A1 Digital

Security-Assessment

Security test of Example - External Infrastructure

Recipient:
Example GmbH
Example Str. 12
1234 Example

Classification: Confidential
Date: 13.06.2025
Version: **1.0**

Contact at A1 Digital International GmbH & Co KG:

Alice Codex
ask.security@a1.digital
+431234567890
Department Security

Lassallestraße 9, A-1020 Wien

1 Change record

Date	Version	Description	Author
09.06.2025	0.1	Initial Creation	Alice Codex
13.06.2025	0.9	Review	Trent Trustworthy
13.06.2025	1.0	Published	Alice Codex

Table 1 - Change record

Table of Contents

2 Management Summary

The results of the security test are summarised briefly below. More detailed descriptions of the individual specific aspects with references to additional resources as well as recommended countermeasures can be found in chapter 5.

2.1 Results

On the dashboard available at `https://dashboard.example.com`, an SQL injection vulnerability was identified that allowed authorized users to read, modify or delete data in the database. Attackers could use this to access user information and passwords. Because the dashboard is connected to the database with a privileged account, the SQL injection can furthermore be leveraged to have command execution on the system.

The application has furthermore been found to have accounts with default credentials. Those allow attackers to easily compromise access controls by testing well known or easily guessable combinations. In combination with the SQL injection this becomes a critical vulnerability, as it allows any attacker to authenticate, and therefore exploit the injection even without their own account.

At the time of the assessment, the `srv02.example.com` system appeared to be using outdated software that had at least the known vulnerability for which there are already publicly available exploits. Attackers could use these exploits to access sensitive data on this system, make the system unavailable, or fully take over the system.

On the target `mail.example.com`, an SMTP server has been identified, on which it was possible to enumerate valid user accounts. Attackers could use the obtained information for follow-up attacks, such as a brute force attack against the discovered users.

In the course of the testing, it was detected that **Remote Desktop Protocol (RDP)** was accessible for the host `srv01.example.com`. RDP has repeatedly been affected by serious security vulnerabilities in the past. Due to the high security risk, RDP should therefore not be directly exposed to the internet.

Some affected systems were configured with weak **SSH settings**. Attackers with access to the network traffic could theoretically break the encryption and thus gain access to sensitive data such as usernames and passwords.

2.2 Recommended next steps

Recommendations for the next 3 months:

- Access to RDP over the internet should be blocked.
- Default accounts should be disabled, or their passwords changed.
- The SQL injection vulnerability should be resolved by using prepared statements in all queries.

Recommendations for the next 6 months:

- It should be ensured that all software in use is up-to-date.
- SMTP commands that allow user enumeration should be disabled.
- SSH and other encrypted protocols should be configured to only support secure and modern cipher, key exchange and MAC algorithms.

Recommendations for the next 12 months:

- An update process should be established for all software in use.

- The newly established security procedures should be tested for effectiveness.

2.3 Overview of weaknesses

The following table provides an overview of the identified weaknesses and an estimate by A1 Digital International GmbH & Co KG of the effort required to implement countermeasures. Figure 1 shows a schematic representation of the identified weaknesses.

Weakness	Risk (CVSS)	Countermeasures
SQL Injection leading to Command Execution	Critical (9.9)	Medium
Default Credential Usage	Critical (9.8)	Low
Outdated Software with known Vulnerabilities	High (8.4)	Medium
Open SMTP Server allows User Enumeration	Medium (5.3)	Low
Remote Desktop Protocol (RDP) publicly accessible	Medium (5.3)	Low
Weak SSH Settings	Medium (4.2)	Low

Table 2 - Overview of weaknesses

The penetration test findings indicated the detection of vulnerabilities, encompassing **2 Critical**, **1 High** and **3 Medium** severity issues:

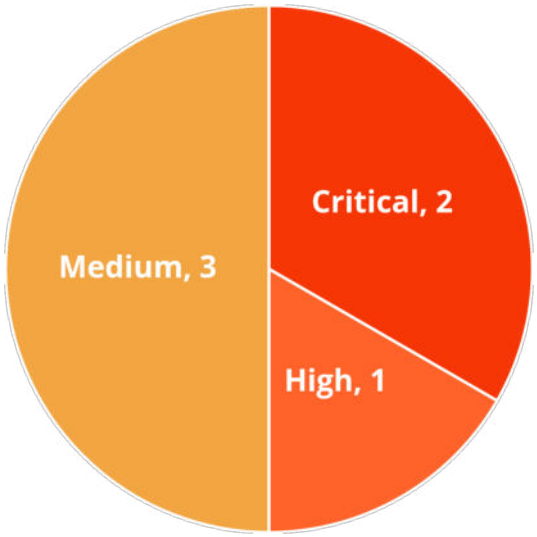


Figure 1 - Distribution of identified vulnerabilities

2.4 Weakness categorisation

A coarse categorisation of the identified weaknesses was made to get an overview of the areas in which the most security-relevant findings were identified. The categories of weaknesses are as follows:

- **Configuration Issue:** Errors in the configuration of software or hardware components.

  If repeated weaknesses have been identified within this category, training for system administrators on how to securely configure the components they support can help.

- **Outdated Software:** Outdated software components with known security-relevant problems.

  If outdated software is a frequently identified problem, it is recommended to establish a continuous update and patch management process to install security-critical updates in a timely manner.

- **Input Validation/Output Encoding:** Missing validation of user inputs or missing correct encoding of outputs of the software.

  Frequent errors in this category are likely related to a lack of secure coding training. Regular secure coding training for software developers could increase security and software quality.

- **Other:** Findings that do not fall into one of the three categories above.

The following table identifies the categorisation of weaknesses within the identified findings.

Weakness	Category
SQL Injection leading to Command Execution	Input Validation / Output Encoding
Default Credential Usage	Configuration Issue
Outdated Software with known Vulnerabilities	Outdated Software
Open SMTP Server allows User Enumeration	Configuration Issue
Remote Desktop Protocol (RDP) publicly accessible	Configuration Issue
Weak SSH Settings	Configuration Issue

Table 3 - Weakness categorisation

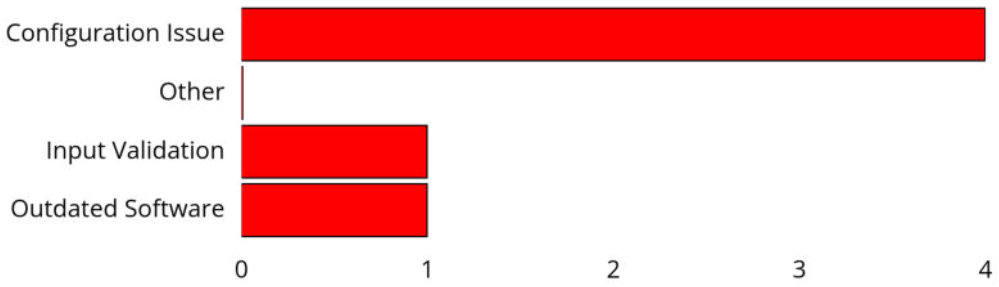


Figure 2 - Chart of weakness count per Category

2.5 Disclaimer

The effort for this test was estimated using a time box approach, i.e., only weaknesses within the agreed time window were identified. The aim was to identify and document as many security-relevant weaknesses as possible in the systems being tested. However, we do not assume any liability for completeness of the findings listed in the report.
The test provides a snapshot at the time of the security assessment, so future IT security risks cannot be derived from it.

3 Scope

Example GmbH commissioned A1 Digital International GmbH & Co KG to perform a security test of the systems listed below.

The security test took place between **09.06.2025** and **13.06.2025**. The security assessment was conducted over a period of 5 person days, a more detailed description regarding the procedure can be found in chapter 4.

3.1 Systems tested

The following systems were considered within the assessment.

IP	Hostname
203.0.133.8	nextcloud.example.com
203.0.133.2	srv01.example.com
-	example.com
203.0.133.6	mail.example.com
203.0.133.5	dashboard.example.com
203.0.133.15	relay.example.com

Table 4 - Systems tested

3.2 User accounts used

Several test accounts were created on the different exposed services with the e-mail address `pentest@example.com`. It is recommended to block/delete these accounts.

After gaining a remote command execution on the target `dashboard.example.com`, a temporary folder has been created at `/tmp/pentest`. This folder and its content should be deleted permanently.

4 Procedure

To cover the widest possible range of possible weakness categories, the test was conducted following the Open Web Application Security Project (OWASP) Testing Guide Version 4 (see chapter 6.6). The aim was to identify all security-relevant weaknesses that were present in the systems at the time of the test.

A number of criteria were defined in advance to enable classification of penetration tests that have been carried out. The following figure is based on the study "implementation concept for penetration tests" [1] from the BSI and is intended to reflect the procedure within this test.

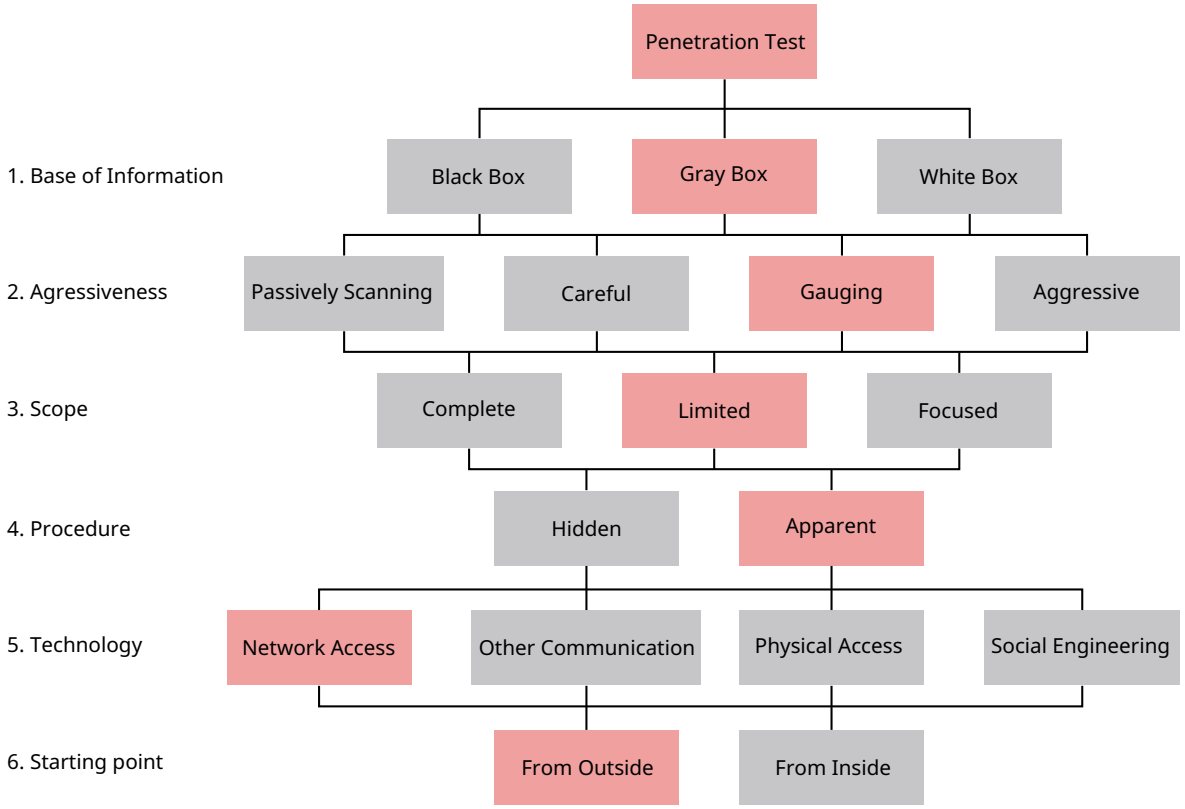


Figure 3 - Implementation concept for penetration tests [1]

4.1 Risk assessment according to CVSSv3.1

The Common Vulnerability Scoring System (CVSS) provides the ability to identify and score the underlying characteristics of a weakness. The result is a numerical value that can range between **0.0 and 10.0**, with **10.0** being the highest and thus most critical value. For a detailed description of the CVSS metrics, see chapter 6.2. To be able to express the risk in words, five different value ranges are defined, which are described in the chapter 6.3. Accordingly, a risk can be classified as **"none", "low", "medium", "high"** and **"critical"**.

1. https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Studien/Penetrationstest/penetrationstest.pdf

5 Identified weaknesses

The weaknesses identified during the test are described below and assigned a risk rating. This risk assessment is carried out according to the CVSSv3.1 standard and was performed by the assessors to the best of their knowledge and belief. The risk assessment may therefore differ from the customer's assessments, as in most cases the assessor does not have sufficient background knowledge to perform a specific business risk assessment.

Each identified weakness described includes recommended countermeasures and references to external resources for further information.

5.1 SQL Injection leading to Command Execution

CVSS Score	**9.9 (Critical)**
CVSS Vector string	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:H/A:H (show in first.org)

Affected Systems

- dashboard.example.com (203.0.113.5:443)

Description

On the dashboard available at `https://dashboard.example.com`, an SQL injection vulnerability was identified that allowed authorized users to read, modify or delete data in the database. Attackers could use this to access user information and passwords. Because the dashboard is connected to the database with a privileged account, the SQL injection can furthermore be leveraged to have command execution on the system.

Recommendations

- Most SQL injection vulnerabilities can be prevented by using **parameterized queries** (also known as **prepared statements**) instead of string concatenation within the query.
  - If the use of prepared statements is not possible in this application, ensure that all user input is properly sanitized before using it within an SQL query.
- The **principle of least privilege** should be implemented universally.
  - A user different from `postgres` should be created and dedicated to the database of the dashboard.
  - This user should only have access to data that is really necessary.
  - Privileges should also be limited to actions on the database itself and not on the whole Database Management System.

Technical Description

An **SQL injection** is a web application vulnerability that allows attackers to send queries directly to the database in order to gain unauthorized access to it. The vulnerability occurs when the user's input data is not sufficiently validated on the server side and is passed directly to the database. The following illustration shows an example of how an **SQL injection** can be exploited.

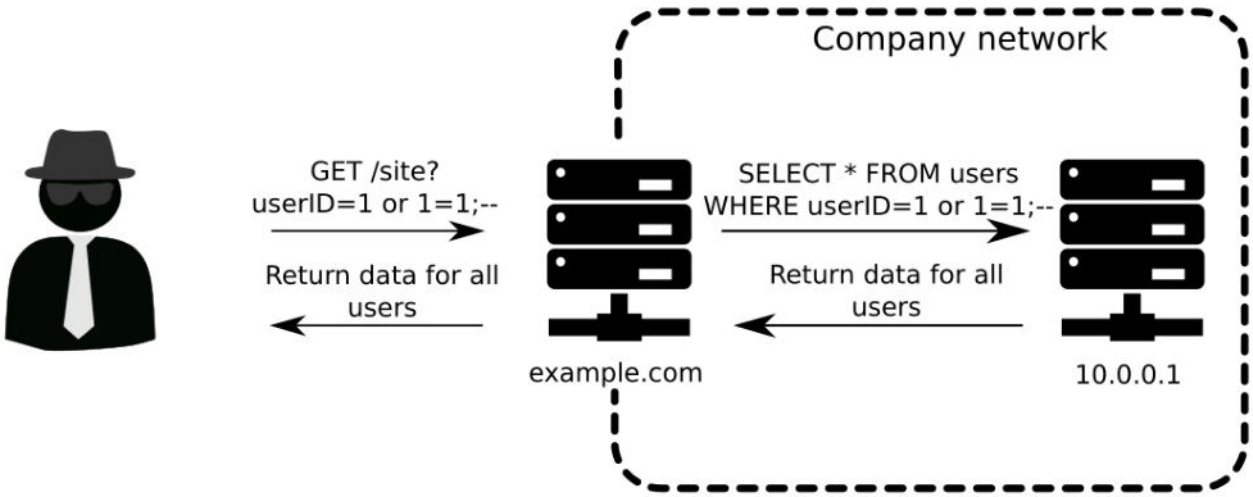


Figure 4 - All user data is queried by exploiting an SQL injection vulnerability

In the dashboard application, the URL- and data parameters are not parsed properly. Due to this, attackers can use an SQL injection to directly query the database. In this example, the type of the injection is **blind Boolean-based**. An attacker cannot have the direct output of the query. However, the parameter `success` of the response will be `true` or `false` depending on whether the result is empty or not. Therefore, an attacker can enumerate the whole database, character by character. This requires a large amount of queries, but can be automated with tools like **sqlmap**. At the end, data like the database version or the hash of the admin password can be retrieved, as shown in the screenshots below.

Figure 5 - Search for the database version

Figure 6 - Search for the admin password

Moreover, the web application is connected to the database with the user `postgres`. As shown below, this user is granted the role of `pg_execute_server_program` which is used to execute commands on the system hosting the database.

```
postgres=#  SELECT oid, rolname FROM pg_roles WHERE pg_has_role('postgres', oid, 'member');
  oid  |          rolname
-------+---------------------------
    10 | postgres
  3373 | pg_monitor
  3374 | pg_read_all_settings
  3375 | pg_read_all_stats
  3377 | pg_stat_scan_tables
  4569 | pg_read_server_files
  4570 | pg_write_server_files
  4571 | pg_execute_server_program
  4200 | pg_signal_backend
```

In the following listing, access gained through this SQL Injection can be seen, confirming the user to be `postgres` and being able to access files.

```
postgres@host whoami
postgres

postgres@host: id
uid=112(postgres) gid=118(postgres) groups=118(postgres),117(ssl-cert)

postgres@host: cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
```

As a consequence, attackers can execute arbitrary commands on the system. This can lead to a full compromise of the system, if attackers manage to escalate their privileges to the superuser (e.g. root). It can also be used to attack internal vulnerable systems that are not directly accessible from the internet. This way, attackers could steal, delete, or encrypt data, compromising confidentiality, availability, or integrity.

References

- https://owasp.org/www-community/attacks/SQL_Injection
- https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html

5.2 Default Credential Usage

CVSS Score	9.8 (Critical)
CVSS Vector string	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H (show in first.org)

Affected Systems

- dashboard.example.com (203.0.113.5:443)

Description

The application `https://dashboard.example.com` has been found to have accounts with default credentials. Those allow attackers to easily compromise access controls by testing well known or easily guessable combinations. Due to default accounts often being highly privileged, this can have far-reaching consequences and gives attackers easy access to the application.

Recommendations

- Default accounts should be disabled.
- A strong **password policy** should be enforced, so users need to change passwords on first login/set custom passwords on signing up. The following characteristics for such a policy are recommended:
  - Passwords should be at least 14 characters long.
  - Passwords should consist of upper and lower case letters, numbers and special characters.
  - The password should not be a common password (e.g., sequence of numbers, sequence of letters, dictionary entry, etc.).
- It should be evaluated if other accounts are using default credentials too.

Technical Description

Default accounts are artifacts which usually originate from development or quality assurance teams, or from the initial setup of a system. Numerous systems require an already existing account to be able to complete the setup or test phase. For this, a manufacturer often creates an admin account with a trivial password. Such accounts can easily be enumerated for example by trying well-known combinations, analyzing the source code or even the application's binary file.

At the time of the assessment, the following usernames set up with a trivial password, were found on the application `https://dashboard.example.com` by trying known combinations:

- `admin`

Moreover, by gaining this authenticated access, unauthenticated attackers can also leverage the vulnerability described in 5.1 SQL Injection leading to Command Execution.

References

- https://www.owasp.org/index.php/Testing_for_default_credentials_(OTG-AUTHN-002)

5.3 Outdated Software with known Vulnerabilities

CVSS Score	8.4 (High)
CVSS Vector string	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H/E:U/RC:U (show in first.org)

Affected Systems

- srv02.example.com (203.0.133.8:443)

Description

At the time of the assessment, the `srv02.example.com` system appeared to be using outdated software that had at least the known vulnerability **CVE-2019-11043**. For the outdated version of **PHP** identified, there are already publicly available exploits. Attackers could use these exploits to access sensitive data on this system, make the system unavailable, or fully take over the system.

Recommendations

- It is recommended to update the outdated software as soon as possible to the latest stable version which includes the most recent security patches.
- A continuous update process should be established, which guarantees that security-critical updates can be installed quickly.
- If updates are not possible, the affected systems should be isolated and locked down to make access more difficult or impossible.

Technical Description

Using third-party software and services on a system requires keeping them updated. New security issues for publicly available software are discovered every day. These vulnerabilities are quickly known by attackers and exploits can be publicly available shortly after discovery. After the discovery of new vulnerabilities in their software, publishers usually release security patches in a timely manner. Installed versions should be updated on the system right after.

The server `srv02.example.com` is running an outdated version of **PHP** and **nginx**. The following table shows the versions detected by the scan and the latest ones that should be installed:

Software	Installed Version	Current Version
PHP	7.2.10	8.2.6
nginx	1.23.1	1.23.4

Table 5 - Software versions detected by the scan

If used with nginx and the FPM module, this version of PHP is known to be vulnerable to **CVE-2019-11043**. There are already published exploits available for vulnerabilities in this version. Attackers can use these exploits to perform remote code execution on the system. This major version of PHP has reached its end of life and is no longer supported. If new critical vulnerabilities are found in the future, no security patch will be available.

For more details on the outdated software products and associated vulnerabilities, see the **Nessus** vulnerability scan attached to this report.

References

- https://owasp.org/Top10/A06_2021-Vulnerable_and_Outdated_Components/
- https://nvd.nist.gov/vuln/detail/CVE-2019-11043

5.4 Open SMTP Server allows User Enumeration

CVSS Score	5.3 (Medium)
CVSS Vector string	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N (show in first.org)

Affected Systems

- mail.example.com (203.0.133.6:25)

Description

On `mail.example.com`, an SMTP server has been identified during the security assessment, on which it was possible to enumerate valid user accounts. Attackers could use the obtained information for follow-up attacks, such as a brute force attack.

Recommendations

- It is recommended to disable the `VRFY` command to avoid user enumeration.
- Other mail servers which were not in the scope of the assessment should be checked for user enumeration.
- Others command like `EXPN` and `RCPT TO` also lead to user enumeration and should be disallowed on all mail servers.

Technical Description

User enumeration is a vulnerability that allows attackers to guess valid user accounts. Scanning techniques are used to collect server responses which are then further analyzed to determine if a user account is valid or not. Attackers could then use the information gained in follow-up attacks, such as brute force attacks, to guess the passwords of identified user accounts.

During the security test, it was found that an SMTP server was running on the target `mail.example.com`. This SMTP server allowed the command `VRFY` for anonymous users. Attackers can use this command to check the existence of certain users on the system. The following listing shows which SMTP commands are supported/ allowed by the server:

```
$ nmap -p 25 mail.example.com --script=smtp-commands
Nmap scan report for mail.example.com (203.0.133.6)
Host is up (0.045s latency)
PORT STATE SERVICE VERSION
25/tcp open smtp Postfix smtp
|_ stmp-commands: mail.example.com (203.0.133.6), PIPELINING, SIZE 10248080, VRFY, ETRN,
ENHANCEDSTATUSCODES, 8BITMIME, DSN, CHUNKING
```

With automated tools, an attacker can easily check for common usernames:

```
$ nmap -p 25 mail.example.com --script=smtp-enum-users
Nmap scan report for mail.example.com (203.0.133.6)
Host is up (0.031s latency)
PORT STATE SERVICE
25/tcp open smtp
| smtp-enum-users:
|_ root
```

The listing above shows, that the user `root` has been detected to be a valid user on the mail server.

References

- https://www.rapid7.com/db/vulnerabilities/smtp-general-vrfy/

5.5 Remote Desktop Protocol (RDP) publicly accessible

CVSS Score	5.3 (Medium)
CVSS Vector string	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N (show in first.org)

Affected Systems

- srv01.example.com (203.0.133.2:3389)

Description

In the course of the testing, the remote maintenance service **Remote Desktop Protocol (RDP)** was detected on the host `srv01.example.com`. RDP has repeatedly been affected by serious security vulnerabilities in the past. Due to the high security risk, RDP should therefore not be directly exposed to the internet.

Recommendations

- It is strongly recommended not to expose sensitive services such as RDP directly to the internet because of the high security risk.
- If access to the RDP server is necessary over the internet, a **Virtual Private Network (VPN)** should be used to limit the network accessibility of the system.

Technical Description

The **Remote Desktop Protocol**, commonly referred to as **RDP**, is a proprietary protocol developed by Microsoft that allows a graphical connection to be made to a computer on the network. The history of RDP from a security perspective is versatile. Since 2002, there have been at least 20 Microsoft security updates specific to RDP, and at least 24 separate CVEs (see References).

During the course of the assessment, it was determined that the RDP remote service from host `srv01.example.com` is publicly accessible via the internet. The following listing illustrates the current state:

```
$ nmap -p 3389 srv01.example.com
Nmap scan report for srv01.example.com (203.0.133.2)
Host is up (0.0.12s latency).
PORT STATE SERVICE
3389/tcp open ms-wbt-server
Nmap done 1 IP adresse (1 host up) scanned in 0.84 seconds
```

References

- https://blog.rapid7.com/2017/08/09/remote-desktop-protocol-exposure/

5.6 Weak SSH Settings

CVSS Score	4.2 (Medium)
CVSS Vector string	CVSS:3.1/AV:N/AC:H/PR:N/UI:R/S:U/C:L/I:L/A:N (show in first.org)

Affected Systems

- relay.example.com (203.0.133.15:2222)

Description

Some affected systems were configured with weak **SSH settings**. Attackers with access to the network traffic could theoretically break the encryption and thus gain access to sensitive data such as usernames and passwords.

Recommendations

- It is recommended to support only secure and modern cipher, key exchange and MAC algorithms.
- If secure configuration (ciphers, key exchange, MAC algorithms) can't be applied, the system should not be directly exposed to the internet. A jump host with proper configuration could be used instead.
- The configuration of other SSH servers, outside the scope of this assessment, should also be examined.

Technical Description

At the time of the assessment, a system using weak **SSH settings** was identified.

The SSH server on `relay.example.com:2222` accepted the following **weak key exchange algorithms**:

- `diffie-hellman-group-exchange-sha1`
- `diffie-hellman-group1-sha1`

The server is also configured to accept several **weak encryption ciphers** that should be disabled in production environments:

- `3des-cbc`
- `aes128-cbc`
- `aes192-cbc`
- `aes256-cbc`
- `blowfish-cbc`
- `cast128-cbc`

Using man-in-the-middle attacks, connections negotiated based on insecure ciphers or using a weak key exchange algorithm could potentially be deciphered, exposing passwords and other sensitive information, which could lead to the compromise of the affected machine.

References

- https://infosec.mozilla.org/guidelines/openssh
- https://linuxhandbook.com/ssh-hardening-tips/
- https://sshcheck.com/
- https://www.sshaudit.com/

6 Appendix

6.1 Contact persons

A1 Digital International GmbH & Co KG

Name	Role	Telephone	Email
Alice Codex	Lead	+431234567890	ask.security@a1.digital
Bob Binary	Pentester	+431234567890	ask.security@a1.digital
Trent Trustworthy	Reviewer	+431234567890	ask.security@a1.digital

Table 6 - Contact persons at A1 Digital International GmbH & Co KG

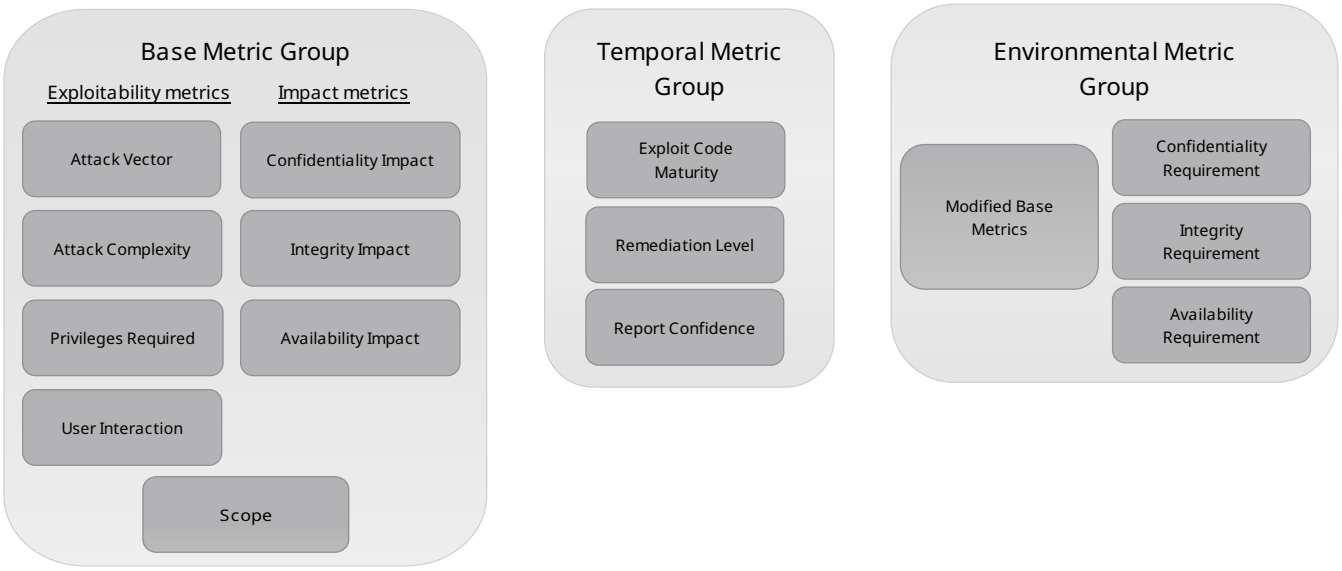
Example GmbH

Name	Telephone	Email
Jane Doe	+4312345678901	jd@example.com
Maximilian Muster	+4312345678902	mm@example.com

Table 7 - Contact persons at Example GmbH

6.2 CVSS v3.1 metrics

CVSS comprises three metric groups: **Base, Temporal** and **Environmental** as shown in the figure below:



Base Metric Group

The **Base Metric Group** expresses the fundamental risk of a weakness and assesses the vulnerable component. No valid CVSS value can be formed without a Base Metric. In turn the Base Metric is divided into Exploitability Metrics and Impact Metrics.

The **Exploitability Metric** reflects the ease and required pre-requisites for successful utilisation of the weakness.

The **Impact Metric** on the other hand reflects the direct consequence of the successful utilisation of the weak point - is the confidentiality, integrity or availability of the affected data/ of the affected system endangered?

Metric	Possible Values
Attack Vector (V) - attack vector	Network (N), Adjacent (A), Local (L), Physical (P)
Attack Complexity (AC) - attack complexity	Low (L), High (H)
Privileges Required (PR) - privileges required	None (N), Low (L), High (H)
User Interaction (UI) - required user interaction	None (N), Required (R)
Scope (S) - affected area	Changed (C), Unchanged (U)
Confidentiality Impact (C) - loss of confidentiality	None (N), Low (L), High (H)
Integrity Impact (I) - loss of integrity	None (N), Low (L), High (H)
Availability Impact (A) - loss of availability	None (N), Low (L), High (H)

Table 8 - Overview of Base Metric Group

Temporal Metric Group

The **Temporal Metric Group** expresses the characteristics of a weak point which may change over time. For example after some time an official patch may be published, which would reduce the Temporal Score.

Metric	Possible Values
Exploit Code Maturity (E) - degree of maturity of the exploit code present	Not Defined (X), High (H), Functional (F), Proof of Concept (P), Unproven (U)
Remediation Level (RL) - countermeasures present	Not Defined (X), Unavailable (U), Workaround (W), Temporal Fix (T), Official Fix (O)
Report Confidence (RC) - measures the reliability of the available information regarding the weakness	Not Defined (X), Confirmed (C), Reasonable (R), Unknown (U)

Table 9 - Overview of Temporal Metric Group

Environmental Metric Group

The **Environmental Metric Group** is specially set for the user environment. This metric allows the adaptation of the scores with respect to the importance of an affected system for the user/customer. The adjustment is done based on the requirements for confidentiality, integrity and availability.

Metric	Possible Values
Confidentiality Requirement (CR) - requirement for confidentiality	Network (N), Adjacent (A), Local (L), Physical (P)
Integrity Requirement (IR) - requirement for integrity	Low (L), High (H)
Availability Requirement (AR) - requirement for availability	None (N), Low (L), High (H)

Table 10 - Overview of Environmental Metric Group

Modified Base Metric Group

In addition, the base metrics can be shown as a modified value (modified base metric). This can be used to describe situations which increase the base score. For example a component could require multiple factors for authentication as standard (PR: High) in order to reach specific resources, whereas in the test environment no authentication was required (PR: None).

Metric	Possible Values
Modified Attack Vector (MAV)	The same values as the associated base metrics + not defined (N).
Modified Attack Complexity (MAC)	
Modified Privileges Required (MPR)	
Modified User Interaction (MUI)	
Modified Scope (MS)	
Modified Confidentiality (MC)	
Modified Integrity (MI)	
Modified Availability (MA)	

Table 11 - Overview of Modified Base Metric Group

Detailed information regarding the base, temporal and environmental metrics and their values are available on the first.org website. [2]

6.3 Text representation of CVSS v3.1 scores

In most cases it is helpful to have a text representation of the numerical CVSS scores. Each individual metric (Base, Temporal and Environmental) can be brought into text form using the following table. [3] [4]

Severity	CVSS Score
None	0.0
Low	0.1 - 3.9
Medium	4.0 - 6.9
High	7.0 - 8.9
Critical	9.0 - 10.0

Table 12 - Text representation of CVSS v3.1 scores

2. https://www.first.org/cvss/v3.1/specification-document
3. https://nvd.nist.gov/vuln-metrics/cvss
4. https://www.first.org/cvss/v3.1/specification-document#Qualitative-Severity-Rating-Scale

6.4 List of Tables

6.5 List of Figures

6.6 OWASP Testing Guide Version 4.2

Information Gathering
Conduct Search Engine Discovery Reconnaissance for Information Leakage (WSTG-INFO-01)
Fingerprint Web Server (WSTG-INFO-02)
Review Webserver Metafiles for Information Leakage (WSTG-INFO-03)
Enumerate Applications on Webserver (WSTG-INFO-04)
Review Webpage Content for Information Leakage (WSTG-INFO-05)
Identify Application Entry Points (WSTG-INFO-06)
Map Execution Paths Through Application (WSTG-INFO-07)
Fingerprint Web Application Framework (WSTG-INFO-08)
Fingerprint Web Application (WSTG-INFO-09)
Map Application Architecture (WSTG-INFO-10)

Configuration and Deployment Management Testing
Test Network Infrastructure Configuration (WSTG-CONF-01)
Test Application Platform Configuration (WSTG-CONF-02)
Test File Extensions Handling for Sensitive Information (WSTG-CONF-03)
Review Old Backup and Unreferenced Files for Sensitive Information (WSTG-CONF-04)
Enumerate Infrastructure and Application Admin Interfaces (WSTG-CONF-05)
Test HTTP Methods (WSTG-CONF-06)
Test HTTP Strict Transport Security (WSTG-CONF-07)
Test RIA Cross Domain Policy (WSTG-CONF-08)
Test File Permission (WSTG-CONF-09)
Test for Subdomain Takeover (WSTG-CONF-10)
Test Cloud Storage (WSTG-CONF-11)

Identity Management Testing
Test Role Definitions (WSTG-IDNT-01)
Test User Registration Process (WSTG-IDNT-02)
Test Account Provisioning Process (WSTG-IDNT-03)
Testing for Account Enumeration and Guessable User Account (WSTG-IDNT-04)
Testing for Weak or Unenforced Username Policy (WSTG-IDNT-05)

Authentication Testing
Testing for Credentials Transported over an Encrypted Channel (WSTG-ATHN-01)
Testing for Default Credentials (WSTG-ATHN-02)
Testing for Weak Lock Out Mechanism (WSTG-ATHN-03)
Testing for Bypassing Authentication Schema (WSTG-ATHN-04)
Testing for Vulnerable Remember Password (WSTG-ATHN-05)
Testing for Browser Cache Weaknesses (WSTG-ATHN-06)
Testing for Weak Password Policy (WSTG-ATHN-07)
Testing for Weak Security Question Answer (WSTG-ATHN-08)
Testing for Weak Password Change or Reset Functionalities (WSTG-ATHN-09)
Testing for Weaker Authentication in Alternative Channel (WSTG-ATHN-10)

Authorization Testing
Testing Directory Traversal File Include (WSTG-ATHZ-01)
Testing for Bypassing Authorization Schema (WSTG-ATHZ-02)
Testing for Privilege Escalation (WSTG-ATHZ-03)
Testing for Insecure Direct Object References (WSTG-ATHZ-04)

Session Management Testing
Testing for Session Management Schema (WSTG-SESS-01)
Testing for Cookies Attributes (WSTG-SESS-02)
Testing for Session Fixation (WSTG-SESS-03)
Testing for Exposed Session Variables (WSTG-SESS-04)
Testing for Cross Site Request Forgery (WSTG-SESS-05)
Testing for Logout Functionality (WSTG-SESS-06)
Testing Session Timeout (WSTG-SESS-07)
Testing for Session Puzzling (WSTG-SESS-08)
Testing for Session Hijacking (WSTG-SESS-09)

Input Validation Testing
Testing for Reflected Cross Site Scripting (WSTG-INPV-01)
Testing for Stored Cross Site Scripting (WSTG-INPV-02)
Testing for HTTP Verb Tampering (WSTG-INPV-03)
Testing for HTTP Parameter Pollution (WSTG-INPV-04)
Testing for SQL Injection (WSTG-INPV-05)
Testing for LDAP Injection (WSTG-INPV-06)
Testing for XML Injection (WSTG-INPV-07)
Testing for SSI Injection (WSTG-INPV-08)
Testing for XPath Injection (WSTG-INPV-09)
Testing for IMAP SMTP Injection (WSTG-INPV-10)
Testing for Code Injection (WSTG-INPV-11)
Testing for Command Injection (WSTG-INPV-12)
Testing for Format String Injection (WSTG-INPV-13)
Testing for Incubated Vulnerability (WSTG-INPV-14)
Testing for HTTP Splitting Smuggling (WSTG-INPV-15)
Testing for HTTP Incoming Requests (WSTG-INPV-16)
Testing for Host Header Injection (WSTG-INPV-17)
Testing for Server-side Template Injection (WSTG-INPV-18)
Testing for Server-Side Request Forgery (WSTG-INPV-19)

Testing for Error Handling
Testing for Improper Error Handling (WSTG-ERRH-01)
Testing for Stack Traces (WSTG-ERRH-02)

Testing for weak Cryptography
Testing for Weak Transport Layer Security (WSTG-CRYP-01)
Testing for Padding Oracle (WSTG-CRYP-02)
Testing for Sensitive Information Sent via Unencrypted Channels (WSTG-CRYP-03)
Testing for Weak Encryption (WSTG-CRYP-04)

Business Logic Testing
Test Business Logic Data Validation (WSTG-BUSL-01)
Test Ability to Forge Requests (WSTG-BUSL-02)
Test Integrity Checks (WSTG-BUSL-03)
Test for Process Timing (WSTG-BUSL-04)
Test Number of Times a Function Can Be Used Limits (WSTG-BUSL-05)
Testing for the Circumvention of Work Flows (WSTG-BUSL-06)
Test Defenses Against Application Misuse (WSTG-BUSL-07)
Test Upload of Unexpected File Types (WSTG-BUSL-08)
Test Upload of Malicious Files (WSTG-BUSL-09)

Client Side Testing
Testing for DOM-Based Cross Site Scripting (WSTG-CLNT-01)
Testing for JavaScript Execution (WSTG-CLNT-02)
Testing for HTML Injection (WSTG-CLNT-03)
Testing for Client-side URL Redirect (WSTG-CLNT-04)
Testing for CSS Injection (WSTG-CLNT-05)
Testing for Client-side Resource Manipulation (WSTG-CLNT-06)
Testing Cross Origin Resource Sharing (WSTG-CLNT-07)
Testing for Cross Site Flashing (WSTG-CLNT-08)
Testing for Clickjacking (WSTG-CLNT-09)
Testing WebSockets (WSTG-CLNT-10)
Testing Web Messaging (WSTG-CLNT-11)
Testing Browser Storage (WSTG-CLNT-12)
Testing for Cross Site Script Inclusion (WSTG-CLNT-13)

7 Imprint

A1 Digital International GmbH & Co KG

Business area: Machine-to-machine communication services, IT solutions, devices and other associated products and services
UID number: ATU 82193323

Representative persons:
Dr. Elisabetta Castiglioni (CEO)
Martin Schiffmann (CFO)

FB number: 654840a
Company legal jurisdiction: HG Vienna
Company headquarters: Vienna
Address: Lassallestraße 9, A-1020 Vienna
Contact details: Telephone: (+43) 5 06640; E-Mail: info@a1.digital
Chamber membership: Wirtschaftskammer Wien
Applicable legal regulations: Telecommunication laws: www.ris.bka.gv.at
Regulatory authority/commercial authorities: Österreichische Regulierungsbehörde für Rundfunk und Telekommunikation (RTR GmbH)