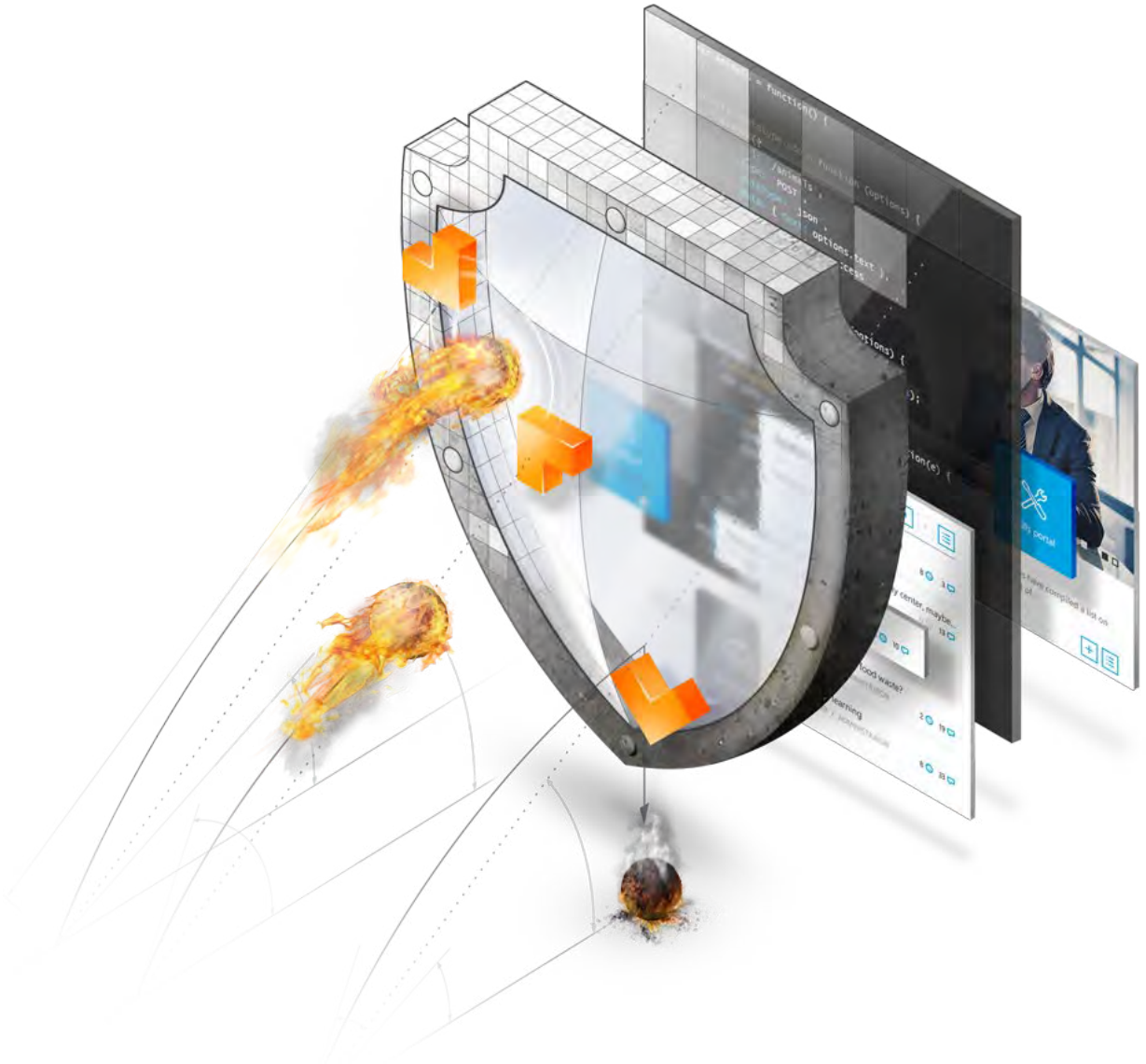
Future Processing

WHY IS SOFTWARE SECURITY IMPORTANT?



WWW.FUTURE-PROCESSING.COM

2018

WHY IS SOFTWARE SECURITY IMPORTANT?

CONTENTS

PROTECTING YOUR BUSINESS

According to the latest Symantec Internet Security Threat Report, in the last 8 years more than 7.1 billion identities were exposed in data breaches.

New classes of attacks, such as ransomware and targeted espionage attacks show new levels of cyber-criminal ambition. Additionally to potential financial losses, such attacks have an unquantifiable, negative impact on customer confidence and on brand value. The new **GDPR law** will introduce a great challenge for all organisations that process personal data. Building the software with security-in-mind approach (which includes regular testing, vulnerability assessment and penetration testing - both using automated tools, and as an expert-driven manual process) allows organisations to go beyond today's compliance requirements, enabling them to be proactive and forward-thinking. The **Ponemon Cost of Data Breach Study** shows that the global average cost of data breach is $3.62 million. The average cost for each lost or stolen records containing sensitive and confidential information was measured as $141 in the study.

The report includes data from 419 companies in 13 countries or regions. It also states that the cost of a data breach depends on the industry, indicating healthcare organisations and financial services leaks as the most costly. Businesses operating in these industries should ensure that they have a proper security programme in place.

The predictions are that the threats are set to worsen. Adoption of new technologies (IoT, Cloud Computing, mobile devices) expands the threat landscape. New bugs and vulnerabilities are discovered every day. And most of the websites serving malicious content were once legitimate websites that have been compromised by attackers.

BREACHES	2014	2015	2016
TOTAL BREACHES	1,523	1,211	1,209
BREACHES WITH MORE THAN 10 MILION IDENTITIES EXPOSED	11	13	15
TOTAL IDENTITIES EXPOSED	1,2B	564M	1,1B
AVERAGE IDENTITIES EXPOSED PER BREACH	805K	466K	927K

Source: *2017 Internet Security Threat Report* https://www.symantec.com/security-center/threat-report

SECURE DEVELOPMENT MUST BE PART OF COMPANY CULTURE

Development lifecycle is a process which guides developers on how to build software. It usually contains the following phases:

- Training
- Requirements
- Design
- Implementation
- Verification
- Release
- Response

Secure Development Lifecycle extends regular development by adding activities on top of the existing process. It ensures that security is not only considered during the testing phase, but remains a continuous concern. Correctly implemented Secure Development Lifecycle allows to build more secure software, helps in addressing compliance requirements and reduces the total cost of development.

While process implementations might vary in detail, the core is common – "shifting left", where practices and secure principles are applied in a holistic manner to the whole development process.

> *Any of these processes can be used in your organisation to decrease the number of risks connected with software development. The choice should be based on current development processes and organisational culture.*

Three steps are required for successful implementation of the process. Firstly, developers, managers and IT policy makers must identify which requirements for security are based on data processed by the application. As a second step, the organisation must assess the current state of security in the software development process. This allows to identify gaps and create a proper roadmap for advancing in security maturity level.

There are several publicly available and documented Secure SDLC processes that can be used to ensure security of developed software:

- **OWASP Secure Software Development Lifecycle Project**
- **NIST System Development Life Cycle** (SP 800-64, Revision 2)
- **Microsoft Secure Development Lifecycle 5.2** (and SDL for Agile)
- **BSIMM** (Building Security In Maturity Mode)

MICROSOFT SECURE DEVELOPMENT LIFECYCLE

Microsoft SDL is a secure development process that was started by Microsoft's Trustworthy Computing team, as a response to Bill Gates' memo sent to all Microsoft employees in 2002. It called for the following fundamental security features to be present in computer systems:

AVAILABILITY

Our products should always be available when our customers need them. System outages should become a thing of the past because of a software architecture that supports redundancy and automatic recovery. Self-management should allow for service resumption without user intervention in almost every case.

SECURITY

The data our software and services store on behalf of our customers should be protected from harm and used or modified only in appropriate ways. Security models should be easy for developers to understand and build into their applications

PRIVACY

Users should be in control of how their data is used. Policies for information use should be clear to the user. Users should be in control of when and if they receive information to make best use of their time. It should be easy for users to specify appropriate use of their information including controlling the use of email they send.'

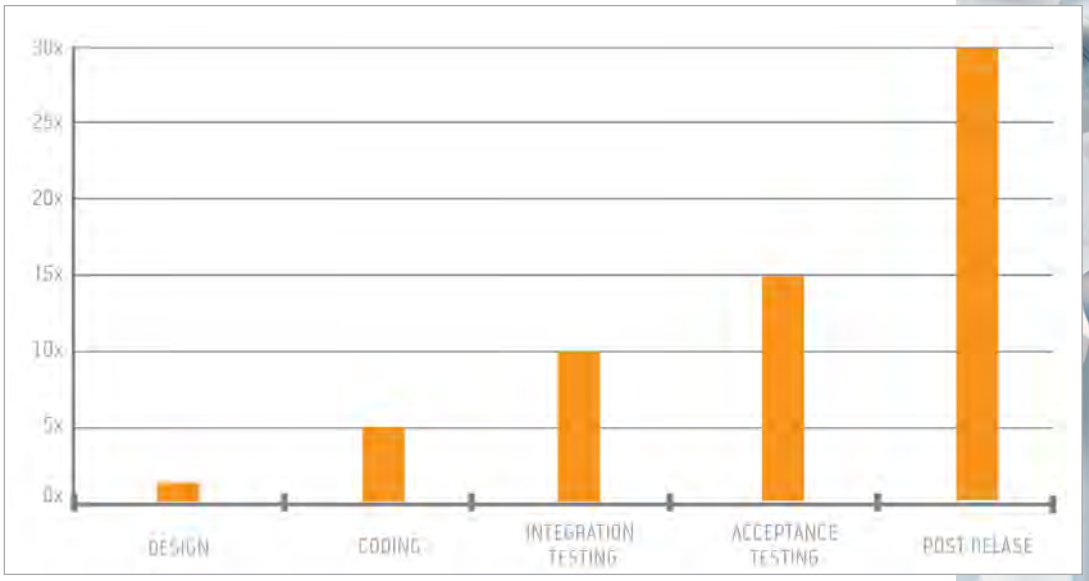
Since 2004, the process has been mandatory for all Microsoft products and it is being constantly improved, accepted and adapted industrywide.

The Microsoft SDL process is well-documented, with step-by-step instructions to be used by a development team to increase the level of security. A set of training presentations, tools and guides are provided.

	Microsoft proposes the following practices to be added to a development lifecycle process
 TRAINING	▪ Core Security Training
 REQUIREMENTS	▪ Establish Security Requirements ▪ Create Quality Gates/Bug Bars ▪ Perform Security and Privacy Assessments
 DESIGN	▪ Establish Security Requirements ▪ Perform Attack Surface Analysis/Reduction ▪ Use Threat Modeling
 IMPLEMENTATION	▪ Use Approved Tools ▪ Deprecate Unsafe Functions ▪ Perform Static Analysis
 VERIFICATION	▪ Perform Dynamic Analysis ▪ Perform Fuzz Testing ▪ Conduct Attack Surface Review
 RELEASE	▪ Create an Incident Response Plan ▪ Conduct Final Security Review ▪ Certify Release and Archive
 RESPONSE	▪ Execute Incident Response Plan

BETTER ROI

The National Institute of Standards and Technology (NIST) claims that code fixes done during the design and implementation phase can be 30 times less expensive than the ones performed after the release. This way security development lifecycle can help to reduce the total cost of development. The amount of time spent on post-development bug remediation, incident response and customer service also decreases.



Relative cost of security fixes, based on time of detection (NIST)

SANS Institute states that including security aspects in the requirements for software and cloud services not only efficiently reduces attack surfaces, but has also been proven to reduce time to market for secure business services.

SDL FOR AGILE

SDL for Agile is a way to integrate the security related activities into the Agile development process, which is based on fast and frequent delivery. This approach requires ensuring that security practices do not interfere with delivery cycle and, at the same time, makes sure that no crucial practices are missing. The practices are divided in three categories:

EVERY-SPRINT PRACTICES

Essential security practices that should be performed in every release. Some examples include:

- Threat modeling
- Running static code analysis tools
- Ensuring that defensive coding techniques are used, e.g.:
  - Ensuring all database access is performed through parameterised queries to stored procedures
  - Mitigating against cross-site scripting (XSS)
  - Using safe redirect
  - Using secure cookie over HTTPS

BUCKET PRACTICES

Important security practices that must be completed on a regular basis but can be spread across multiple sprints during the project lifetime. Examples include:

- Conducting a security review
- Creation of privacy-related documents
- Fuzz testing
- Attack surface analysis
- Data flow and input validation testing

ONE-TIME PRACTICES

Foundational security practices must be established once at the start of every new Agile project. This includes:

- Creation of a baseline threat model
- Selection of correct software tools and frameworks (XML parsers, compiler versions, libraries)
- Identifying team's privacy and security experts

To ensure that tasks are part of team effort, technical user stories are added for SDL requirements and included in project backlog. This allows them to be estimated, planned and delivered as part of the iterative delivery cycle.

DEVSECOPS – BUZZWORD OR NECESSITY?

DevSecOps is a concept which connects traditional DevOps and Security engineers. It accepts the fact that the responsibility for secure development lays not only on a limited number of security engineers, but on the whole development teams. It puts the focus on introducing automation of security activities.

This includes incorporating the following into Continuous Integration and integrated development environments (IDEs):

- static code analysis
  (which should include security-related rules)
- dynamic testing (using scanners)
- automation of security requirement checks
- planning and implementing security features
  (Security as a Code)
- ensuring secure deployments
  (Infrastructure as a Code)

Another important part of SecDevOps responsibilities is introducing best practices and developing security culture.

> *This approach ensures that a security engineer's time is spent in the most effective manner, without wasting it on repetitive tasks. The use of tools allows building a more secure application in an already pressured short iteration cycle of Agile development.*

SECDEVOPS

is a list of rules defined to make your security effort work for your project, not against it:

- *Leaning in* over *Always Saying 'No'*
- *Data & Security Science* over *Fear, Uncertainty and Doubt*
- *Open Contribution & Collaboration* over Security-Only Requirements
- *Consumable Security Services with APIs* over *Mandated Security Controls & Paperwork*
- *Business Driven Security Scores* over *Rubber Stamp Security*
- *Red & Blue Team Exploit Testing* over *Relying on Scans & Theoretical Vulnerabilities*
- *24x7 Proactive Security Monitoring* over *Reacting after being Informed of an Incident*
- *Shared Threat Intelligence* over *Keeping Info to Ourselves*
- *Compliance Operations* over *Clipboards & Checklists*

MEASURING AND ENFORCING SECURITY REQUIREMENTS IS PART OF THE PROCESS

Capturing security requirements for your project before actual development has been commenced provides many advantages, e.g. :

- Gives better understanding of application security risks and possible remediation to management, development team and Business Analysts
- Makes you compliant – when done right, security requirements are based on privacy analysis and compliance requirements that apply to data stored and processed by the system
- Makes security work anticipated, planned and cost-bound – adding security non-fictional requirements as Acceptance Criteria for User Stories makes them easy to use during estimation, development, and ensures easy mapping between requirements and controls applied during development process
- Makes them testable – once you define a requirement, it can be included in the test activities
- Improves transparency of contract with the supplier – when using a third party for software development, agreeing on a detailed list of requirements will ensure a common view on what a secure application is

OWASP Application Security Verification Standard is a well-established framework that can be used to build a set of actionable and measurable security requirements. ASVS 3.0.1 requirements cover the following features of a system:

- V1.Architecture, design and threat modeling
- V2. Authentication
- V3. Session management
- V4. Access control
- V5. Malicious input handling
- V7. Cryptography at rest
- V8. Error handling and logging
- V9. Data protection
- V10. Communications
- V11. HTTP security configuration
- V13. Malicious controls
- V15. Business logic
- V16. File and resources
- V17. Mobile
- V18. Web services (NEW for 3.0)
- V19. Configuration (NEW for 3.0)

REQUIREMENTS ARE GROUPED INTO THREE SECURITY VERIFICATION LEVELS:

- **ASVS Level 1** contains a basic list that can be applied to all applications,
- **ASVS Level 2** should be used for applications containing assets which require protection,
- **ASVS Level 3** is for the most critical systems – medical equipment, applications containing valuable intellectual property or processing a vast amount of financial transactions.

	EXAMPLE OF ASVS REQUIREMENTS FOR APPLICATION CONFIGURATION	LEVEL 1	LEVEL 2	LEVEL 3
V19. Configuration	All components should be up to date with proper security configuration(s) and version(s). This should include removal of unneeded configurations and folders such as sample applications, platform documentation, and default or example users.	✓	✓	✓
V19. Configuration	Communications between components, such as between the application server and the database server, should be encrypted, particularly when the components are in different containers or on different systems.		✓	✓
V19. Configuration	Communications between components, such as between the application server and the database server should be authenticated using an account with the least necessary privileges.		✓	✓
V19. Configuration	Verify application deployments are adequately sandboxed, containerised or isolated to delay and deter attackers from attacking other applications.		✓	✓
V19. Configuration	Verify that the application build and deployment processes are performed in a secure fashion.		✓	✓
V19. Configuration	Verify that authorised administrators have the capability to verify the integrity of all security-relevant configurations to ensure that they have not been tampered with.			✓
V19. Configuration	Verify that all application components are signed.			✓
V19. Configuration	Verify that third party components come from trusted repositories.			✓
V19. Configuration	Ensure that build processes for system level languages have all security flags enabled, such as ASLR, DEP, and security checks.			✓

SECURITY AND COMPLIANCE SHOULD BE ALIGNED

Systems used to process sensitive data (medical data, credit cards, Personally Identifiable Information) need to comply with complex regulations and laws related to system security. Standards such as General Data Protection Regulation (GDPR), Payment Card Industry Data Security Standard (PCI DSS), Health Insurance Portability and Accountability Act (HIPAA) or ISO/IEC 27001 Information Security not only require you to protect the data, but also introduce financial fines for the lack of proper security controls.

Standards recommend to regularly test applications and infrastructure for vulnerabilities. This can be done by penetration testing and vulnerability scanning. Penetration testing, when performed by a skilled and experienced individual, tests real-world risks to the business and verifies the actual state of system security. It will uncover problems that would not be found by automated tools.

PCI DSS

requires to fulfil the following goals through penetration testing: 'determine whether and how a malicious user can gain unauthorised access to assets that affect the fundamental security of the system, files, logs and/or cardholder data. Confirm that the applicable controls, such as scope, vulnerability management, methodology, and segmentation, required in PCI DSS are in place.'

ISO 27001

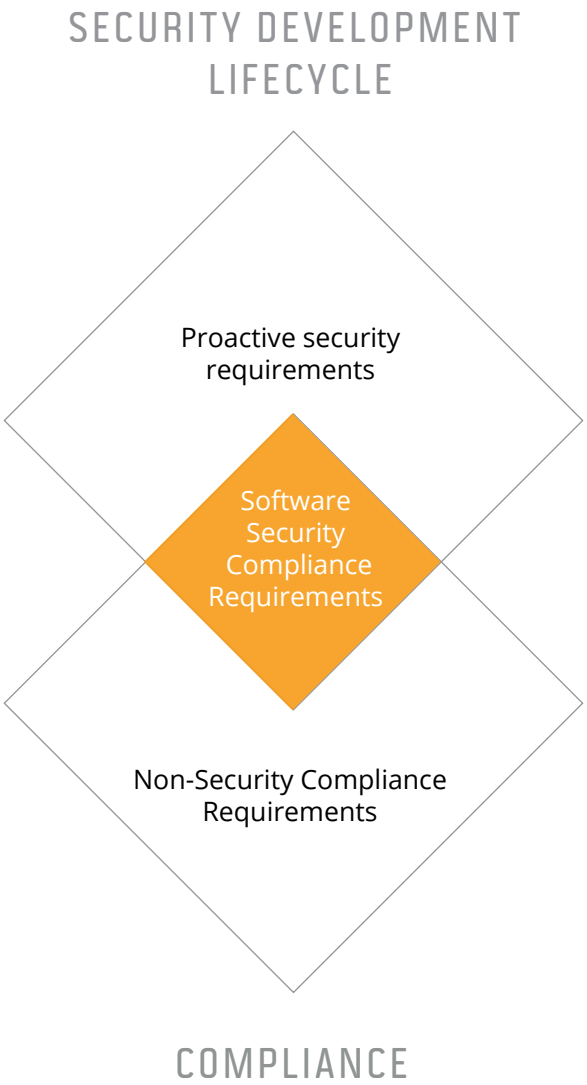
security control objective A12.6 (Technical Vulnerability Management) states that 'information about technical vulnerabilities of information systems being used shall be obtained in a timely fashion, the organisation's exposure to such vulnerabilities evaluated and appropriate measures taken to address the associated risk.'

GDPR

requires to ensure that security measures of systems used to store and process Personal Data (any information about an identified or identifiable individual) are regularly tested.

NIST GUIDE FOR IMPLEMENTING THE HIPAA

recommends to conduct penetration testing (where trusted insiders attempt to compromise system security for the sole purpose of testing the effectiveness of security controls), if reasonable and appropriate'.



ABOUT FUTURE PROCESSING

We are an experienced IT services company specialising in solving business problems of industry leaders worldwide through the use of the latest technologies. Established in 2000 in Poland, our team has grown from two people into over 900 IT specialists.

> *We pride ourselves in the high level of technical expertise and an individualised approach to each project which result in the highest quality of our solutions. Because our goal is maximising the return on investment for our clients, we always advise on the best, and not necessarily the easiest route to take during a project.*

Perhaps most importantly, we provide a range of managed security services that help our clients stay safe an increasingly complex digital world.

FUTURE PROCESSING WILL HELP YOUR COMPANY BUILD SECURE SOFTWARE.

We provide a range of managed security services that can help you feel safe. Contact us to find out more about:

- **Web Application Security Assessment**

  We will examine your websites, web applications and web servers to find security weaknesses and vulnerabilities that could give hackers an opportunity to damage or steal data processed in your system.

- **Penetration Testing**

  Penetration testing, also known as pen testing, or pen-test, is a security analysis of a software system performed by skilled security professionals simulating the actions of a hacker.

- **Secure Development Lifecycle (SDL) Governance**

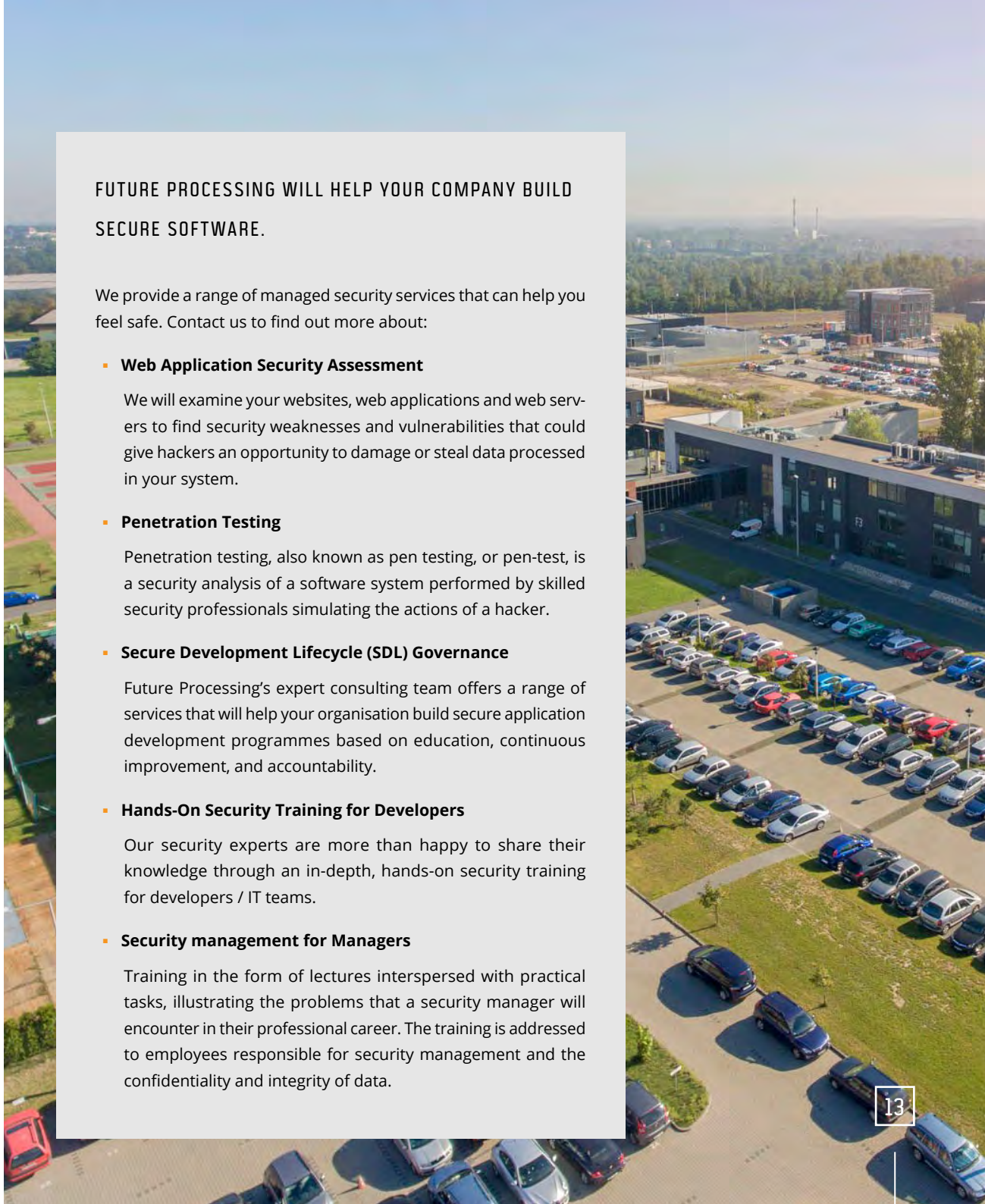
  Future Processing's expert consulting team offers a range of services that will help your organisation build secure application development programmes based on education, continuous improvement, and accountability.

- **Hands-On Security Training for Developers**

  Our security experts are more than happy to share their knowledge through an in-depth, hands-on security training for developers / IT teams.

- **Security management for Managers**

  Training in the form of lectures interspersed with practical tasks, illustrating the problems that a security manager will encounter in their professional career. The training is addressed to employees responsible for security management and the confidentiality and integrity of data.



WE HOPE THIS DOCUMENT HAS HELPED YOU UNDERSTAND THE IMPORTANCE OF SECURITY IN SOFTWARE DEVELOPMENT

If you need more advice, contact us and see how we can help.

CONTACT US:

Future Processing

Future Processing
ul. Bojkowska 37A
44-100 Gliwice
POLAND

+48 32 461 23 00
sales@future-processing.com
www.future-processing.com