HACKTHEBOX

Penetration Test Report

HTB CPTS Format

Target Machine: Interpreter

Assessor: Samurai

February 22, 2026

Contents

1 Statement of Confidentiality

This document contains confidential and proprietary information pertaining to the security posture of the assessed environment. The information herein is intended solely for the authorized recipients. Any disclosure, copying, or distribution of this report without explicit permission is strictly prohibited.

2 Executive Summary

2.1 Assessment Overview

A penetration test was conducted against the target system designated as “Interpreter”. The objective of the engagement was to identify and exploit vulnerabilities to demonstrate the potential for a full system compromise.

The assessment resulted in a **full system compromise**, leading to administrative (`root`) access. The attack chain leveraged an unauthenticated Remote Code Execution (RCE) vulnerability in an exposed Mirth Connect service, followed by local information disclosure within configuration files, and ultimately a Server-Side Template Injection (SSTI) against an internal local service.

2.2 Summary of Findings

- **Critical**: Unauthenticated RCE in Mirth Connect (CVE-2023-43208)
- **Critical**: Server-Side Template Injection (SSTI) in internal API
- **High**: Hardcoded Database Credentials in Configuration Files

3 Internal Network Compromise Walkthrough

3.1 Initial Foothold: Mirth Connect Exploitation

During the initial reconnaissance, a Mirth Connect instance was discovered. Research indicated that older versions of Mirth Connect are vulnerable to CVE-2023-43208, an unauthenticated Remote Code Execution vulnerability.

An exploit script was utilized to successfully bypass authentication and execute arbitrary commands on the target. This granted an initial reverse shell under the context of the `mirth` service account.

3.2 Local Enumeration & Database Access

Operating as the `mirth` user, local enumeration was performed. Investigating the Mirth Connect installation directory (`/usr/local/mirthconnect/conf`) revealed a configuration file named `mirth.properties`. This file contained hardcoded MariaDB credentials:

```
database = mysql
database.username = mirthdb
database.password = MirthPass123!
```

mirth.properties snippet

These credentials were used to authenticate to the local MariaDB instance. Inspecting the `mc_bdd_prod` database yielded the password hash for the user `sedric`. Furthermore, dumping the `CHANNEL` configuration tables revealed an internal HTTP destination connector pointing to a local web application at `http://127.0.0.1:54321/addPatient`, expecting an XML payload.

3.3 Privilege Escalation: Local SSTI to Root

Interaction with the local `/addPatient` service revealed that the application parsed XML input and was vulnerable to Server-Side Template Injection (SSTI) within the `<firstname>` tag. The application utilized a Jinja2-style templating engine.

The developer had attempted to secure the input by implementing a regex filter: `^[a-zA-Z0-9._'"=+/]+$`. This filter effectively blocked spaces, preventing the execution of standard reverse shell commands.

To bypass this restriction, a Python PTY reverse shell was hex-encoded (containing only letters `a-f` and numbers). The payload was wrapped in a Python `exec()` function within the template tags to decode and run the hex string:

```
<patient>
    <firstname>{{exec(bytes.fromhex("...hex_payload...").decode())}}</firstname>
    <lastname>pwn</lastname>
</patient>
```

SSTI Hex-Encoded Bypass Payload

Sending this payload via `urllib.request` triggered the execution of the reverse shell, granting terminal access as the `root` user and securing full control over the machine.

4 Technical Findings Details

4.1 Finding 1: Unauthenticated RCE in Mirth Connect

Severity: Critical
Description: The application is running a version of Mirth Connect vulnerable to CVE-2023-43208, allowing unauthenticated attackers to execute arbitrary code.
Remediation: Update Mirth Connect to version 4.4.1 or later immediately.

4.2 Finding 2: Hardcoded Database Credentials

Severity: High
Description: The `mirth.properties` configuration file contained plaintext database credentials, allowing lateral movement into the backend MariaDB database.
Remediation: Utilize encrypted password vaults or environment variables to store sensitive credentials rather than plaintext files.

4.3 Finding 3: SSTI in Internal Patient API

Severity: Critical
Description: The internal `/addPatient` API evaluates user-supplied XML fields as templates. The implementation of a restrictive regex failed to prevent command execution, allowing for code execution via hex-encoded payloads.
Remediation: Do not use user input directly in template strings. Ensure that the templating engine is configured to treat user input strictly as data, not as executable code.

5 Appendix

5.1 A.1 Exploited Hosts

Host	Method	Notes
Interpreter	CVE-2023-43208 (Mirth Connect RCE)	Granted initial access as `mirth`.
Interpreter	Server-Side Template Injection (SSTI)	Granted local root access.

5.2 A.2 Compromised Users

Username	Type	Method
`mirth`	Service Account	Unauthenticated RCE via CVE-2023-43208.
`root`	Administrator	Local SSTI via hex-encoded regex bypass in internal API.