Windows Protocol Test Suites
Overview and Updates

Mengyan (Helen) Lu     Jiajun Wang
Microsoft
Agenda

Protocol Test Suite Overview

Protocol Test Suite Open Source

Linux Integration with Test Suite Automation

Test Environment on Azure
What is a Windows Protocol Test Suite

Evaluates whether a protocol implementation meets certain interoperability requirements.

- Originally developed for in-house testing of Microsoft Open Specifications.
- Used to test/verify Windows behavior.
- Also used to test 3rd-party implementations.

Do not cover every protocol requirement, and do not certify an implementation, but can be a useful indication of interoperability.
Why we developed Protocol Test suites

Ensure quality of Microsoft Open Specifications

- Accuracy and Usability

Help to identify your implementation issues

- Make debugging easier
  Message sequence, encrypted messages logging
- Test mainstream scenarios

Source code in test suite can be your reference in development phase
Protocol Test Methodology

1. Open Specification defines messages, sequences, behaviors
2. Develop protocol test suite (synthetic client) according to Open Specification
3. Run against Windows to verify Open Specification
4. Run against 3rd-party implementation to help identify & debug issues
5. Interoperability between partner’s server and Windows client

Server-side implementation

Windows Client

Synthetic Server

Test Suite

Windows Server

Partner’s Server

Message Over the Wire

Open Specification

Microsoft
Example Test Case

Protocol Specification:
If SMB2 client sends message A to SMB2 server, SMB2 server should respond with message B.
If message A is invalid, server should return error message C.

Test Case 1:
1. Establish connection;
2. Client.Send\( (msgA) \);
3. Expect server respond with \( msgB \), and
   •  Assert \( (msgB.Field1 == valid1) \)
   •  Assert \( (msgB.Field2 == valid2) \)
   •  ...
   •  Assert \( (msgB.FieldN == validN) \)

Test Case 2:
1. Establish connection;
2. Client.Send\( (invalid\ msgA) \);
3. Expect server respond with \( msgC, \ and \)
   •  Assert \( (msgC.Status == ERROR) \)
## Windows Protocol Test Suite Types

<table>
<thead>
<tr>
<th>Type</th>
<th>Test Suite Name</th>
<th>Protocols</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protocol Families</td>
<td><strong>FileServer</strong></td>
<td>MS-SMB2, MS-FSRVP, MS-SWN, MS-DFSC, MS-RSVD, MS-SQOS, MS-FSA, MS-HVRS</td>
</tr>
<tr>
<td></td>
<td><strong>Kerberos</strong></td>
<td>MS-KILE, MS-KKDCP, MS-PAC</td>
</tr>
<tr>
<td>RDP (Client &amp; Server)</td>
<td><strong>RDP Client TestSuite</strong></td>
<td>MS-RDPBCGR, MS-RDPEDISP, MS-RDPEGFX, MS-RDPEGT, MS-RDPEI, MS-RDPEMT, MS-RDPEUDP, MS-RDPEUSB, MS-RDPEVOR, MS-RDPRFX</td>
</tr>
<tr>
<td></td>
<td><strong>RDP Server TestSuite</strong></td>
<td>MS-RDPBCGR, MS-RDPEMT</td>
</tr>
<tr>
<td></td>
<td><strong>ADFamily</strong></td>
<td>MS-ADTS-LDAP, MS-ADTS-PublishDC, MS-ADTS-Schema, MS-ADTS-Security, MS-APDS, MS-DRSR, MS-FRS2, MS-LSAD, MS-LSAT, MS-NRPC, MS-SAMR</td>
</tr>
<tr>
<td></td>
<td><strong>BranchCache</strong></td>
<td>MS-PCCRC, MS-PCCRR, MS-PCCRTP, MS-PCHC, MS-CCROD</td>
</tr>
<tr>
<td>Individual Protocols</td>
<td><strong>MS-SMB</strong></td>
<td>MS-SMB</td>
</tr>
<tr>
<td></td>
<td><strong>MS-SMBD</strong></td>
<td>MS-SMBD</td>
</tr>
<tr>
<td></td>
<td><strong>MS-ADFSPIP</strong></td>
<td>MS-ADFSPIP</td>
</tr>
<tr>
<td>Overview Documents</td>
<td><strong>MS-ADOD</strong></td>
<td>MS-ADOD</td>
</tr>
<tr>
<td></td>
<td><strong>MS-AZOD</strong></td>
<td>MS-AZOD</td>
</tr>
</tbody>
</table>
Protocol Test Framework, Protocol Test Manager

- Protocol Test Framework
  - PTF Adapter
  - PTF Config
  - Logging
  - ...

- Protocol Test Manager
  - FileServer Test Suite
  - ADFamily Test Suite
  - ...

Visual Studio Unit Test Framework

PTM CLI: a command line interface for PTM
https://github.com/Microsoft/WindowsProtocolTestSuites/wiki/PtmCli

PtmCli.exe -p [-profile profileName]
[-s | -selected] [-r | -report reportFile]
[-categories categories] [-outcome pass, fail, inconclusive, notrun]
[-sortby name|outcome] [-separator comma|space]
Agenda

- Protocol Test Suite Overview
- Protocol Test Suite Open Source
- Linux Integration with Test Suite Automation
- Test Environment on Azure
Test Suite Open Source

- Open sourced on GitHub in 2016
- Two formats: source code and MSI file
- **MIT License**: Short and permissive

- 11 Test Suites (family)
- 45 Windows Protocols
- Protocol Test Framework
- Protocol Test Manager
Build Test Suites

• Clone code from Github
• Install Prerequisites Software
  ```
  cd WindowsProtocolTestSuites\InstallPrerequisites
  \InstallPrerequisites.ps1 -Category 'FileServer'
  ```
• Build Protocol Test Framework
• Build Test Suite

Check more details on GitHub: [WindowsProtocolTestSuites](https://github.com/WindowsProtocolTestSuites) / README.md
Build an Active Community

Issues
Pull Requests
Feature Requests

New Features
• Log an issue before working on the feature

Update Documentation
• User Guide
• Test Design Specification

New Test Cases
• Add new test cases

Report Issues
• Report issues
• Bug fix
• Verify fix
Agenda

Protocol Test Suite Overview

Protocol Test Suite Open Source

Linux Integration with Test Suite Automation

Test Environment on Azure
Linux Integration

• Trigger Test Suites Remotely
• Control Linux SUT in Test Suite
PowerShell Core

- Cross Platform (Windows, Linux and macOS).
- SSH Transport.
- Public Key Authentication.
Trigger Test Suites Remotely

https://github.com/Microsoft/WindowsProtocolTestSuites/wiki/Remotely-Trigger-Test-Suites
SUT Control Adapter

• Used to control SUT status during testing.

• Currently support:
  • PowerShell
  • CMD
  • Interactive
Example: Failover Cluster

Disable Node Using Windows API

Test Suite

Domain Controller

Node 1

Failover

Node 2

Cluster Storage
Example: Failover Cluster
Example: Failover Cluster

- Test Suite
- Domain Controller
- Node 1
- Node 2
- Cluster Storage
Example: Failover Cluster

- **Test Suite**
- **Domain Controller**
- **Node 1**
- **Node 2**
- **Cluster Storage**
- **Failover**

- **SUT Control Adapter**
- **Linux Script over PowerShell Core**
Agenda

- Protocol Test Suite Overview
- Protocol Test Suite Open Source
- Linux Integration with Test Suite Automation
- Test Environment on Azure
Network Topology