure the effect of implemented technical and organisational security measures. Use testing tools. For example, test for default passwords, passwords stored in files, SQL injection, script injection, cross-site scripting, missing validation, unauthorised access to logs, backups, and temporary locations.
    - Test access control and actual access to users:
        - Example of a GET request: log in user 1, copy all links, log out user 1. Log in user 2, etc. Check all links from one user that are accessible for unauthorised users or other users. Test all levels of permissions, using at least two users at each access level.
        - Simple testing by non-technicians. Use a browser, for example, and check if GET requests in the browser give access to private URLs.
        - Advanced testing of technicians. Use burp suit or burp repeater, for example, to investigate POST requests and session cookies.
        - Parameters that can be iterated (e.g. ID=42, 43, 44, etc.) should be iterated and checked for access control.
        - Investigate cookie and session management, such as how login and access control work based on cookies. (If a new cookie is generated after logging in, is the session destroyed on the server side or solely deleted from the user's browser?)
- **Fuzz testing:** Fuzz testing is conducted by intentionally triggering errors in the software. This can be done using tools that send random and malformed data (incorrect input values) to all possible input fields in the software, either manually or using intelligent tools that analyse vulnerabilities in web applications. If the software has multiple interfaces, efforts should be made to test each of them. Use tools that can analyse output and apply security considerations.
- **Penetration testing/vulnerability analysis:** Perform penetration tests or vulnerability analysis on newly developed software before release, or at regular intervals, to uncover vulnerabilities. Ideally, different testers should be used every time. Security tests, such as this, must be legal and authorised attempts to find, exploit, and detect vulnerabilities. Following this, measures to make the software more robust should be implemented. Note that it may be necessary to have signed agreements in place for penetration testing.

    - This can be done both internally and externally. Assess what is most appropriate in terms of new functionality/services.
    - Use security experts to both perform testing and analyse the results.
    - Test the security features of data protection by design. Examples of vulnerabilities that may be detected are passwords, installation of Remote Controls in web browsers, clients, or servers, database dumping, data transfer, or stored data including caching, interrupting, or session and connection hijacking, exploitation of cookies, encryption cracking, exploitation of inadequate access control, etc.
- Test in multiple instances
    - Test environments, such as code review of the application (testing for vulnerabilities in the code).
    - Integration and system testing environments, such as static and dynamic tests of applications.
    - Testing in production instances, such as application fuzzing, scanning for vulnerabilities, and penetration testing of applications and infrastructure.
- Implement a regime for automatic execution of test sets before release, such as
    - security tests that are run every time an application is built or further developed. If errors occur during production,
    - similar errors can be avoided by creating an automated test for the error, and adding it to the test set.

Examples of tools that can be used for security testing:

- OWASP testing guide to OWASP's top 10 vulnerabilities, SANS/CIS 20 vulnerabilities
- White box fuzz testing
- Burp suite and burp repeater for code review
- Bug bash and "Feedback Hub".

Review the attack surface

- Verify that attack vectors discovered in the design phase have been addressed.
- Verify that new attack vectors introduced during implementation are identified and addressed.
- Review the threat model in relation to newly developed software.
- Re-assess data protection impacts that were assessed during the requirements phase, as requirements can change during the development process without being assessed.
- Pay particular attention to whether more data is being transferred than what the software requires, and if such data is lacking protection or legal basis.

Why set requirements for verification and testing?

- The data protection and security requirements that were included in the test set must be implemented, and implemented correctly, cf. Articles 5, 7, 25, 35, and 11-23.
- Incorrect input values must not contribute to breaches of confidentiality, integrity, or availability, cf. Article 32.
- The attack surface must not contain vulnerabilities that can contribute to breaches of confidentiality, integrity, or availability, cf. Article 32.

6 Release

This checklist is dynamic, not exhaustive, and will be updated regularly. If you have any suggestions or comments, we would like to hear from you

How to create a software-related incident response plan?

- Create an incident response plan of the software to be released. This should include impact assessment, measures, and continuous improvement of the software.
- Establish a contact point or response centre with its own communication channels for reporting incidents, taking into account internal and external incident reporting. For example, encouraging and having a good dialogue with "whistle-blowers" will be crucial to whether or not users will report vulnerabilities, deviations, and errors. User reports of security incidents can help increase the robustness of the software if these incidents are properly managed.
- To deal with future threats to the software, the plan must cover:
    - Contact information and communication channels
        - in the case of data protection and/or security breaches
        - for technical support
- Binding agreements that specify suitable response times for relevant suppliers
- Establish risk management procedures for the various scenarios described in the requirements phase for risk assessment of data protection and security. Who easy and likely is it that different scenarios occur, what is the impact, who should be informed, should the system be turned off immediately, what is mandatory logging and triggering of alarms, etc. Example:
    - What is the consequence if you are too proactive and shut down a system when an incident occurs?
    - Is it likely that evidence concerning alarm triggers (thresholds) is removed upon multiple login attempts by the same user from multiple IP addresses, many users from the same IP address, etc.?
    - What should happen if someone discovers sensitive personal data being stored in the wrong place? Who should they contact?
- Definition of what the plan covers, and what qualifies as an incident.
- Definition of the life cycle of a deviation, as well as procedures for detecting, analysing, reporting, handling, and normalising.
    - Detect
        - Security monitoring of servers, end points, and network should discover suspicious activity that could exploit vulnerabilities in the software, resulting in data protection and security incidents.
        - Security monitoring and alarms triggered by suspicious patterns help to provide an overview of threats, vulnerabilities, and security incidents.
    - Analyse and Verify
        - Analyse the incident, review the logs, gain an overview of what happened, and the scope of the incident.
        - Verify if suspicious activity is an actual security breach or a false positive.

        - Follow the company procedure for forensics, including who will lead the investigation, when the police or other experts will be called upon for assistance, etc.
    - Report
        - Report security breaches according to internal guidelines. Note that warning of a deviation can also alert the attackers, so be careful how you raise the alarm.
        - Report deviations that include personal data breaches of confidentiality, integrity, and availability to the Data Protection Authority *within* 72 hours.
        - Inform the data subject of deviations that include breaches of confidentiality, integrity, and access to personal data.
        - Consider sharing the resulting experiences with the industry or sector
        - Should other authorities be informed, such as the police, The Norwegian National Security Authority (NSM), The Norwegian Centre for Information Security (NorSIS), Financial Supervisory Authority of Norway (Finanstilsynet), Norwegian Board of Health Supervision (Helsetilsynet) etc.
    - Handle
        - Security incidents should be handled according to the business continuity plan. Are deviations handled differently depending on whether they have a direct impact on the customer or if the effects will be internal within the company? It should be noted that an unplanned patch can serve as a warning to potential attackers that a vulnerability exists.
        - Implement vulnerability reducing measures, such as patching, changing procedures, extended logging, regulate access, closing ports, etc.
        - Test if the implemented measures are working as intended, and are not causing new vulnerabilities.
    - Normalise
        - Restore management, operation, and maintenance to their normal state.
- Description of the configuration and handling of logs, including guidelines from the data protection regulation.
- Perform an evaluation of the incident response plan, and how the experiences can have consequences for the development process, the company, and others affected.
- Stakeholders must be identified, and a procedure should be established to describe when and how they will be informed. Notification of personal data breaches to the Data Protection Authority (within 72 hours) and communication to the data subjects.
- Recommended procedures for patching the developed software, including related software (including those from third parties). The procedures must form part of the company's overall plan.

The business continuity plan should be updated according to the requirements above. Events that could trigger updates include new services, altered criticality, changes to the emergency shutdown procedure, changes to the criticality matrix, changes to alert lists, etc.

Full security review of developed software

- The security review should be based on reviews carried out during the previous phases of the development process.

- Use different expert groups for the review, discuss different scenarios, and assess the consequences and possible measures.
- Consult with roles/persons who are directly involved in the activities (what has worked well, and what has not worked well – continuous improvement).
- A full security review should be included in the control gates and be carried out prior to release. All activity should be reviewed to reveal any deviations, and should include
    - data protection requirements
    - security requirements
    - acceptable level of risk
    - the results of the data protection impact assessment (DPIA)
    - risk assessment, including risk treatment plan and risk acceptance
    - attack surfaces
    - output from tools
    - design analysis (Are there any deviations from the planned and currently applicable design specifications?)
    - code analysis (Are manual and static analyses performed in accordance with the revealed focus areas, are the results aligned with tolerance levels?)
    - tools and third-party components (Are only authorised tools and third-party components used during the development process?)
    - dynamic testing, penetration testing or vulnerability analysis (Are the findings aligned with the defined tolerance levels?)
    - the results of penetration tests and vulnerability analysis
    - a review of the finished software against the original threat models, to uncover implementation anomalies and, if necessary, put into place measures to handle them

Approval of release and archiving

- The software must be approved before release, to ensure that data protection and security requirements are met.
- Approval should be carried out by a manager with the responsibility/authorisation to do so, with support from stakeholders in relevant disciplines
- All relevant data from the entire development process should be archived, including all specifications, source code, binary code, private symbols, data protection impact assessment (DPIA), risk assessments, documentation, business continuity plans, licenses, and terms of service for third-party software. Archiving is important for future needs of support tasks, reduce long-term software management and maintenance costs, and enabling rollback to a previous version. For example, archiving can be carried out in Escrow.
- Based on the results of a full security review, the responsible/authorised manager will decide if the software should be released. (Persons who are not responsible can provide recommendations for release).

Why set requirements for release management?

- There are requirements for incident management and business continuity. Having a plan for incident management may help to handle serious incidents properly and

efficiently, and to protect against adverse consequences from errors or accidents, cf. Articles 32, 33, and 34.

- The final security review of the software must be committed in the internal control system or the information security management system (ISMS), cf. articles 30 and 32.
- The release must be approved, and archiving of the activities carried out during development, cf. articles 30 and 32.

7 Maintenance (operation)

This checklist is dynamic, not exhaustive, and will be updated regularly. If you have any suggestions or comments, we would like to hear from you.

How to handle incidents and data breaches?

- Implement and operate a plan for incident response management (prepared during the release activity).
- Security incidents must be given high priority.
- Handle incidents and data breaches:
    - **Detect** abnormal activity, traffic, security incidents, and data breaches.
    - **Analyse/Verify** whether abnormal activity, traffic, security incidents, and data breaches are actual security breaches or false positives
    - **Report** security breaches and data breaches according to internal guidelines for incident response handling.
    - **Handle** security incidents and data breaches according to the organisation's continuity plan for restoring the **normal state** of maintenance, service and operation.
- Incident response training covering unexpected scenarios should be done periodically.

Maintenance, service and operation

- Identify and allocate roles, responsibilities, and authority.
- Handle the data subjects' rights and request related to this, such as data access, modification, deletion, data portability, consent, information, transparency, etc.
- Continuously assess the effectiveness of technical and organisational security measures for uncovering vulnerabilities.

  Examples:
    - security tests (such as vulnerability analysis and penetration testing, continuous automated health checks of software, infrastructure and network).
    - training (such as topic- and industry-specific drills, desktop, games, etc.)
    - testing and measurement of the security culture (such as campaigns, surveys to be answered, etc.)

- Metrics comparing the effect of security measures to their intended purpose.
- Data, platform, network, and software maintenance includes:
    - identifying and monitoring potential points of attack, such as applications, servers, networks, endpoints, etc.
    - error debugging, updating and patching of server and client software and third-party components
    - performance improvements
    - logging of system events and user activity for security checks
    - periodic reviews of logs to uncover security breaches

    - correction, deletion, and phasing out of data and applications, such as developers with oversight over the entire production process, and who continuously implement measures to improve IT solution
    - upgrading and phasing out of software libraries
    - tackling new security challenges and handling new vulnerabilities
    - updating and maintaining crypto algorithms and keys

- Follow NSM's measures for data attacks (https://www.nsm.stat.no/globalassets/dokumenter/veiledninger/systemteknisk-sikkerhet/s-02-ti-viktige-tiltak-mot-dataangrep.pdf).
- Update contingency and continuity plans.
- Conduct periodically training of the plans
- Conduct regular internal and external audits, to document compliance with regulations, and to eliminate data breaches.
- Periodically audit data processors using the data processing agreement and relevant auditing criteria, such as legislation, codes of conduct, internal regulations, and security frameworks.
- Check and review user's and supplier's access.
- Perform regular risk and vulnerability analyses, based on earlier risk and vulnerability analyses.
- Conduct data protection impact assessments when significant changes, or development of, software occurs.
- Establish and present the current status of privacy and security to the management.

Why impose requirements for maintenance, service and operation?

- The data controller must have a full record of processing activities relating to personal data, and data processors must keep a similar record of their actions on behalf of different data controllers, cf. Article 30.
- There are security requirements for the processing of personal data, cf. Article 32.
- There are requirements for data protection by design and by default in solutions, programs, apps, and systems that manage personal data, cf. Article 25.
- There are requirements for a data protection impact assessment upon start up or for significant changes relating to the processing of personal data, cf. Article 35.
- There are requirements to ensure the data subject's rights, cf. articles 12-23.
- There are requirements to ensure compliance with the privacy principles, cf. Article 5.
- Secure maintenance, service and operation are anchored in the organisation's management.
- The data controller is required to make use of data processors who are bound by the General Data Protection Regulations, cf. Article 28.