APPROACH

Annual Pentest Report

2023

April 2023

01
Executive Summary

Welcome to this new edition of our annual penetration testing statistic report.

2022 has been quite busy for our ethical hacking team. The number of demands for penetration tests more than doubled compared to 2021. This is a clear sign that enterprises are becoming more aware of the necessity to regularly assess their security posture with in-depth security analysis from professional and independent ethical hackers.

In correlation, we also observe that the number of vulnerabilities discovered remains proportional to the number of projects.

In the first two editions, we focused first on Web and Mobile, and then on APIs due to the emergence of API usage in the digital world and the necessity to implement API security in the design of digital solutions.

In this year's report, we have decided to focus on the importance of including hardly detectable vulnerabilities referred to as 'Business Logic Flaws' in security analysis. Unlike technical vulnerabilities, these flaws target the logical part of the application, making them unique to each application and challenging to automate.

As you will see, detecting these vulnerabilities requires specific skills and their exploitation can cause significant damage.

This report is based on a comprehensive analysis of all application pentests performed by our ethical hacking team in Belgium in 2022 and provides valuable insights into the key trends.

What is the value of this report?

There are several values of an annual penetration testing statistics report, including:

- Providing a snapshot of the current state of application security: this report can help organisations understand the latest trends and vulnerabilities.
- Identifying areas of weakness: By highlighting the most common vulnerabilities found during penetration testing, we can help organisations identify areas of weakness in their security posture and take steps to address them.
- Enhancing cyber security awareness: This report can be used to raise awareness about the importance of cyber security and the risks associated with cyber-attacks.
- Justifying security investments: Our pentest report can help organisations justify investments in cyber security by demonstrating the potential impact of cyber-attacks and the effectiveness of security measures.

Who is this report addressed to?

Our pentest statistics report is a valuable resource for anyone interested in understanding the current state of cyber security and the latest trends and vulnerabilities.

This report could be of interest to a wide range of profiles, including:

CISOs and other security executives

Stay up to date with vulnerabilities and threat trends and adapt their roadmap

Security Operations Centre (SOC) teams

Stay up to date with threat and vulnerability specificities to implement response strategies

IT and security teams

Know about latest threats and vulnerabilities to implement appropriate counter measures

Compliance and regulatory professionals

Raise awareness on compliance and data privacy risks

Developers and product managers

Better integrate cyber security in the development lifecycle

Business leaders and stakeholders

Quantify business risks and the steps being taken to mitigate them

02
Report Methodology

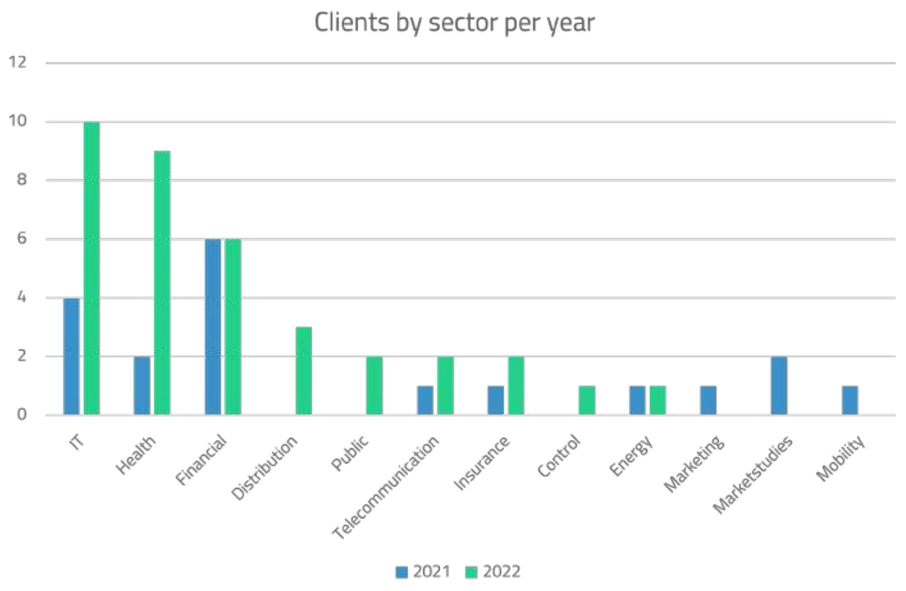
No changes have been made on our method compared to previous editions. The more significant changes come from the fact that more data has been processed thanks to the volume of projects and vulnerabilities identified in the past year.

Sampling

Our analysis is based on a sample of 60 projects in 2022, on 38 different clients amongst 9 industries in Europe.

All projects selected as part of the sampling adopted a "grey box" approach for the execution of the pentest.

We have observed a rise in the demand for pentests from the healthcare industry, primarily because of the immense value of the data they process and store.



Risk Levels and Scoring

As a reminder, here is the risk categorisation we use when defining a vulnerability's risk level:

- Critical: open issue leading to a direct threat
- High: open issue highly exploitable
- Medium: open issue with significant risk
- Low: open issue with minimal risk
- Negligible: open issue with zero risk

Note that this scoring has been defined in such way that it also considers the context of the vulnerability.

Data Confidentiality

We apply strict controls in order to ensure the confidentiality of our customers' data before, during and after the project.

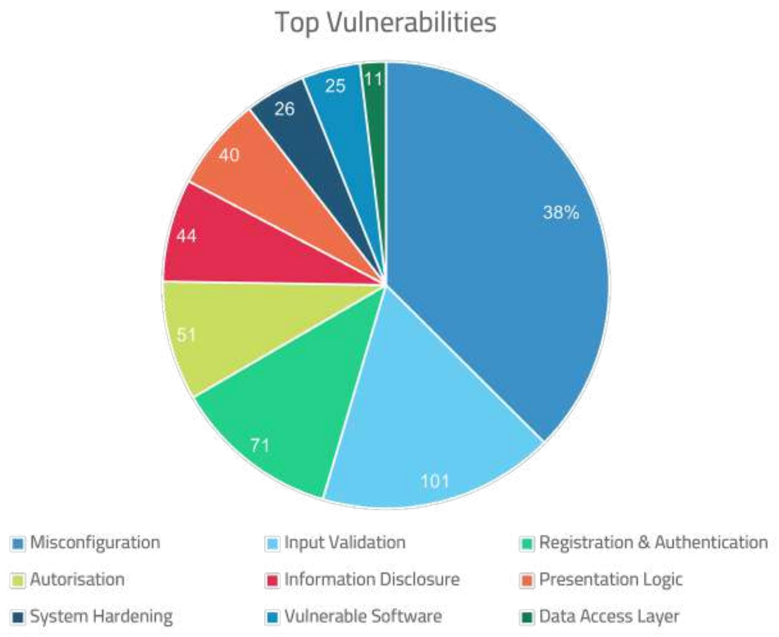
Reports and other customer information are securely stored in our systems according to our retention policies and contractual obligations, in compliance with our customers' CVDP (Coordinated Vulnerability Disclosure Policies).

The sample has been selected in such a way that it is impossible to identify any of our customers.

03
Findings

In this section, we analyse all of the pentests performed by our ethical hacking team in 2022 to identify the most common vulnerabilities and we compare the results to previous years.

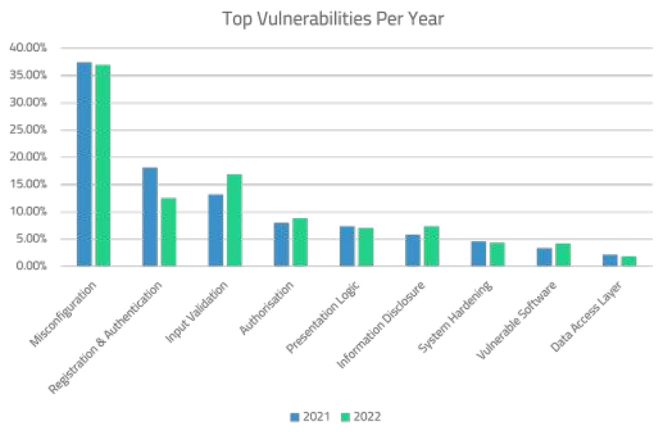
Common Top Vulnerabilities



With similar results in 2020 and 2021, misconfigurations remains the #1 vulnerability amongst all our projects.

As a reminder, OWASP security misconfiguration refers to insecure security controls resulting from misconfigurations, such as improper security hardening, incorrectly configured cloud service permissions, unnecessary features like default accounts, or enabled but never changed default passwords. It seems like, despite the rise of cyber-attacks, compliance requirements and the concern of boards, basic cyber hygiene is still not applied.

Input validation moved from third to second position, followed by registration and authentication issues but overall, the graph remains relatively stable compared to the previous year.

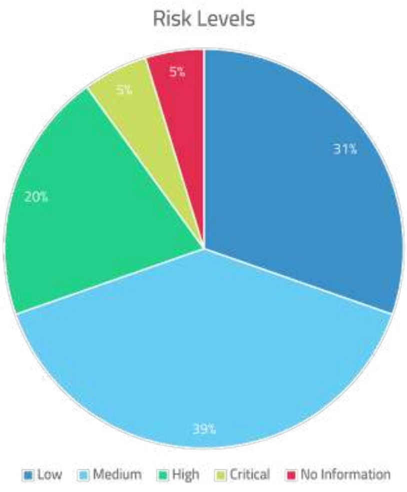


Vulnerabilities Per Risk Level

25% of all vulnerabilities have a risk level of either critical or high. This is a significant issue and leaves a considerable attack surface for adversaries to exploit. It's akin to leaving 25% of your doors and windows open when you leave your home.

Compared to last year, we note a clear focus on high and critical vulnerabilities which are decreasing respectively by 4% (from 24% to 20%) and 5% (from 10% to 5%).

This is an encouraging sign, showing that efforts are being made to fix these issues.



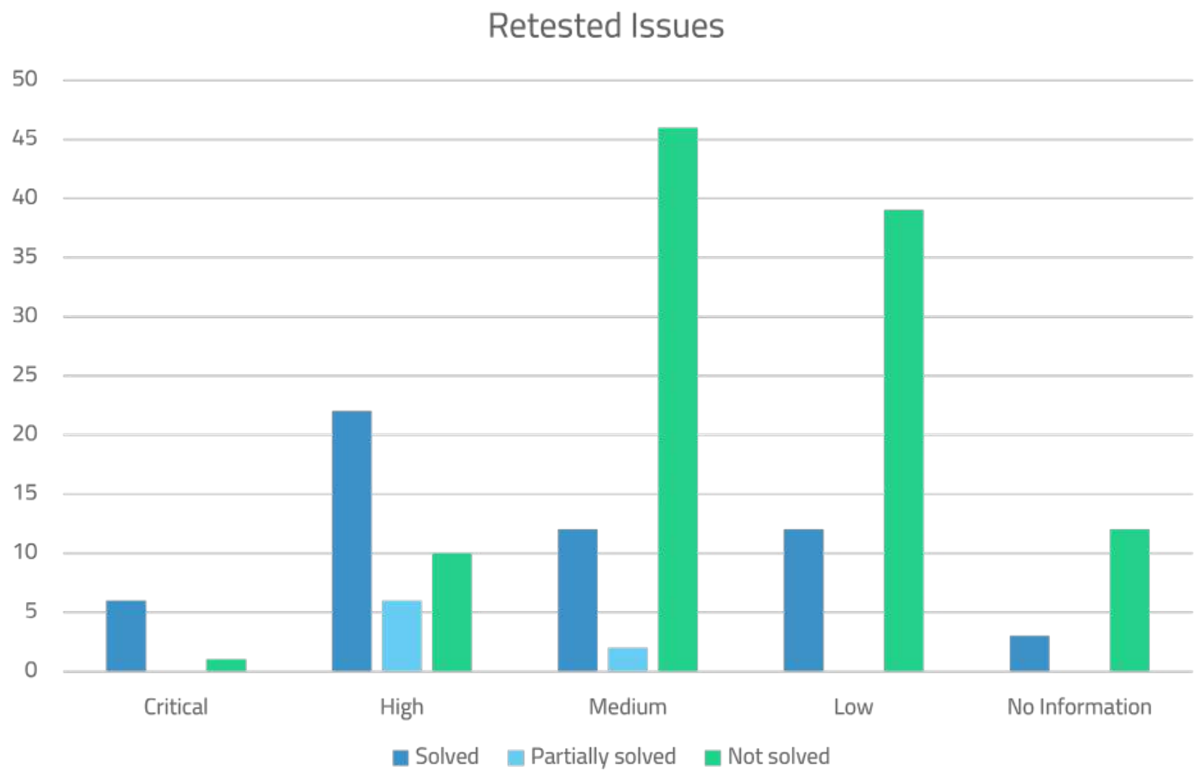
After retesting

Retests are crucial because they provide a way to verify whether the identified vulnerabilities have been adequately addressed and fixed.

Results show a significant decrease of issues solved after re-testing, with the number of unsolved issues rising from 44% to 63%.

In our conversations with customers, we have noticed that the primary reason for this is the challenges they face in addressing the issue. These challenges can be attributed to three main factors:

1. They don't have the skills or competence and don't know how to get this solved.
2. The issue is systemic in the application and would require a significant amount of effort to remediate.
3. Developers focus their energy on solving critical and high vulnerabilities. There is a tendency to accept the risk of not solving medium and low ones.

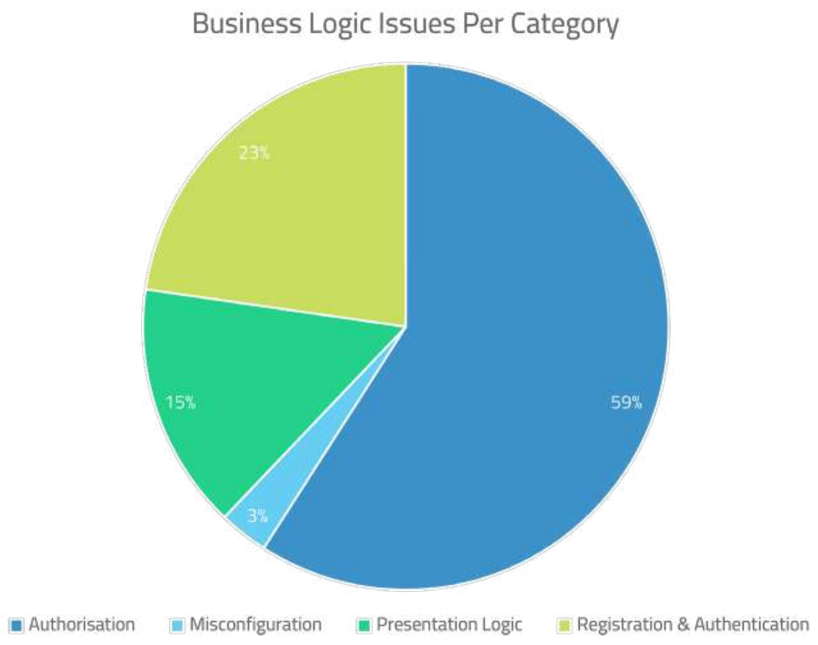


04 Testing the Business Logic of Your App

In this section of the report, we focus on Business Logic Flaws, a type of vulnerability that is critical but difficult to detect.

Business logic (in web and mobile applications) is the workflow that defines how the application should operate. Business logic flaws occur when attackers exploit vulnerabilities in the intended business logic flow to achieve malicious objectives.

Business logic flaws can fall into different categories depending on the part of the application they affect. As we suspect, the registration process, authentication and authorisation represent 82% of the total amount of vulnerabilities identified.

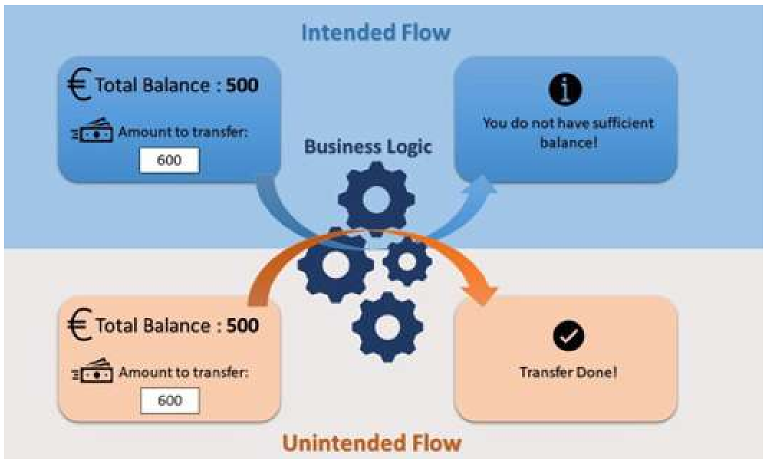


Taking a banking application example, its business logic allows a user to transfer an amount of money that is less or equal to the total amount the user has.

An attacker may discover a flaw that enables them to transfer an amount of money exceeding their total balance.

Each application has its own business logic that depends on the domain of the business, the number and complexity of operations and most especially on the way the developers of the application decide to implement this business logic.

If not thought out thoroughly, the implementation of the business logic can be abused by attackers which may compromise the purpose of the whole application.



Business Logic Example: Intended vs Unintended Flows

So why is it important to talk about business logic flaws?

- Business logic vulnerabilities are pervasive and vary in terms of risk level. They are a common occurrence in web and mobile applications.
- These vulnerabilities are mostly left untested as normal application testing and code reviews are not sufficient to find them.
- Most importantly, they are very dangerous, as many of them target the core of the business and undermine the intended purpose of the application. Failing to protect against such vulnerabilities not only leads to financial losses and data breaches but can also severely damage the business's reputation.

How are they different from traditional/technical vulnerabilities?

Hard to detect

Unlike many technical vulnerabilities, business logic vulnerabilities are challenging to detect because they typically follow the expected behaviour of the application, albeit with subtle deviations. For the most part, technical vulnerabilities have a specific pattern or use payloads that make them detectable by automated scanners, Web Application Firewalls (WAF), Intrusion Detection Systems (IDS) and other automated solutions.

Most business logic vulnerabilities on the other hand, do not have a specific signature to be detected. Since each application workflow may differ from others, the vulnerability will also be different making automating the detection a challenging task.

Requires manual tests with knowledge of the domain

As previously stated, automated tools are mostly ineffective in detecting these vulnerabilities, meaning that manual testing is necessary. Additionally, these vulnerabilities can be highly specific to the business's domain. Therefore, the penetration tester who will perform such tests, should have good knowledge of the application business logic in order to understand potential ways to break it.

This knowledge is derived from both extensive testing experience across multiple applications that may share similar logic, as well as navigating the tested application to understand its underlying logic. In some cases, it may also involve reviewing any publicly available documentation related to the application and its components.

Can affect many aspects of the application

As will be explained in the attack vector examples, business logic flaws can target many aspects of the application. This includes the authentication process, users' authorisation, or the business process itself... Unlike technical vulnerabilities which are mostly known to target a specific aspect of the application.

Needs unconventional tests

Typically, developers and testers focus on ensuring that the application functions as intended, overlooking potential manipulations and forced actions that may lead to business logic flaws.

To effectively test for these vulnerabilities, it's essential to consider not only what the developer intended to achieve but also what an attacker might attempt to do.

Examples of Business Logic Attack Vectors

As explained earlier, business logic flaws can be very specific to the business domain and can be different from one application to another. However, they sometimes share themes allowing them to be classified into categories. Hereunder, we give some examples of common attack vectors that target business logic.

Abusing authentication logic

Authentication can be a dangerous target for business logic flaws. Developers tend to assume that the user does not have much control on the operation flow, and thus the intended flow will always be followed. Password reset and forgotten password functionalities are common places where these flaws can appear. Developers assume that users only have control over the initial reset request, while an attacker attempts to manipulate any subsequent actions in the flow.

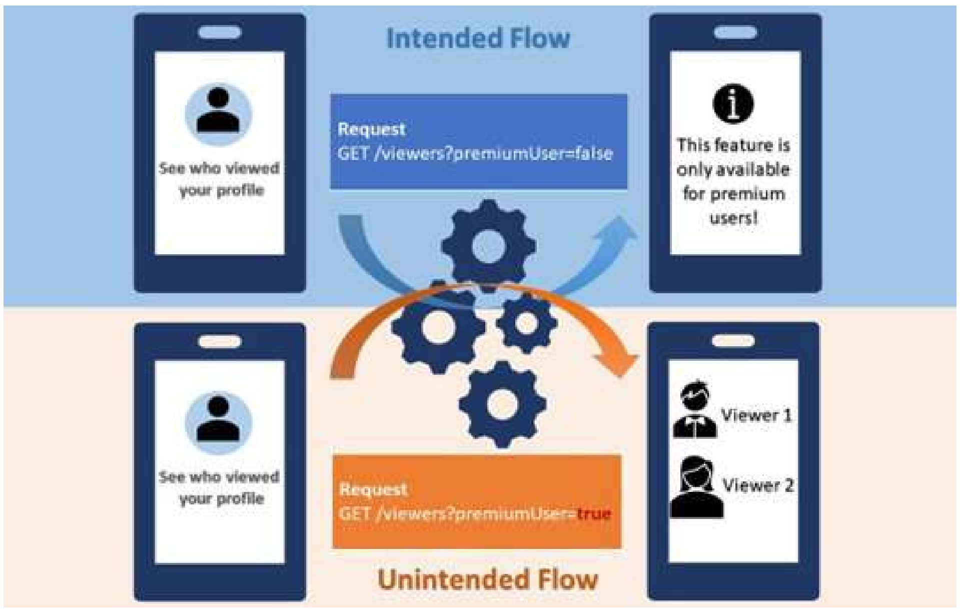
As an example, the forgotten password feature often relies on a unique token generated with the password reset request to guarantee that only the authorized user with the correct token can reset the password. However, attackers may attempt to exploit this process by deviating from the intended flow and acquiring the token in an unintended manner, such as exploiting a weak token generation mechanism, intercepting the request with the valid token, or tampering with the token parameter to bypass authentication checks.

Abusing authorisation logic

Developers may think that implementing strict security measures in the early stages of the application flow, like ensuring only authenticated users can access the application, will grant secure handling to any subsequence flow. Incorrectly assuming this can result in access control problems that disrupt the intended logical flow of determining who is authorized to access certain data or sections of the application.

One example of these flaws is relying on a user ID to access specific resources. Developers may assume that users have no control over this parameter, but attackers can manipulate the user ID to access resources and information belonging to other users. They could also use an administrator ID to access restricted functionalities and escalate their privileges.

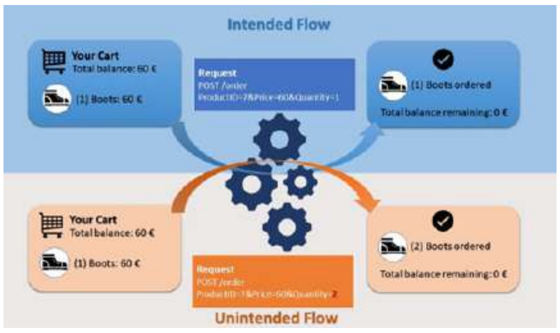
Another example, from a mobile application perspective, is where access to some features or functionalities is restricted to premium users. Developers often overlook the fact that attackers can intercept traffic between a mobile application and the backend. This is especially true for mobile applications, where developers assume that users can only interact with the interfaces provided on the mobile application. For instance, if a parameter is used to determine whether this is a normal or premium user (e.g., premium=true), an attacker can manipulate this parameter to gain premium access.


Intended Flow
See who viewed your profile
Request
GET /viewers?premiumUser=false
This feature is only available for premium users!
See who viewed your profile
Viewer 1
Viewer 2
Request
GET /viewers?premiumUser=true
Unintended Flow

Example of abusing the authorization logic (accessing premium reserved functionalities)

Manipulating the legitimate input

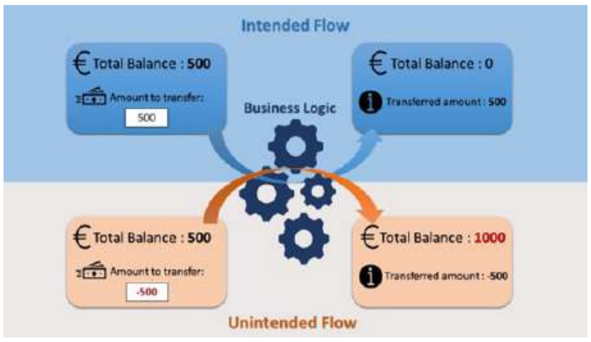
Applications rely on user input, and one of the essential development security rules is to never trust it. While technical vulnerabilities arise from inadequate input validation, which allows attackers to inject malicious payloads and exploit vulnerabilities, business logic vulnerabilities stem from attackers manipulating certain parameters to perform actions that violate the intended flow of the application. This type of vulnerability can cause significant damage in domains such as online banking and shopping.



Example of manipulating the input (modifying the quantity of the item in the final submission step)

For example, in an online banking application, an attacker chooses to transfer an amount of money, the application checks whether the attacker has sufficient money to do the transfer, which is valid in this step. if a parameter is used to transfer the amount of money, an attacker can manipulate it to transfer more than they actually have. Similarly, attackers may try to modify the price or quantity of a product when submitting an order if the developers used parameters to define these values.

Using unexpected input



Example of using unexpected input (negative number for amount of transferred money)

Another way to abuse the business logic starting from user's input, is by providing unexpected input.

For example, if a banking application is expecting a number to define the amount of money to be transferred, an attacker can attempt to provide a negative number to test how the logic of the application would deal with negative numbers, and whether the attacker can abuse that logic to add money to its account (total – (- transferred) = total + transferred).

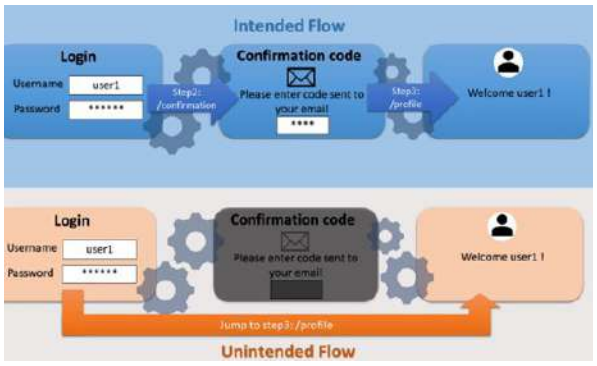
Another way in which input can be abused in this context, is when providing input that is out of the range defined by the developers. For example, providing out of range numbers or a very long string which may result in further vulnerabilities depending on various factors.

Wandering in the workflow

Developers can make a dangerous assumption that all users will follow the defined steps of the application logic in the correct order, submit each step and click "next" to proceed to the next step. However, attackers can attempt to jump ahead in the flow to bypass certain checks and abuse the application.

One example of such an attack vector is when an application uses a weak two-factor authentication where an attacker attempts to jump over the second factor check. If the attacker knows the authentication call used after providing the two-factors, and the developers did not implement any checks to ensure that the user has completed the entire authentication process.

In that case, the attacker can use the credentials, and when prompted to enter the second factor, manually send the authentication call to bypass the two-factor authentication logic. This method can also be used to skip certain steps in the flow and bypass checks enforced in those steps.



Wandering in the workflow example (bypassing 2FA logical flow)

Breaking the intended logic

This category is broad and can be very specific depending on the domain of the business implementing the application. Online shops and banking application are once again a popular attack surface for such flaws.

For example, an attacker prepares a valid transfer request, but repeatedly sending the same request at the same time, resulting in the attacker being able to transfer more money/credits than they have.

How to protect against such flaws

Test the application for flaws

To detect and prevent such flaws, applications need to undergo testing that includes manual penetration tests conducted by experienced testers who possess appropriate skills such as knowledge and experience in exploiting these vulnerabilities, as well as strong analytical skills to comprehend the application, its objectives, and the various features to identify potential flaws.

Skilled testers will be able to identify potential areas in the application that should be tested against business logic flaws in the scoping phase prior to the test, which helps both the tester and the business to plan the required tests.

Testing should be performed repeatedly, especially when new features or functionalities are added to the application.

Consider business logic flaws while developing

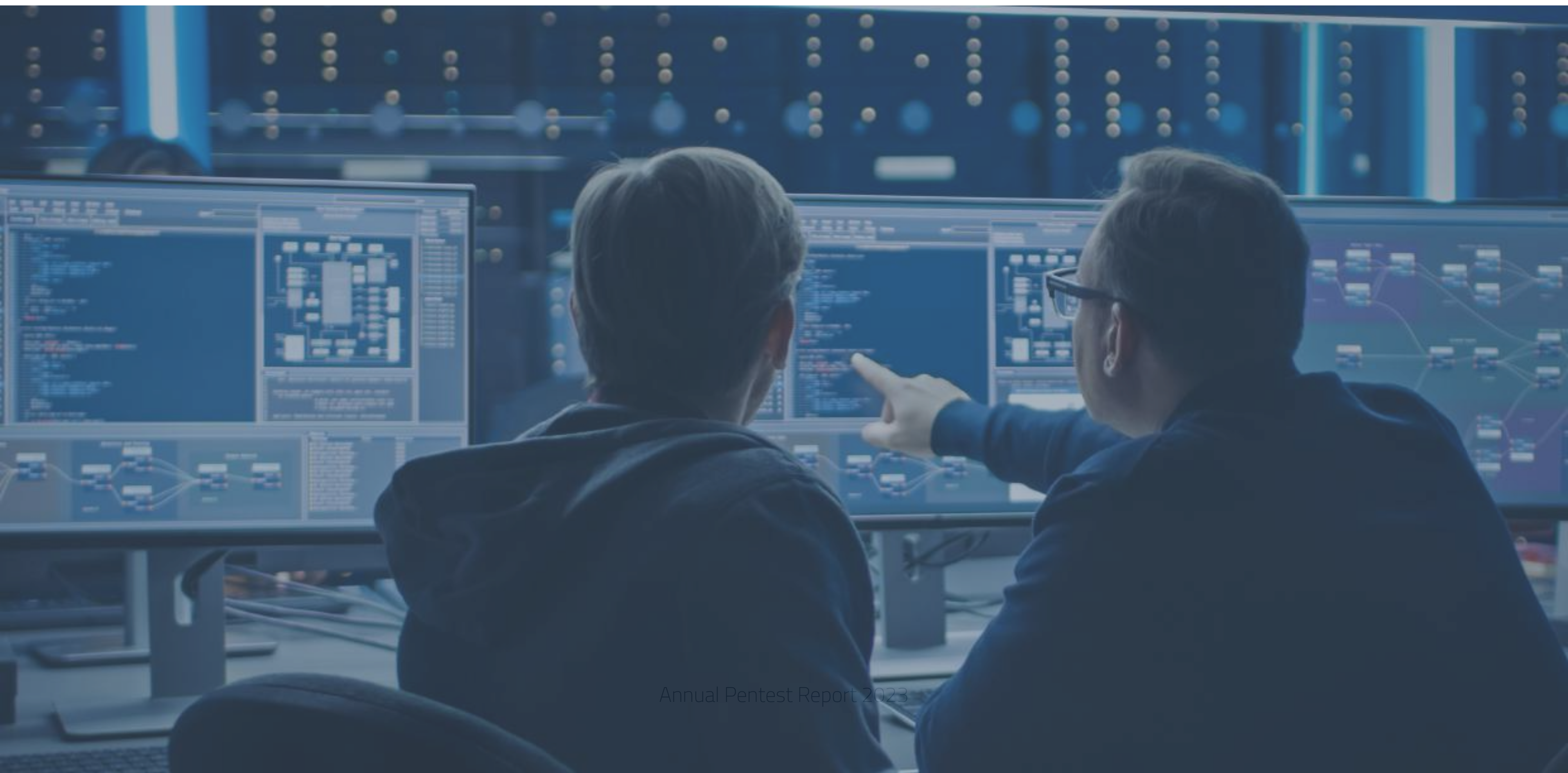
Developers need to be mindful and informed that the application's behaviour will not always align with the intended flow they have designed.

Attackers will attempt various methods to deviate from the expected flow and exploit the application's purpose.

Use application threat modelling

Threat modelling is a structured approach to identify potential threats to a system and determine the value of potential mitigations in reducing or removing those threats.

Implementing a threat model helps in identifying design flaws in business logic by defining use and misuse cases to analyse any potential abuse of business logic.



05 Takeaways & Recommendations

Key takeaways

With this analysis, the following takeaways are important to consider when you are developing and maintaining applications:

- Misconfiguration remains the primary type of vulnerability identified (more than 1/3).
- Solving or remediating identified vulnerabilities remains a big challenge for developers and IT.
- Organisations seems to adopt a risk-based approach by focusing on solving high and critical vulnerabilities and weaknesses only, accepting generally to not solve medium and low vulnerabilities.
- Business logic flaws are common, especially in applications with complex operations and logic.
- Business logic flaws have in vast majority high or critical risk levels.

Recommendations

- Continue to raise awareness within your IT and developer teams to adopt basic cyber security hygiene.
- Enforce hardening policies on operated infrastructure and middleware.
- Test your applications on a regular basis: pentest, automated scan, static code review, etc.
- Train your development team on DevSecOps practices and tools.
- Make sure the test includes business logic flaws.
- In applications with complex business logic, implementing a threat model is recommended as it helps in identifying business logic flaws.
- To ensure the security of the application, implementing a Web Application Firewall (WAF) can mitigate many existing and potential risks.
- Promote our report in your developer community.

06

About Our Ethical Hacking Team

Our Ethical hacking team is a highly specialized service line that operates within our Security Operations Centre (SOC). Here are some key figures and details about our team:

- In 2022, we completed **over 100 ethical hacking projects.**
- We have worked with **over 60 clients around the world**, including small and medium-sized businesses, large corporations, and both public and private sector organisations.
- Through our projects, we have **identified over 600 vulnerabilities** across a wide range of domains, including web, API, mobile, network and infrastructure components, OT/IoT, Wi-Fi, and Active Directory. Of these, 120 were classified as high severity and 31 as critical. We have also found some vulnerabilities that we cannot disclose to major software vendors.
- Our team members hold **internationally recognised certifications** such as OSCP, eMAPT, eWAPT, OSWP, CRTO, eCPPT, OSCE, CRTP, and ACIP (IoT pen tester).
- We follow best practices:
- We actively **participate in bug bounty and CTF events** like HackTheBox and Cyber Security Challenge Belgium. We are also members of several offensive security communities.
- In addition to our broad expertise, we have specific red team capacity with **professional tools and skills to simulate real-world attacks** and provide actionable insights to our clients.

To learn more about our SOC service and Approach, you can scan the QR code provided.

About Approach

Approach is a pure-play cyber security and privacy firm.

For more than 20 years, we have been building trust in the cyberspace and helping our clients deal with cyber-attacks, incidents and breaches.

We offer 360-degree solutions to improve your cyber resilience: anticipate, prevent, protect, detect, respond and recover.

We provide tailored and local services matching your needs: consulting and audit services, training and awareness, security technology implementation and development services, and outsourced Managed Security Services thanks to our own Security Operations Centre (SOC).

Approach is a scaleup company with a team of a hundred people spread across several sites in Belgium and Switzerland. Our company is ISO 27001 certified and ISO27701 verified. Approach has received the label: Cybersecurity Made in Europe ™.

www.approach.be

Thank you!

Approach Louvain-la-Neuve

Axis Parc
Rue Edouard Belin 7
1435 Mont-Saint-Guibert
Belgium

+32 10 83 21 11

Approach Antwerp

Pamica Building
Rouaansekaai 1
2000 Antwerp
Belgium

+32 3 366 21 76

www.approach.be

Follow us: approach-belgium

Do you want to subscribe to our newsletter? info@approach.be

Do you need to talk about your needs and receive an offer? sales@approach.be

Interested in joining us? jobs@approach.be

To learn more about our ethical hacking solutions: head to our website

APPROACH