Tai-hoon Kim   Hojjat Adeli
Haeng-kon Kim   Heau-jo Kang
Kyung Jung Kim   Akingbehin Kiumi
Byeong-Ho Kang (Eds.)

Communications in Computer and Information Science 257

Software Engineering, Business Continuity, and Education

International Conferences ASEA, DRBC and EL 2011,
Held as Part of the Future Generation
Information Technology Conference, FGIT 2011,
in Conjunction with GDC 2011,
Jeju Island, Korea, December 2011, Proceedings

Springer

A Test Management System for Operational Validation

Myoung Wan Kim[1], Woo Yeol Kim[2],
Hyun Seung Son[2], and Robert Young Chul Kim[2]

[1] Institute of Technology, Infnis, Inc.,
Seoul, 135-080, Korea
kimmw@infnis.com

[2] Dept. of CIC(Computer and Information Communication), Hongik University,
Jochiwon, 339-701, Korea
{john,son,bob}@selab.hongik.ac.kr

Abstract. NMS (Network Management System) is a central monitoring system that can manage equipment on network environments. This should be used for efficient and centralized management of network equipment. On NMS, it enables the real-time transmission and monitoring of the data on states, problems, composition, and statistics of equipment that make a network. But we need to verify whether it works on operations or functions of NMS operations or not. To do this, this paper suggests a test management system for the efficient verification of NMS environments. In order to develop a test management system, requirements from each NMS shall be extracted, and designed and materialized based on them. The suggested system enables efficient test management, result analysis, and comparative verification of test versions.

Keywords: NMS, test, test management system, verification.

1 Introduction

NMS (Network Management System) is a central monitoring system that can manage equipment on a network. This is used for efficient and centralized management of network equipment [1,2]. NMS enables to the real-time transmission and monitoring of the data on states, problems, composition, and statistics of equipment that make a network. When problems with the equipment are occurred, an alarm signal can be transmitted to a manager for a speedy measurement. Statistics and states of networks can also be analyzed based on the collected information [3].

Figure 1 shows the network of NSM. Each company manages its own network, which should make NMS system test system requirements, functions, and operations.

In order to verify NSM of various environments, the existing test has been conducted manually via a checklist with the managed results based on documentation [4,5,6]. There are the problems with manual operation of the test results in that it is difficult to integrate test results as various testers conduct testing simultaneously because the analysis and management of the test results does not appropriately executed. The version management of test cases is also not sufficient and the reuse of test cases is difficult.

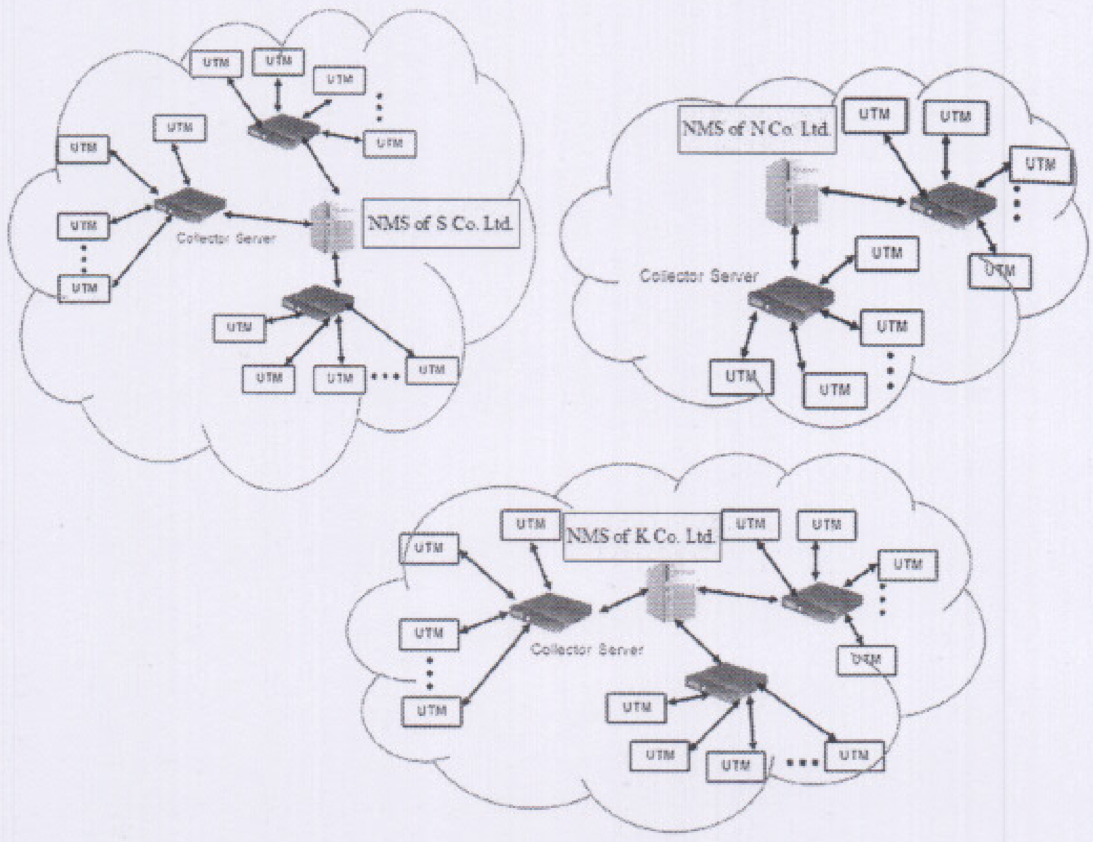


Fig. 1. Composition of NSM

This paper suggests test management system to solve such problems. In order to develop a test management system, each NMS environment is analyzed, which helps to extract requirements. Based on the extracted requirements, a test management system is designed and managed. The suggested system enables efficient test management, result analysis, and comparative verification of test versions. The currently used NMS tests were conducted as a case study and the results were applied to the suggested test management system.

This paper consists of the following. Chapter 2 explains about NMS as related studies. Chapter 3 describes the suggested test management system. Chapter 4 shows the application system of this suggested test management system as an applied case study. Finally, Chapter 5 provides conclusions and describes the future studies.

2 Network Management System

NMS centrally monitors communication networks on the network [7]. NMS consists of NMS servers providing main functions, which collect receiving data from equipment, sending them to NMS Server, and saving data to DB server. NMS is organized as Figure 2. Each UTM equipment on the network transmits system monitoring data, alarm data related to problems, log data, and equipment registration data to NMS system via collector server while NMS Server provides central management of the related network.

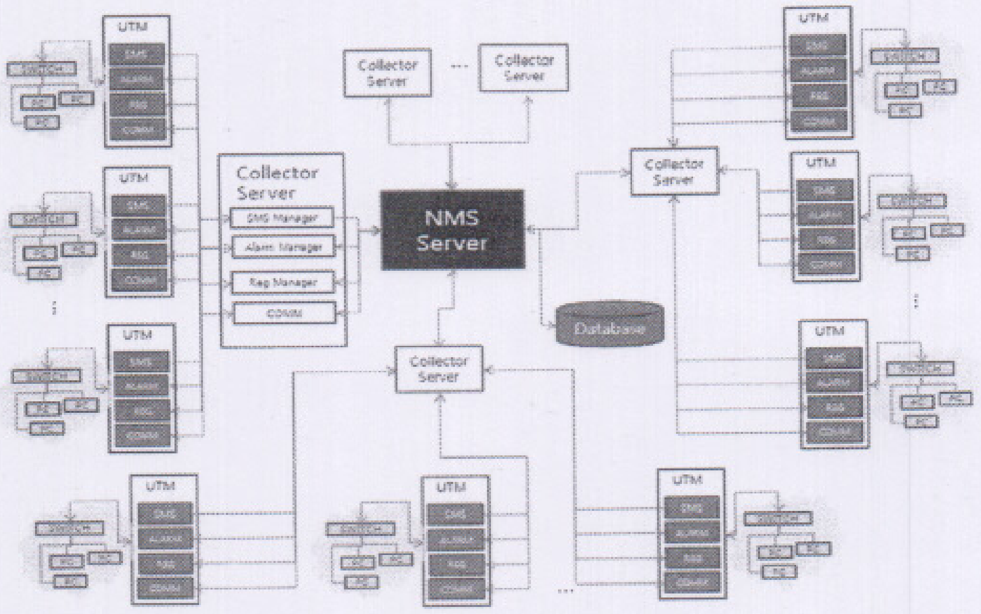


Fig. 2. NMS organization

NMS Server can centrally manage network equipment based on the information on UTM equipment states, problems, and log received from Collector Server. It provides with user management UI for managing Collector Server and UTM equipment. It analyzes the data received from UTM equipment and provides statistical information. Based on the collected and analyzed data, it organizes the status of interface links of UTM equipment and IPSEC tunnel links as well as the monitoring screen, generating a report.

Individual connecttion to each UTM equipment can bring the information of setup of interface, routing, NAT, firewall, IPS, web filter, and content filter and the setup change is available. Multiple UTM equipments can be grouped to order firewall rules, group IPS rules, group web filter rules, and content filter rules simultaneously.

3 Proposed Test Management System

3.1 The Test Management System Architecture

Figure 3 is architecture of the suggested test management system. The values to be fed firstly in the test management system include that the version information, types of test manuals, and test cases are extracted through analysis. The user authorization is implemented via the account management module. After authorization, the information is managed in the TESTID management mode.

After the test cases are extracted through analyses, the relevant test cases are tested while the results are managed through the test manual management module. The test manual management module can manage preconditions, testing steps, and actual results. Finally, the testing for test results is managed in the test result management module, which provides the functions such as the result document printing and preview.

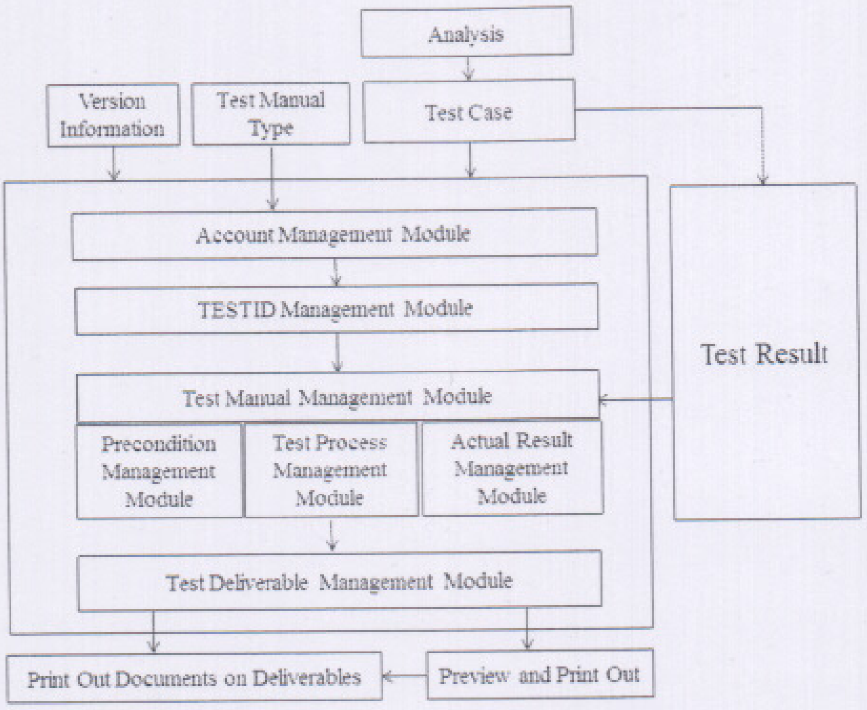

Analysis
Version Information
Test Manual Type
Test Case
Account Management Module
TESTID Management Module
Test Manual Management Module
Precondition Management Module
Test Process Management Module
Actual Result Management Module
Test Deliverable Management Module
Test Result
Print Out Documents on Deliverables
Preview and Print Out

Fig. 3. Test Management System Architecture

3.2 Integrated data model

The integrated data model is to manage the data generated in the test management system. Figure 4 is an expression of this integrated data model in the E-R diagram.

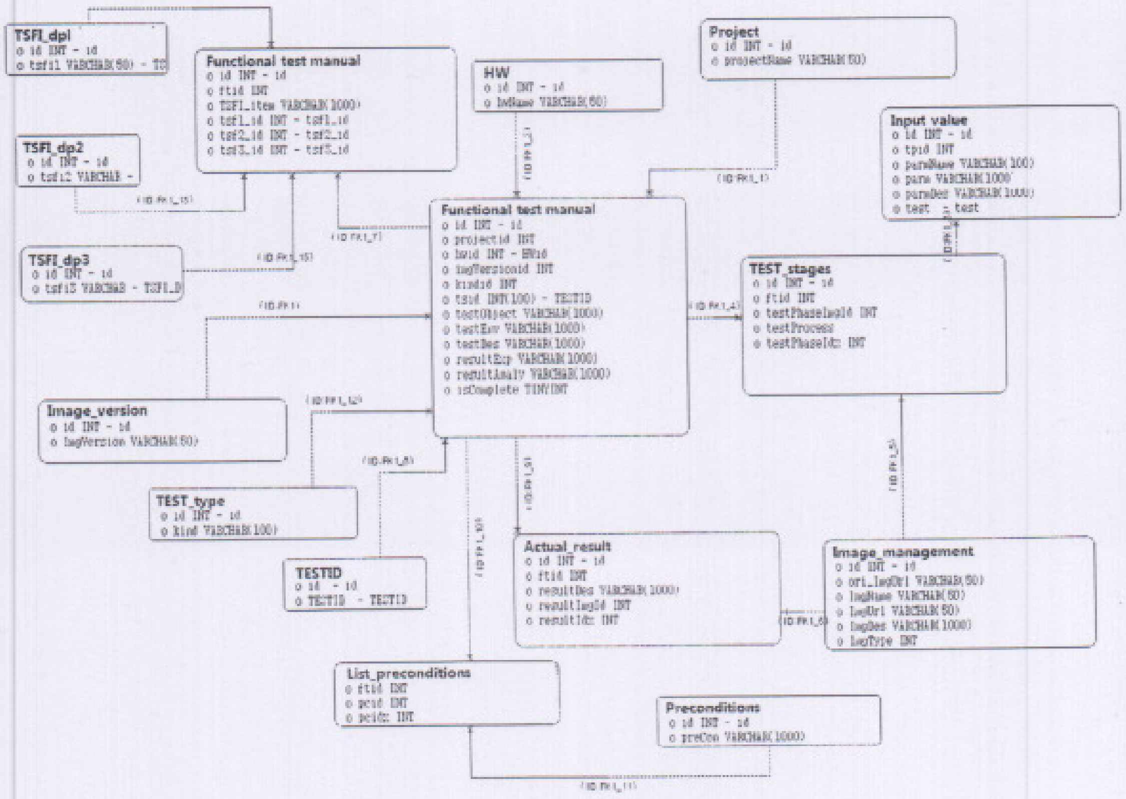
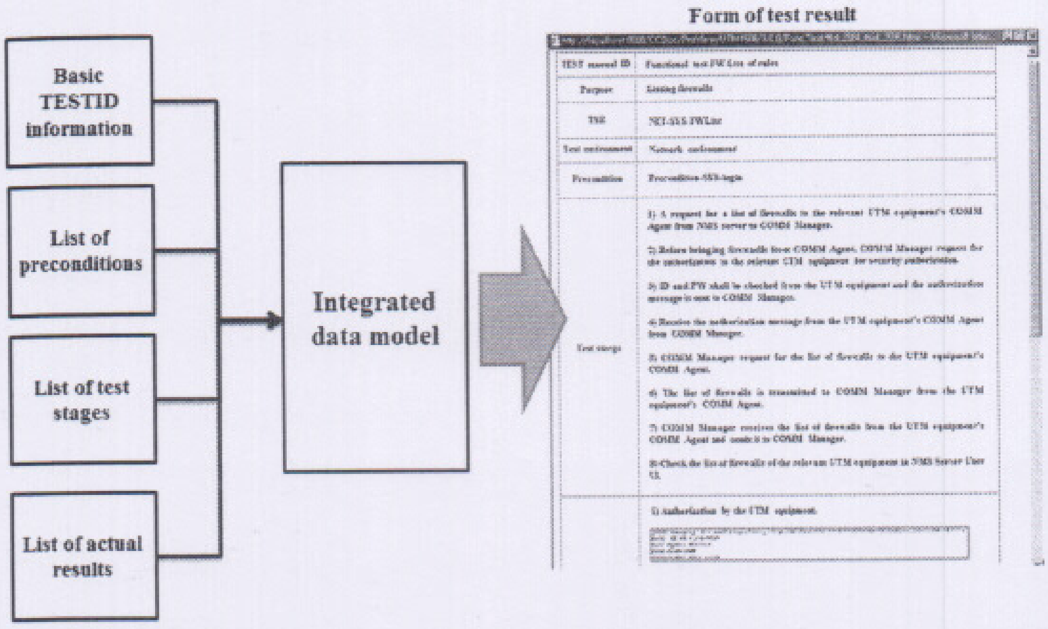


Fig. 4. Integrated data model for a test management system

The integrated data model integrates the basic TESTID information, list of preconditions, list of testing stages, and list of actual results, enabling the printing of test result forms through the test manual management module and the test deliverables management module in Figure 5.



TEST manual ID	Functional test FW List of rules
Purpose	Listing firewalls
TSE	NET-SYS FWList
Test environment	Network environment
Precondition	Precondition-SSL-login
Test stages	1) A request for a list of firewalls in the relevant UTM equipment's COMM Agent from NMS server to COMM Manager. 2) Before bringing firewalls from COMM Agent, COMM Manager request for the authorization to the relevant UTM equipment for security authorization. 3) ID and PW shall be checked from the UTM equipment and the authorization message is sent to COMM Manager. 4) Receive the authorization message from the UTM equipment's COMM Agent from COMM Manager. 5) COMM Manager request for the list of firewalls to the UTM equipment's COMM Agent. 6) The list of firewalls is transmitted to COMM Manager from the UTM equipment's COMM Agent. 7) COMM Manager receives the list of firewalls from the UTM equipment's COMM Agent and sends it to COMM Manager. 8) Check the list of firewalls of the relevant UTM equipment in NMS Server User UI.
	1) Authorization by the UTM equipment. [illegible]

Fig. 5. Generation of the test results

The specifications on the basic TESTID information, list of preconditions, list of testing stages, and list of actual results were defined while the Pseudo Code of data-integrating algorithm in Figure 6 was prepared based on the defined classification codes. The data-integrating algorithm can lead to the extraction of test results.

```
String report = A1 + A2 + A5 + A3 + A4     } Basic TESTID information
If( B1.length > 0 ){
      for( B1.length ){
report += B1                               } Information on preconditions
      }
}

If( C1.length > 0 ){
       for( C1.length ){
              report += C2 + C1
              If( D1.length > 0 ){
              for( D1.length ){
               report += D3 + D2 + D1      } Test stage and input values
               }
          }
       }
}

report += A7                               } Expected results

If( E1.length > 0 ){
        for( E1.length ){
             report += E2 + E1 + E3        } Actual results
        }
}

Report += A8                               } Analysis of test information
```

Fig. 6. Data-integrating algorithm

4 NMS Verification Using the Test Management System

This chapter explores the functions of the developed test management system and verifies by it by using the test management system and carrying out NSM tests based on the actual test cases.

The followings take place to verify the test management system.

1) COMM Manager testing
2) Feeding the test results in the test management system
3) Confirming the printing of test results in the test management system

4.1 COMM Manager Testing

In order to bring the firewall rule setup list from NMS Server, the firewall rule list is transmitted based on the communication between COMM Manager in Collector Server and COMM Agent in UTM equipments. Afterwards, the test confirmation in NMS Server user's UI will be carried out.

1) Testing a request for the list of firewall rules to COMM Agent of the relevant UTM equipment from NMS Server to COMM Manager

```
[COMM Manager]  FilterCmd called!!!
[COMM Manager]  new create FWPolicyInfo called!!!
```

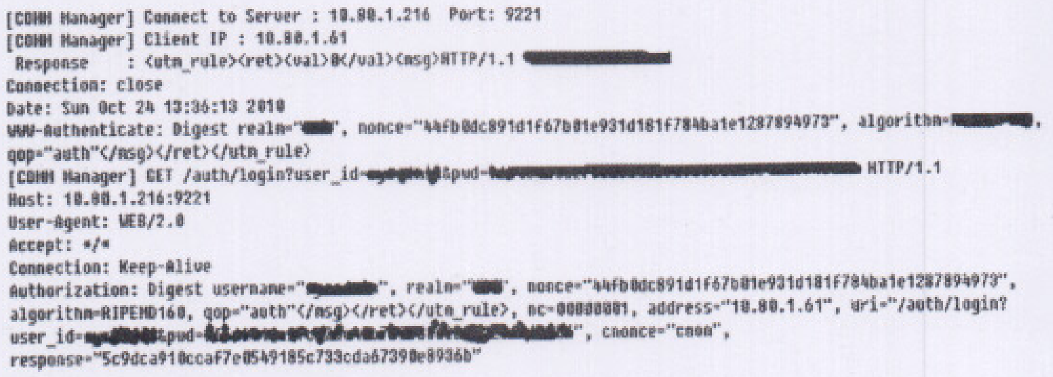
2) Before importing the firewall rules from COMM Agent, testing an authorization request from COMM Manager to the relevant UTM equipment for security authorization

```
[COMM Manager]  GET /auth/login?user_id=[redacted]&pwd=[redacted] HTTP/1.1
Host: 10.80.1.216:9221
User-Agent: WEB/2.0
Keep-Alive:300
Connection: keep-alive
```

3) ID and PW to be checked in the UTM equipment and testing the transmission of authorization messages to COMM Manager

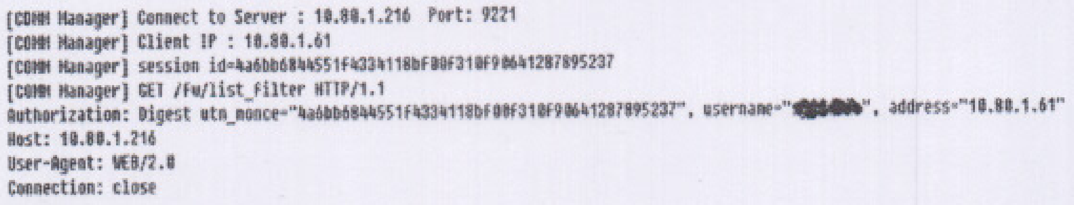
```
13:40:37.595659 IP 10.80.1.61.2637 > 10.80.1.216.ipcs: S 601707624:601707624(0) win 65535 <ms
s 1460,nop,nop,sackOK>
13:40:37.595760 IP 10.80.1.216.ipcs > 10.80.1.61.2637: S 1256001678:1256001678(0) ack 6017076
25 win 5840 <mss 1460,nop,nop,sackOK>
13:40:37.595758 IP 10.80.1.61.2637 > 10.80.1.216.ipcs: . ack 1 win 65535
13:40:37.596807 IP 10.80.1.61.2637 > 10.80.1.216.ipcs: P 1:101(100) ack 1 win 65535
13:40:37.596834 IP 10.80.1.216.ipcs > 10.80.1.61.2637: . ack 101 win 5840
13:40:37.607723 IP 10.80.1.216.ipcs > 10.80.1.61.2637: P 1:855(854) ack 101 win 5840
13:40:37.610874 IP 10.80.1.61.2637 > 10.80.1.216.ipcs: P 101:240(139) ack 855 win 64681
13:40:37.654447 IP 10.80.1.216.ipcs > 10.80.1.61.2637: . ack 240 win 6432
13:40:37.655428 IP 10.80.1.61.2637 > 10.80.1.216.ipcs: P 240:283(43) ack 855 win 64681
13:40:37.655581 IP 10.80.1.216.ipcs > 10.80.1.61.2637: . ack 283 win 6432
13:40:37.655970 IP 10.80.1.216.ipcs > 10.80.1.61.2637: P 855:898(43) ack 283 win 6432
13:40:37.657562 IP 10.80.1.61.2637 > 10.80.1.216.ipcs: P 283:479(196) ack 898 win 64638
13:40:37.671176 IP 10.80.1.216.ipcs > 10.80.1.61.2637: P 898:1121(223) ack 479 win 7504
13:40:37.671446 IP 10.80.1.216.ipcs > 10.80.1.61.2637: FP 1121:1144(23) ack 479 win 7504
13:40:37.673730 IP 10.80.1.61.2637 > 10.80.1.216.ipcs: . ack 1145 win 64392
13:40:37.685095 IP 10.80.1.61.2637 > 10.80.1.216.ipcs: P 479:502(23) ack 1145 win 64392
13:40:37.685105 IP 10.80.1.61.2637 > 10.80.1.216.ipcs: F 502:502(0) ack 1145 win 64392
13:40:37.686266 IP 10.80.1.216.ipcs > 10.80.1.61.2637: . ack 503 win 7504
13:40:37.686212 IP 10.80.1.61.2637 > 10.80.1.216.ipcs: R 601708127:601708127(0) win 0
13:40:37.688717 IP 10.80.1.61.2639 > 10.80.1.216.ipcs: S 2759155723:2759155723(0) win 65535 <
mss 1460,nop,nop,sackOK>
13:40:37.688776 IP 10.80.1.216.ipcs > 10.80.1.61.2639: S 1250485726:1250485726(0) ack 2759155
724 win 5840 <mss 1460,nop,nop,sackOK>
```

4) Testing an authorization message receipt from COMM Agent of the UTM equipment in COMM Manager



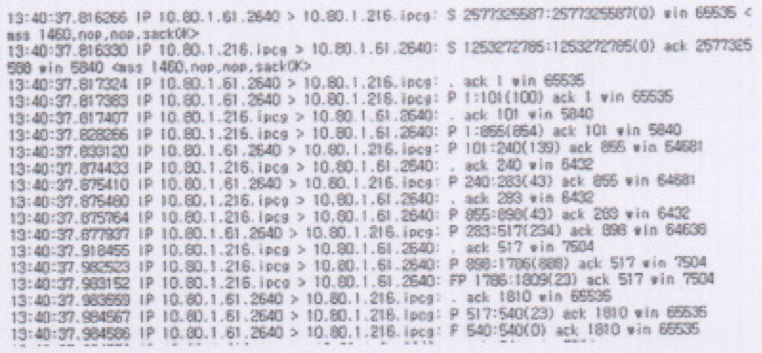
```
[COMM Manager] Connect to Server : 10.80.1.216  Port: 9221
[COMM Manager] Client IP : 10.80.1.61
 Response    : <utm_rule><ret><val>0</val><msg>HTTP/1.1 [redacted]
Connection: close
Date: Sun Oct 24 13:36:13 2010
WWW-Authenticate: Digest realm="[redacted]", nonce="44fb0dc891d1f67b01e931d181f784ba1e1287894973", algorithm=[redacted],
qop="auth"</msg></ret></utm_rule>
[COMM Manager] GET /auth/login?user_id=[redacted]&pwd=[redacted] HTTP/1.1
Host: 10.80.1.216:9221
User-Agent: WEB/2.0
Accept: */*
Connection: Keep-Alive
Authorization: Digest username="[redacted]", realm="[redacted]", nonce="44fb0dc891d1f67b01e931d181f784ba1e1287894973",
algorithm=RIPEMD160, qop="auth"</msg></ret></utm_rule>, nc=00000001, address="10.80.1.61", uri="/auth/login?
user_id=[redacted]&pwd=[redacted]", cnonce="cnon",
response="5c9dca910ccaf7e0549185c733cda67390e8936b"
```

5) Testing a request for firewall rules from COMM Manager to the UTM equipment's COMM Agent



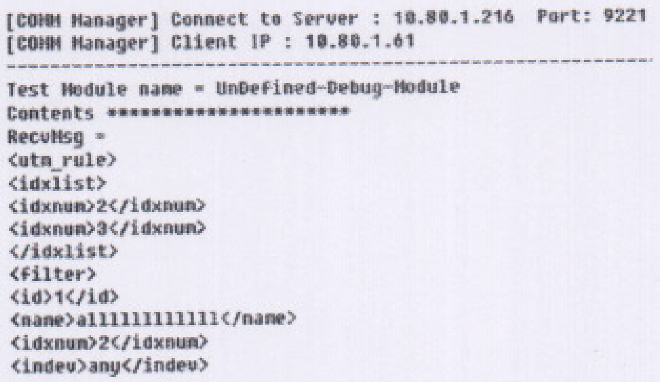
```
[COMM Manager] Connect to Server : 10.80.1.216  Port: 9221
[COMM Manager] Client IP : 10.80.1.61
[COMM Manager] session id=4a6bb6844551f4334118bf00f310f90641287895237
[COMM Manager] GET /fw/list_filter HTTP/1.1
Authorization: Digest utm_nonce="4a6bb6844551f4334118bf00f310f90641287895237", username="[redacted]", address="10.80.1.61"
Host: 10.80.1.216
User-Agent: WEB/2.0
Connection: close
```

6) Testing the transmission of the firewall list from the UTM equipment's COMM Agent to COMM Manager



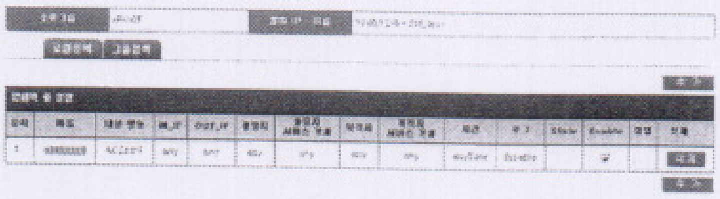
```
13:40:37.816266 IP 10.80.1.61.2640 > 10.80.1.216.ipcg: S 2577325587:2577325587(0) win 65535 <
mss 1460,nop,nop,sackOK>
13:40:37.816330 IP 10.80.1.216.ipcg > 10.80.1.61.2640: S 1253272785:1253272785(0) ack 2577325
588 win 5840 <mss 1460,nop,nop,sackOK>
13:40:37.817324 IP 10.80.1.61.2640 > 10.80.1.216.ipcg: . ack 1 win 65535
13:40:37.817383 IP 10.80.1.61.2640 > 10.80.1.216.ipcg: P 1:101(100) ack 1 win 65535
13:40:37.817407 IP 10.80.1.216.ipcg > 10.80.1.61.2640: . ack 101 win 5840
13:40:37.828266 IP 10.80.1.216.ipcg > 10.80.1.61.2640: P 1:855(854) ack 101 win 5840
13:40:37.833120 IP 10.80.1.61.2640 > 10.80.1.216.ipcg: P 101:240(139) ack 855 win 64681
13:40:37.874433 IP 10.80.1.216.ipcg > 10.80.1.61.2640: . ack 240 win 6432
13:40:37.875410 IP 10.80.1.61.2640 > 10.80.1.216.ipcg: P 240:283(43) ack 855 win 64681
13:40:37.875480 IP 10.80.1.216.ipcg > 10.80.1.61.2640: . ack 283 win 6432
13:40:37.875764 IP 10.80.1.216.ipcg > 10.80.1.61.2640: P 855:898(43) ack 283 win 6432
13:40:37.877937 IP 10.80.1.61.2640 > 10.80.1.216.ipcg: P 283:517(234) ack 898 win 64638
13:40:37.918455 IP 10.80.1.216.ipcg > 10.80.1.61.2640: . ack 517 win 7504
13:40:37.982523 IP 10.80.1.216.ipcg > 10.80.1.61.2640: P 898:1786(888) ack 517 win 7504
13:40:37.983152 IP 10.80.1.216.ipcg > 10.80.1.61.2640: FP 1786:1809(23) ack 517 win 7504
13:40:37.983669 IP 10.80.1.61.2640 > 10.80.1.216.ipcg: . ack 1810 win 65535
13:40:37.984567 IP 10.80.1.61.2640 > 10.80.1.216.ipcg: P 517:540(23) ack 1810 win 65535
13:40:37.984586 IP 10.80.1.61.2640 > 10.80.1.216.ipcg: F 540:540(0) ack 1810 win 65535
```

7) After receiving the firewall rules from COMM Manager to the UTM equipment COMM Agent, testing the transmission to NMS Server



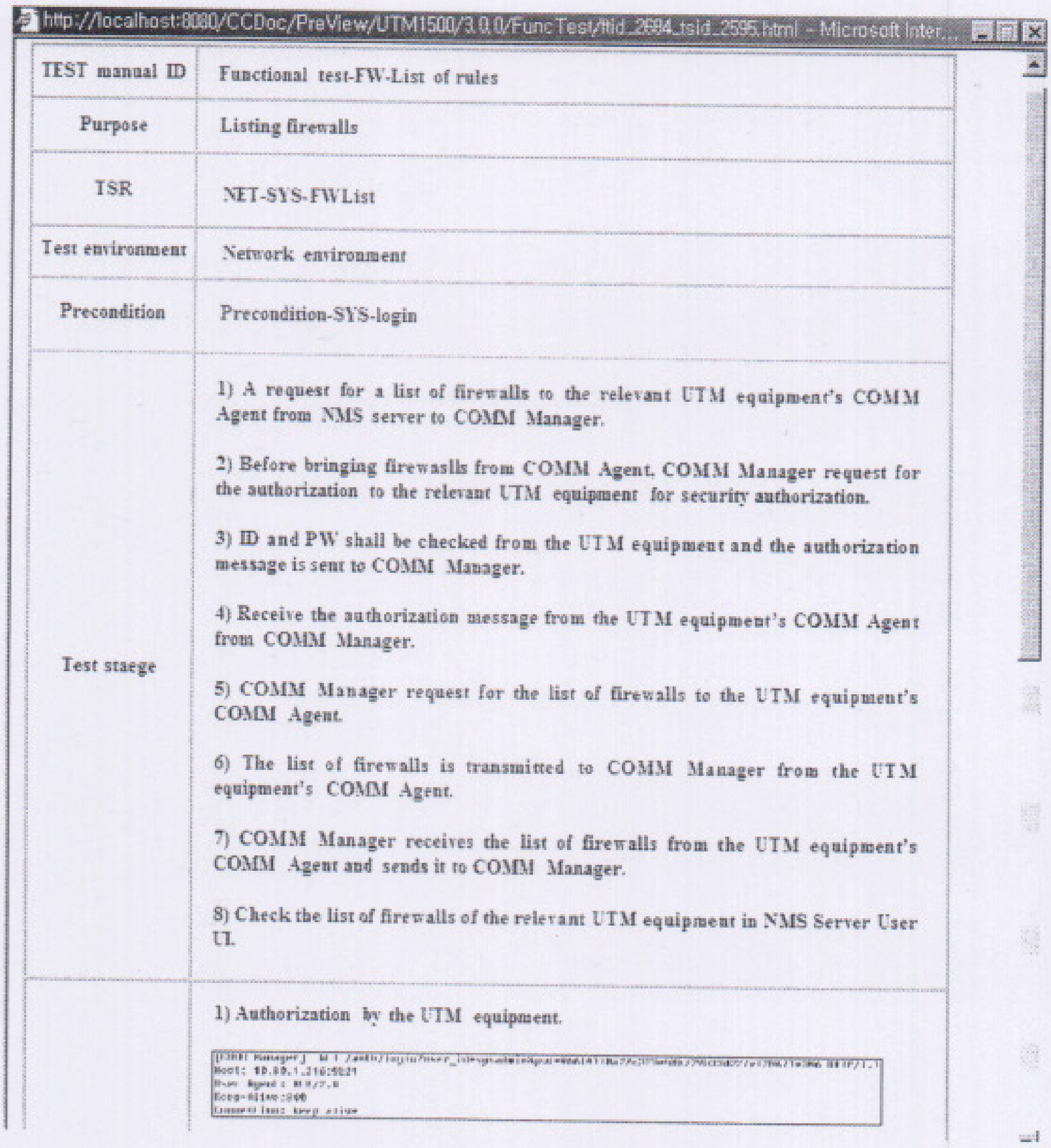
```
[COMM Manager] Connect to Server : 10.80.1.216  Port: 9221
[COMM Manager] Client IP : 10.80.1.61
----------------------------------------------------------
Test Module name = UnDefined-Debug-Module
Contents ********************
RecvMsg =
<utm_rule>
<idxlist>
<idxnum>2</idxnum>
<idxnum>3</idxnum>
</idxlist>
<filter>
<id>1</id>
<name>a111111111111</name>
<idxnum>2</idxnum>
<indev>any</indev>
```

8) Testing the confirmation of the relevant UTM equipment's firewall rule list in NMS Server user's UI



4.2 Confirmation of the Printing of Test Results

The results of tests of each stage are fed into the test management system. Pressing Preview or Print ensures printing based on the following form in Figure 7 through the test result management module.



http://localhost:8080/CCDoc/PreView/UTM1500/3.0.0/FuncTest/ttid_2694_tsid_2595.html - Microsoft Inter...

TEST manual ID	Functional test-FW-List of rules
Purpose	Listing firewalls
TSR	NET-SYS-FWList
Test environment	Network environment
Precondition	Precondition-SYS-login
Test staege	1) A request for a list of firewalls to the relevant UTM equipment's COMM Agent from NMS server to COMM Manager. 2) Before bringing firewaslls from COMM Agent, COMM Manager request for the authorization to the relevant UTM equipment for security authorization. 3) ID and PW shall be checked from the UTM equipment and the authorization message is sent to COMM Manager. 4) Receive the authorization message from the UTM equipment's COMM Agent from COMM Manager. 5) COMM Manager request for the list of firewalls to the UTM equipment's COMM Agent. 6) The list of firewalls is transmitted to COMM Manager from the UTM equipment's COMM Agent. 7) COMM Manager receives the list of firewalls from the UTM equipment's COMM Agent and sends it to COMM Manager. 8) Check the list of firewalls of the relevant UTM equipment in NMS Server User UI.
	1) Authorization by the UTM equipment.

```
[COMM Manager] [illegible]
Host: 10.10.1.216:9021
User Agent: [illegible]
Keep-Alive:[illegible]
Connection: keep alive
```

Fig. 7. Test result form

5 Conclusion

The test to verify the efficient operation of NMS was manually conducted based on a documented checklist. The test results were managed in Excel or Word, requiring the analysis or management of test results. Many testers tested each assigned part and collected all the results, resulting in problems with integrated management. This produced requirements for the test caser version management and reuse.

This paper designed and materialized the test management system to overcome such problems with NMS. The integration became more convenient as many testers' test results were centrally managed via the test management system. As the test results were documented and kept after computation, the safe storage was ensured. The test management system enabled separate saving of test results by version while the test case reuse are helped with efficient testing. Also, the interrelatedness of test cases was found while the automated report printing for test results became possible.

While there is a system that manages erroneous operation (bugs, errors, or faults) by using a tool like Bugzilla in the existing open sources and Bugzilla tool can be conveniently used as a means for communication between developers and testers, the suggested test management system not only provides overall test management of the system but also organically derives the result documents according to circumstances.

The test management tool and the test automation tool can be integrated to derive a way to support more efficient testing as a future study.

Acknowledgments. This research was supported by 2010 Hongik University Research Fund and National Research Foundation of Korea(NRF) through the Human Resource Training Project for Regional Innovation.

References

1. Nam Su, K., Hye Kyoung, R., Jai Ho, C., Gi Moo, C.: NMS Trends and The Case of Application of NMS for Effective Network Management. In: KICS 2003, pp. 11–34 (2003)
2. YoungJin, P., YoungMin, K., JaeWon, P., ChiYoung, L., NamYong, L.: A Conceptual Quality Evaluation Model of NMS Software Systems. In: Proceedings of The 32th KIISE Fall Conference, vol. 32(2), pp. 391–393 (2005)
3. Seong-ho, K.: Design and Implementation of NMS Platform using Multithread. KNOM Reveiw 4(1), 39–47 (2001)
4. Seokhwan, J., Jeongdong, K., Doo-Kwon, B.: A Flexible Unit Testing Tool for Test Driven Development. Journal of KIISE: Computing Practices and Letters 15(2), 140–144 (2009)
5. Joong Hee, M., Seong Hee, J., Sung Hoon, K., Yong Rae, K.: The Experimental Comparison of Fault Detection Efficiency of Black Box Testing Methods. In: Korea Computer Congress 2007, vol. 34(1)(B), pp. 41–46 (2007)
6. Eunjung, C., Byoungju, C.: Test Process Execution Tool: Test PET. Journal of KIISE: Computing Practices and Letters 10(2), 125–133 (2004)
7. Myoung-Wan K., Kim, R.Y.C.: A Study on Modeling NMS for the Effective Management among the Security VPNs. In: The IWIT 2009 Fall Green IT Conference, vol. 7(2), pp. 196–200 (2009)