

# Project: CrowdSec – Windows Brute-force Enhanced Detection Test

🔗 GitHub Repo: FarahMae/crowdsec-windows-bf-enhanced-test  
🔧 Issue Name: Windows Brute-force detection enhancement (Issue #1235)

## 🔍 Introduction

This project contributes a test case for a real issue raised in the [CrowdSec Hub](#) — enhancing detection of brute-force login attempts from **Windows Security logs**. While CrowdSec supports many services out-of-the-box, some Windows event types (like failed logins) may not trigger detections due to missing parser logic or unverified log structures.

This project creates a **dedicated test case** using CrowdSec's **cscli hubtest** feature. It feeds a simulated but realistic Windows login failure log to the parser, checks the results, and helps confirm whether the current detection logic works as expected.

## 🔧 Step-by-Step Guide

### 1. Clone the CrowdSec Hub repo

```
git clone https://github.com/crowdsecurity/hub  
cd hub
```

### 2. Install CrowdSec manually (because Kali is unsupported by packagecloud)

```
wget  
https://github.com/crowdsecurity/crowdsec/releases/latest/download/crowdsec-rele  
ase.tgz tar -xvzf crowdsec-release.tgz cd crowdsec-v1.6.8 sudo ./wizard.sh
```

✓!! During setup, select the services to monitor (like SSH, SMB, MySQL, etc.), then  
>Select CrowdSec version

```
cscli version
```

Expected output:

```
version: v1.6.8-f209766e ...
```

### 4. List existing tests

```
sudo cscli hubtest list
```

### 5. Create a new test for the Windows Brute-force enhancement

```
sudo cscli hubtest create windows-bf-enhanced --type syslog
```

### 6. Fix Permission Denied issue while editing the log file

```
sudo nano .tests/windows-bf-enhanced/windows-bf-enhanced.log
```

Paste:

```
Feb 15 08:42:30 WIN-HOSTNAME Microsoft-Windows-Security-Auditing: An account  
failed to log on. Subject: Security ID: NULL SID Logon ID: 0x0 Account Name: -  
Account Domain: - Logon Type: 3 Account For Which Logon Failed: Security ID:  
NULL SID Account Name: testuser Workstation Name: WIN-HOSTNAME Failure  
Information: Failure Reason: Unknown user name or bad password Status:  
0xc000006d Sub Status: 0xc000006a
```

### 7. Edit config.yaml

```
sudo nano .tests/windows-bf-enhanced/config.yaml
```

Paste this correct and working configuration:

```
parsers:- crowdsecurity/syslog-logs- crowdsecurity/dateparse-enrich  
scenarios:postoverflows:- "" log_file: windows-bf-enhanced.loglog_type: syslog
```

✓ This is the fixed version you uploaded as  
[35cb7c8b-f5d3-49ed-879f-b2d502678645.yaml](#).

### 8. Run the test once to generate assertion suggestions

```
sudo cscli hubtest run windows-bf-enhanced
```

⚠!! You'll see output with suggested assertions like:

```
results["s00-raw"]["crowdsecurity/syslog-logs"][0].Evt.Parsed["program"] ==  
"Microsoft-Windows-Security-Auditing" ...
```

### 9. Fill the generated parser.assert file

```
sudo nano .tests/windows-bf-enhanced/parser.assert
```

Example contents:

```
len(results) == 2 len(results["s00-raw"]["crowdsecurity/syslog-logs"]) == 1  
results["s00-raw"]["crowdsecurity/syslog-logs"][0].Success == true  
results["s00-raw"]["crowdsecurity/syslog-logs"][0].Evt.Parsed["program"] ==  
"Microsoft-Windows-Security-Auditing" ...
```

## 🔧 Make sure to fix this line:

```
-  
basename(results["s00-raw"]["crowdsecurity/syslog-logs"][0].Evt.Meta["datasource  
_path"]) == "/home/kali/..."+  
results["s00-raw"]["crowdsecurity/syslog-logs"][0].Evt.Meta["datasource_path"]  
== "windows-bf-enhanced.log"
```

### 10. Add empty scenario.assert

```
sudo nano .tests/windows-bf-enhanced/scenario.assert
```

Content:

```
len(results) == 0
```

### 11. Run final test to confirm success

```
sudo cscli hubtest run windows-bf-enhanced
```

✓ Output should show:

```
All tests passed, use --report-success for more details.
```

## ✓ Conclusion

This project successfully simulates a CrowdSec test case to verify the parsing of **Windows Security logs** related to brute-force login attempts. The test uses realistic event entries and confirms that the current syslog and enrichment parsers correctly process the log line structure.

Although the default parser does not yet trigger a scenario on this log, this test lays the foundation for enhancing future detection logic. It provides reproducibility for maintainers and supports [Issue #1235](#) in the official CrowdSec repository.

## 💡 Recommendations

- 🔧 **Parser Improvement:** Extend the Windows parser to extract fields like **Account Name**, **Logon Type**, and **Status** to enable brute-force detection logic.
- 🔧 **Scenario Contribution:** Based on the parsed output, a custom scenario YAML file can be added to match patterns like repeated **0xc000006d** login failures.
- 📦 **PR Opportunity:** Once the parser and scenario enhancements are made, this test can be included as a validation unit test in the core hub.
- 📁 **More Logs:** Add additional log samples including different logon types (e.g. RDP, network, interactive) for broader coverage.