Design Specification for Network Fault Diagnosis System

Lu Cao*, Baosheng Wang
National University of Defense Technology, Changsha, China
*Corresponding author’s e-mail: tianzhiyi3633@mail.nudt.edu.cn

Abstract. As the network becomes more and more complex and heterogeneous, the problem of network management becomes more and more prominent. The research and application of network management is of great significance for ensuring the normal operation of network and improving the reliability and availability of network system. The main problems of network management are network configuration, failure, performance, security, planning and scaling, etc. On the basis of in-depth research and discussion on network management and fault management, this paper designs the system requirements of network fault diagnosis system.

1. System overview

1.1. System functions
TSC (Trouble Shooting Control) and TSA (Trouble Shooting Agent) adopt distributed architecture. TSC runs on the server side and provides CLI interface for users to use, including link monitoring, network topology collection, network diagnosis, query display and other functions. TSA runs on the device as a bridge between TSC and equipment CLI to process and collect relevant information.[1-3]

1.1.1 Link monitoring function
In the network under the condition of normal can be added from TSC to terminal or link to monitor equipment as the source and destination address, in the fore and aft in configuration between two routers equipment simulation message source and destination addresses for contract, when the link is interrupted, packet loss when the fault condition, will immediately start the diagnosis function, a link in the network fault for induction and deduction, sums up the root cause.[4-8]

1.1.2 Whole network topology collection function
When the network topology is relatively complete and normal, it can be collected from the TSC triggered network topology, including the neighbor information and management port information of each device.

1.1.3 Diagnostic function of the whole network
According to the collected topology information of the whole network, traverse the equipment of the whole network, check the port authentication status, port link status, line card status, etc., and record the faults.

1.1.4 Query display function
1) Query link monitoring status/details.
2) List of link monitoring failure reasons/detailed reasons.
3) Diagnostic task log.
4) Network topology information.

2. Overall design

2.1. Overall architecture design
TSC and TSA adopt a distributed architecture. As a process module, TSA runs all network devices, including switches, routers and gateway devices. TSA is responsible for establishing persistent Telnet connection with device CLI, interacting with device CLI and returning data to TSC according to commands issued by TSC. TSC runs on the server as a process module that handles link monitoring, topology collection, and troubleshooting functions.
1) TSC connects the TSA program through TCP short link according to the address of the device business port management port, and sends relevant query information and configuration information through this connection;
2) After the TSA, the IMI process of the device is started. When the process is started, the Telnet long connection is established with the CLI of the device, and the general query interface and information return interface are provided to TSC. TSA does not deal with specific diagnostic logic, but only serves as a bridge between TSC and the CLI of the device. TSA will issue the package on a regular basis based on the package information issued by TSC.

2.2. Program structure
The programming of TSC control terminal and TSA agent terminal adopts the idea of module layering, which is divided into three layers: scheduling layer, control layer and basic service layer, as shown in Figure 1.
1) Scheduling layer: as the general entry of the program, corresponding control layer modules are respectively scheduled according to external input, which also includes timer timeout triggering, event triggering and socket message monitoring activation scheduling, etc.
2) Control layer: according to the message distribution scheduling of the scheduling layer, it is implemented in the way of business-oriented logical processing as the core.
3) Basic service layer: this layer is not targeted at specific business logic, but only responsible for providing the most basic general functional unit function interface for the use of the control layer module, such as message transceiver interface, timer setting, debug printing, database connection query interface and so on.

2.2.1 Overall structure of TSC
Figure 1. TSC Overall structure chart

1) CLI: Command line management module, which is responsible for responding to user input and output echo information.
2) Thread: pseudo-thread management module, responsible for timer, event, socket activation scheduling.
3) LM: link monitoring module, which is responsible for responding to the link configuration of CLI, interacting with the TSA module of the device through the module of the basic service layer, obtaining the forwarding path, regularly sending and detecting the situation of the message, and diagnosing the cause of path abnormality in time.
4) COR-DIAG: diagnostic processing module, which is responsible for the whole network topology collection, the whole network topology diagnosis, forwarding path diagnosis, fault scenario diagnosis and analysis and other functions.
5) DB: database service module, which is responsible for interacting with MySQL data of authentication server.
6) Debug: Debug information module, responsible for debugging information printing and logging.
7) IPC: process/thread communication service module, which is responsible for the communication interaction processing between the main thread CLI module and the sub-thread control layer module.
8) Task: Task processing service module, which is responsible for recording the information of each diagnostic and monitoring TASK to a file.
9) MSG: Message processing service module, which is responsible for establishing interactive communication with TSA through TCP connection.
10) DIAG: Diagnostic service module, which is responsible for providing access to device data information interface for control layer module to query, use and analyze, such as device interface information, device routing table, neighbor table, forwarding path, etc.

2.2.2 Overall structure of TSA
As shown in figure 2.
1) SCH-MSG: Message scheduling module, which is responsible for receiving messages sent from TSC and parsing and scheduling corresponding handlers.
2)Thread: pseudo-thread management module, responsible for timer, event, socket activation scheduling.
3) COR-DIAG: according to the messages sent by TSC, the corresponding callback processing functions are registered respectively to process the business logic, mainly including triggering packet issuing, device query, obtaining forwarding statistical count and other functions.
4)Debug: Debug information module, responsible for debugging information printing and logging.
5) MSG: Message processing service module, which is responsible for TCP connection with TSC to establish interactive communication, encapsulate message header, and provide sending interface for control layer module to call.
6)LM: Link monitoring service module, which is responsible for sending a detection packet according to the package information issued from TSC upper layer.
7)Telnet: Telnet client module, which is responsible for establishing Telnet long connection locally with device CLI and providing query interface to control layer module to directly query device information.

![Figure 2. TSA Overall structure chart](image)

2.3. Design of main basic services

2.3.1 Communication framework design between TSC and TSA
Considering the 1-to-many relationship between TSC and TSA, it is not suitable for TSC and TSA to use long connection to exchange data, as too many connections will lead to the exhaustion of system socket handle resources, so TSC and TSA use TCP short link to exchange data, namely the build-on-search mode, and destroy the connection immediately after checking, as shown in Figure 3.
1) After the TCP connection is established, TSC sends the connection request message to TSA again.
2) After TSA responds, TSC sends diagnostic message to obtain data.
3) The message scheduling module of TSA analyzes and distributes it to the callback function of the control layer module.
4) After logical processing, the TSA control layer module encapsulates data and sends it.
5) TSC receives the data message and sends it back to TSC’s control layer module for processing and processing.
6) After the TSA control layer module finishes processing, the message scheduling module of TSA sends a message that the data has been sent to TSC.
7) After TSC receives the data and sends the message, close the TCP connection.

![Figure 3. Message formats between TSC and TSA](image)

### 2.3.2 Design of TSA Key Basic Services

#### 2.3.2.1 Communication between TSA and device CLI

When the TSA is started, it will establish a Telnet long connection with the CLI of the device and automatically enter the Enable view. All subsequent operations will be performed by default in the Enable view. Since some devices will be configured to automatically disconnect from Telnet timeout, after the connection is established, the timer will be started to send ‘\r’ characters to the device CLI regularly to ensure that the device CLI is always connected to avoid repeated login and Telnet negotiation.

Telnet module provides command sending interface and result acquisition interface for upper control module, which must be used together:
- `tsa_bas_telnet_send_cmd`: show commands can be sent directly, for example "show interface summary".
- `tsa_bas_telnet_gets`: By line read device echo results, must be called many times until the function returns Error.

#### 2.3.2.2 TSA detection packet generation and statistics

Probe packet generation and statistics are divided into three steps as shown in the figure above: traffic statistics configuration, triggering packet sending and traffic statistics query. PTS library is used to communicate between TSA and NPAS.

Traffic statistics configuration: the TSC specifies the source address, destination address and the configured line card number. The target line card is specified to be sent to the corresponding NPAS through the CPTS library, and the NPAS stores this information. Therefore, this process relies on the forwarding module, and only the router device supports statistics at present.
Trigger message sending: TSC specifies the source address, destination address, next-hop interface MAC, outgoing interface, and the number of messages sent. According to the information sent by TSC, the LM module of TSA constructs the ID message of the four-layer protocol number 153 (FFWD identifies it as the detection message of the system by this protocol number) and sends it through RAW Socket. After it is received by local FFWD, it will forward it normally according to the destination address. When FFWD forwards and drops packets, it recognizes that the four-layer packets with protocol number 153 have been configured to conduct traffic statistics from the NPAS. If so, it will increase the statistical count in the NPAS and record the reason for packet loss. Traffic statistics query: TSC specifies the source address, destination address and query line card number; NPAS returns the forwarding statistics of the matched messages, the statistics of packet loss, and the reason for packet loss.

2.3.3 TSC key infrastructure service design

2.3.3.1 TSC start

TSC has two scheduling threads, in which the main thread is used for program initialization and response to user CLI interface operation, and the diagnostic thread is used for background link monitoring, real-time diagnosis and other functions. Each module initialization includes the following contents:

1) debug module initialization: mainly including log file handle initialization.
2) IPC module initialization: Unix socket is established for communication between the main thread and the diagnostic thread.
3) DB module initialization: connect the MySQL data of the authentication server.
4) Task module initialization: create the task report directory according to the current time, initialize the Task ID resource pool.
5) Initialization of DIAG module: create device information binary tree and recover device information from file.
6) LM module initialization: create link monitoring binary tree, recover link monitoring node from file, and register IPC communication callback function.
7) Initialization of COR-DIAG module: register IPC communication callback function.

2.3.3.2 TSC thread communication

In principle, the main thread control layer thread control module and diagnosis module belong to the same module, the main thread for cli user query shows that in order to ensure that the write protect of data between threads, the user of the cli configuration must be transferred to diagnose thread through the ipc module for processing, using Unix socket between the two threads to communicate, in the diagnosis of thread through thread scheduling algorithm for scheduling, but in fact for serial diagnostic tasks.

IPC module is packaged in accordance with TLV structure, and the upper module only needs to provide type and value parts, of which type includes:

```c
TSC_IPC_TLVTYPE_LM = 0, // Use of link monitoring modules
TSC_IPC_TLVTYPE_COLLECT_TOPO, // Network topology collection
TSC_IPC_TLVTYPE_AUTO_SYSCHECK, // Network topology diagnosis
```

2.4. Design of business function process
2.4.1 Link monitoring function
Communication monitoring, the function is used to link its sustainable detects link communication, send special detection message forward along the path to the timing, the router packet receiver statistics collection, in the end of the path if any packet loss or link, will start the diagnosis function, the forwarding path into/out of each device interface for inspection analysis, the comprehensive fault phenomenon, which is deduced a device or where there is a problem between two devices, such as loose cables to port the link between the two jump down, line card failure, etc.

2.4.1.1 Link monitoring state machine
(1) Collecting state: After the monitoring node is added, the collecting and forwarding path is started, and the device is set along the forwarding path to carry out statistical counting on the detection messages of the source and destination address of this monitoring.
(2) RUNNING state: if the link is detected regularly and no packet loss occurs in the forwarding path, this state will be maintained.
(3) WAIT_CHECK: If a link fault is detected, add a detection event to the scheduling queue and wait for detection. In principle, it will be dispatched immediately.
(4) Checking: In the checking state, all devices along the forward path are checked for interface and traffic statistics/causes of packet loss.
(5) WAIT_RESTART: After the test is completed, start the timer and wait for the link probe to be triggered again.
(6) Restarting: After the timer timeout, reset the statistics of detection message along the forwarding path and start sending detection message again.
(7) END: The end of link monitoring processing. For the sake of displaying the core part, this state is not indicated in the state machine diagram. This state can be migrated from any state when an unexpected error occurs during program running or the user configuration deletes link monitoring.

2.4.1.2 Forwarding path acquisition process
The basic logic of the process is to start from the source device and search ID route and loc route successively from the device to search the next hop until it finds the direct connection port with the destination address of the device. For the multi-path scenario of loc route, the routing algorithm in FFWD is transplanted to TSC to ensure that the routing is consistent with the actual forwarding path.

2.4.1.3 Main process of forwarding path diagnosis
For all devices on the forwarding path, the diagnostic check of the outgoing/incoming interface will be performed unconditionally. For router devices, additional queries will be made to detect the statistical count of sending and receiving/discard ing packets, which is used to analyze which segment of link has packet loss. The diagnostic process only supports the diagnosis of link connectivity and the statistics of which two hops have packet loss.

2.4.1.4 Interface diagnosis process
The interface diagnosis is based on the EAP state. If the EAP state is normal, the interface is considered to be normal. If the EAP state is normal, the interface state and line card state shall be
checked step by step, and the corresponding abnormal state information shall be collected, and the unified comprehensive analysis shall be conducted at last.

2.4.1.5 Failure reason derivation process

| priorit y | Loss of bags | characterization | Derogating Process |
|-----------|--------------|-----------------|--------------------|
| 1         | TSC_DIAG_CAUSE_LC_DRO P | Line card packet loss | Bag loss has been recorded, but there is no ffwd reason for packet loss |
| 2         | TSC_DIAG_CAUSE_FWD_DROP | Forwarding Dropouts | Bag loss has been recorded, there are ffwd reasons for packet loss |
| 3         | TSC_DIAG_CAUSE_EAP_PROTO_ERROR | Eap protocol error | Normal Growth of Receiving and Receiving Packages |
| 4         | TSC_DIAG_CAUSE_PATH_CHANGE | Forward path changes | New forwarding path changed when detected |
| 5         | TSC_DIAG_CAUSE_NPE_TX_ERROR | NPE send fault | adjacent interface, the last hop has no transmission received, this jump has no transmission received, then the last hop interface transmission direction failure |
| 6         | TSC_DIAG_CAUSE_NPE_RX_ERROR | NPE Receiving Fault | If there is a transmission or reception in the previous hop, there is a fault in the receiving direction of the previous hop interface |
| 7         | TSC_DIAG_CAUSE_NPE_ERROR | NPE fault | If there is no transmission or reception in the previous hop, there is a fault in the direction of transmission and reception in the previous hop |
The cause of the problem as shown in Table 1, already listed according to the diagnosis of state information collected during the process of derivation process, considering the reasons for the analysis of the derivation process of the above list may appear at the same time (such as line card error can lead to NPE error, eap protocol error, etc.), so, according to the seriousness of the fault and causality, before and after the prioritization, the greater the number the higher priority, will, in turn, covering the low priority in the process of analysis the cause of the problem, and finally presents the most serious mistakes.

3. Conclusion
This document is mainly to solve the network fault diagnosis system to achieve the system requirements of the technical solutions, mainly for technical solutions, design ideas, functional design and key technologies for the design and description.

Reference
[1] Pang, Y., Jia, L., Zhang, X., et al. (2020) Design and implementation of automatic fault diagnosis system for wind turbine[J]. Computers & Electrical Engineering, 87:106754.
[2] Shahnazari, H., Mhaskar, P., House, J. M., et al. (2019) Modeling and fault diagnosis design for HVAC systems using recurrent neural networks[J]. Computers & Chemical Engineering, 126(JUL.12):189-203.
[3] Loureiro, G. B., Ferreira, J. C. E., Messerschmidt, P. Z. . (2020) Design structure network (DSN): a method to make explicit the product design specification process for mass customization[J]. Research in Engineering Design, 31(2):197-220.
[4] Zhou, Z., Zhong, M., Wang, Y. (2019) Fault Diagnosis Observer and Fault-Tolerant Control Design for Unmanned Surface Vehicles in Network Environments[J]. IEEE Access, 7:173694-173702.

[5] Lee, C., Suh, J. H., Yu, M. H., et al. (2020) Design considerations of local controller for ITER coil power supply with high availability and reliability[J]. Fusion Engineering and Design, 159:111763.

[6] Wu, G., Chen, C., Yang, N., et al. (2019) Design of Differential GPS System Based on BP Neural Network Error Correction for Precision Agriculture[C]// Chinese Intelligent Automation Conference. Springer, Singapore.

[7] Xiong, S., Shi, T. (2020) Deep residual network for enhanced fault diagnosis of rotating machinery[J]. Journal of Physics: Conference Series, 1707(1):012010 (6pp).

[8] Alexandre, dos, Santos, et al. (2019) An Approach to Address Safety as Non-Functional Requirements in Distributed Vehicular Control Systems[J]. Journal of Control, Automation and Electrical Systems, 30(5):700–715.