
Juphoon Protocol Framework

Simple Traversal of UDP through NATs

Published: Nov 2006

For more information on Juphoon Protocol Framework, see <http://www.juphoon.com>

Juphoon STUN Client Function Definition

Juphoon System Software Corporation.

<http://www.juphoon.com>

Tel: +86-574-87287820

Fax: +86-574-87304379

Copyright © 2007, Juphoon System Software Corporation.

All rights reserved.

CONTENTS

1. INTRODUCTION.....	4
1.1 STUN.....	4
1.2 AUDIENCE	4
1.3 SCOPE.....	4
1.4 ABBREVIATIONS	4
2. SYSTEM ENVIRONMENT.....	4
2.1 BASIC DATA TYPES	5
3. DEFINITIONS	5
3.1 STUN CLIENT	6
3.2 STUN SERVER	6
3.3 SHARED SECRET REQUESTS AND RESPONSES	6
3.4 STUN BINDING REQUESTS AND RESPONSES	6
4. OVERVIEW OF OPERATION.....	6
4.1 DISCOVERY OF PUBLIC IP ADDRESS AND PORT MAPPINGS	7
4.2 DETECT THE TYPE OF THE NAT A STUN CLIENT IS BEHIND.....	8
5. USER INTERFACES	8
5.1 STRUCTURES	8
5.1.1 <i>ST_ZOS_INET_ADDR</i>	8
5.1.2 <i>PFN_STUNCALLBACK</i>	9
5.2 CONFIG INTERFACES.....	9
5.2.1 <i>Stun_CfgGetTaskPriority</i>	9
5.2.2 <i>Stun_CfgGetTaskStackSize</i>	10
5.2.3 <i>Stun_CfgGetTaskQueueSize</i>	10
5.2.4 <i>Stun_CfgGetTaskTimerNum</i>	10
5.2.5 <i>Stun_CfgGetLogLevel</i>	11
5.2.6 <i>Stun_CfgGetLocalIpv4</i>	11
5.2.7 <i>Stun_CfgGetServName</i>	12
5.2.8 <i>Stun_CfgGetServIpv4</i>	12
5.2.9 <i>Stun_CfgGetServUPort</i>	12
5.2.10 <i>Stun_CfgGetQryNum</i>	13
5.2.11 <i>Stun_CfgSetTaskPriority</i>	13
5.2.12 <i>Stun_CfgSetTaskStackSize</i>	14
5.2.13 <i>Stun_CfgSetTaskQueueSize</i>	14
5.2.14 <i>Stun_CfgSetTaskTimerNum</i>	15
5.2.15 <i>Stun_CfgSetLogLevel</i>	15
5.2.16 <i>Stun_CfgSetLocalIpv4</i>	15
5.2.17 <i>Stun_CfgSetServName</i>	16
5.2.18 <i>Stun_CfgSetServIpv4</i>	16
5.2.19 <i>Stun_CfgSetServUPort</i>	17

5.2.20	<i>Stun_CfgSetQryNum</i>	17
5.3	TASK INTERFACES	18
5.3.1	<i>Stun_Start</i>	18
5.3.2	<i>Stun_Stop</i>	18
5.4	USEFUL INTERFACES	18
5.4.1	<i>Stun_Lookup</i>	19
5.4.2	<i>Stun_LookupU</i>	20
5.4.3	<i>Stun_LookupX</i>	21
5.4.4	<i>Stun_LookupUX</i>	22

List of Tables

TABLE 2-1: BASIC DATA TYPES	5
-----------------------------------	---

List of Figures

FIGURE 2-1 STUN CLIENT	5
FIGURE 4-1 STUN CONFIGURATION	6
FIGURE 4-2 USE STUN_LOOKUP TO GET THE PUBLIC IP AND PORT.....	7
FIGURE 4-3 USE STUN_LOOKUPX TO GET THE PUBLIC IP AND PORT.....	8

1. Introduction

1.1 STUN

Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs) (STUN) is a lightweight protocol that allows applications to discover the presence and types of NATs and firewalls between them and the public Internet. It also provides the ability for applications to determine the public Internet Protocol (IP) addresses allocated to them by the NAT. STUN works with many existing NATs, and does not require any special behavior from them. As a result, it allows a wide variety of applications to work through existing NAT infrastructure.

1.2 Audience

The readers of this document are assumed to have a working knowledge of Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs) (STUN).

1.3 Scope

This document provides the definitions of STUN interfaces for users to learn the addresses bindings allocated by the NAT. It does not describe how STUN is designed and realized.

1.4 Abbreviations

The following abbreviations are used in this document:

Abbreviation	Description
IP	Internet Protocol
NAT	Network Address Translator
STUN	Simple Traversal of User Datagram Protocol through Network Address Translators
UDP	User Datagram Protocol
ZOS	Zero Operating System

2. System Environment

This section describes the environment in which STUN is designed to operate. Figure 2-1 illustrates the framework.

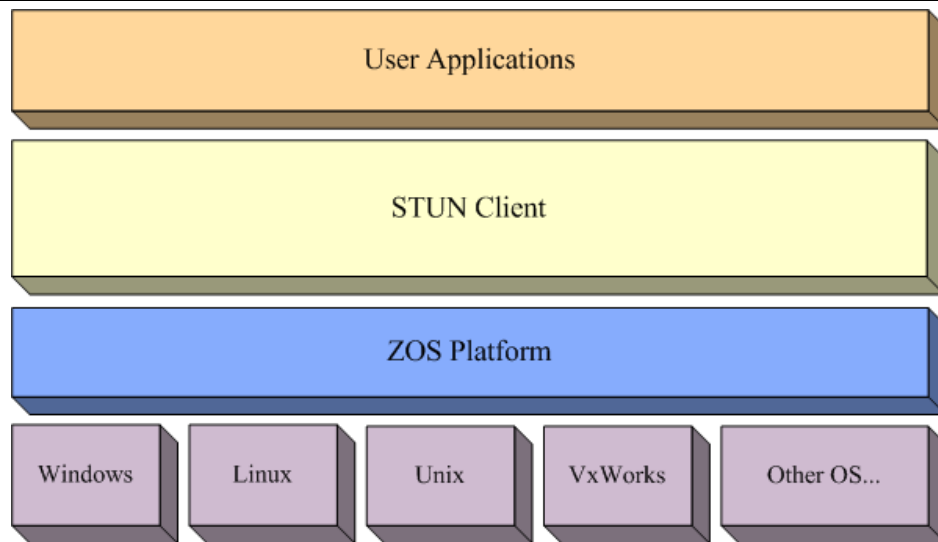


Figure 2-1 STUN Client

2.1 Basic Data Types

There are some basic data types provided by ZOS platform. Table 2-1 lists these types used by the STUN Client.

Name	Type
ZDOUBLE	double
ZFLOAT	float
ZLONG	long
ZINT	int
ZSHORT	short
ZCHAR	char
ZULONG	unsigned long
ZUINT	unsigned int
ZSIZE_T	unsigned int
ZUSHORT	unsigned short
ZUCHAR	unsigned char
ZBOOL	int
ZVOID	void

Table 2-1: Basic Data Types

3. Definitions

STUN is a simple client-server protocol. A client sends a request, and a server returns a response.

3.1 STUN Client

A STUN client (also just referred to as a client) is an entity that generates STUN requests. A STUN client can execute on an end system, such as a user’s PC, or can run in a network element, such as a conferencing server.

3.2 STUN Server

A STUN server (also just referred to as a server) is an entity that receives STUN requests, and sends STUN responses. STUN servers are generally attached to the public Internet.

3.3 Shared Secret Requests and Responses

Shared Secret Requests are sent over TLS over TCP.

Shared Secret Requests ask the server to return a temporary username and pass word. This username and password are used in a subsequent Binding Request and Binding Response, for the purposes of authentication and message integrity.

Shared Secret Requests and Responses are not within the scope of this document.

3.4 STUN Binding Requests and Responses

Binding requests are used to determine the bindings allocated by NATs.

The client sends a Binding request to the server, over UDP. The server examines the source IP address and port of the request, and copies them into a response that is sent back to the client.

4. Overview of Operation

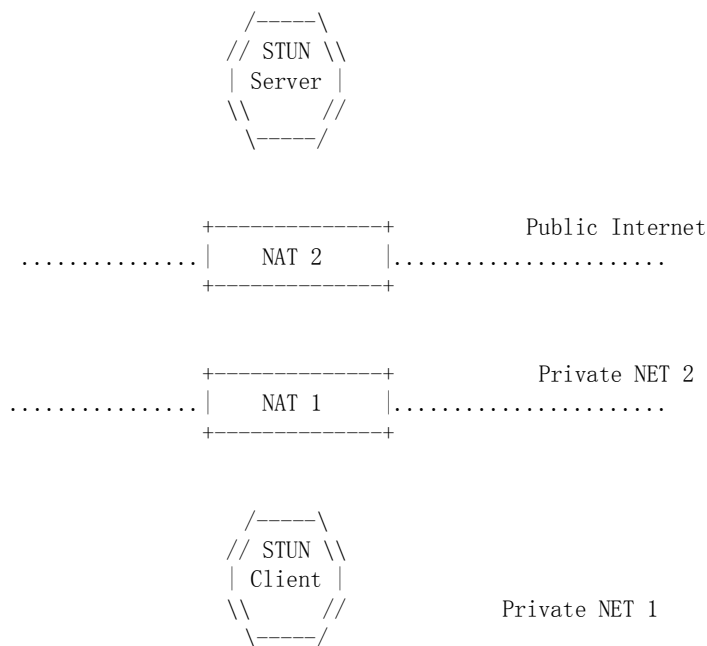


Figure 4-1 STUN Configuration

A typical STUN configuration is shown in the figure above. A STUN client connects to private network 2 through NAT1. Private network 2 connects to the public internet through NAT2. The STUN server resides on the public Internet. And it should be noted that the configuration in Figure4-1 is not the only permissible configuration.

The STUN client is typically embedded in an application which needs to obtain a public IP address and port that can be used to receive data. When the application starts, the STUN client within the application sends a STUN Shared Secret Request to its server, obtains a username and password, and then sends it a Binding Request.

4.1 Discovery of Public IP Address and Port Mappings

The STUN Binding Request is used to discover the presence of a NAT, and to discover the public IP address and port mappings generated by the NAT. When a Binding Request arrives at the STUN server, it may have passed through one or more NATs between the STUN client and the STUN server. As a result, the source address of the request received by the STUN server will be the mapped address created by the NAT closest to the server. So in Figure4-1, the source address of the Binding Request sent by the STUN client is the mapped address created by NAT2.

The STUN server copies the source IP address and port into a STUN Binding Response, and sends it back to the source IP address and port of the STUN request.

When the STUN client receives the STUN Binding Response, it compares the IP address and port in the packet with the local IP address and port it bound to when the request was sent. If these do not match, the STUN client is behind one or more NATs.

There are two interfaces provided by Juphoon STUN Client to upper users for discovering public IP address and port mappings. Please refer to [Stun_Lookup](#) and [Stun_LookupX](#) for more information.

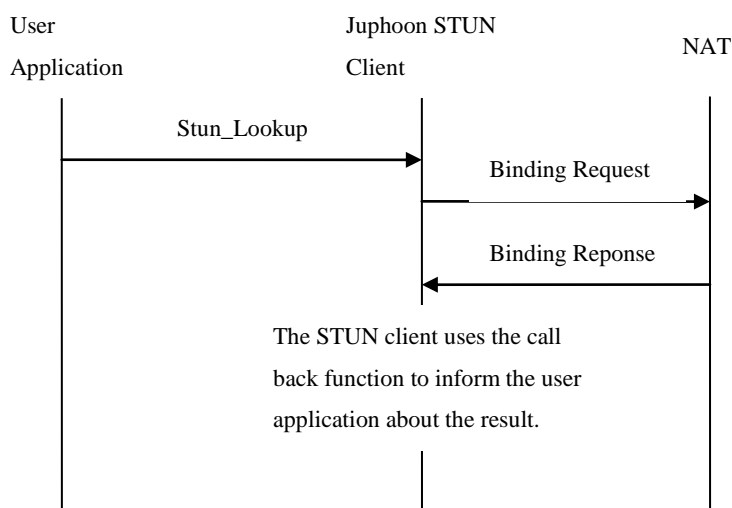


Figure 4-2 Use [Stun_Lookup](#) to get the public IP and port

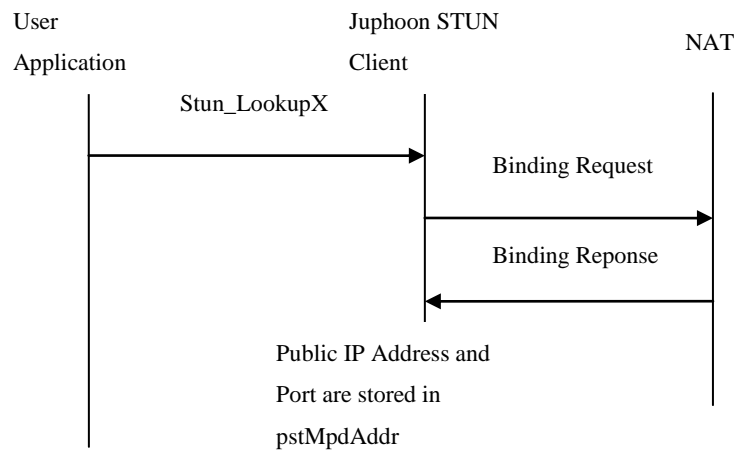


Figure 4-3 Use [Stun_LookupX](#) to get the public IP and port

4.2 Detect the Type of the NAT a STUN client is behind

To determine the type of the NAT a STUN client is behind, STUN allows the client to ask the server to send special Binding responses that can be used by the client to learn which type the NAT is.

Details on how to detect the type of the NAT a STUN client is behind are not within the scope of this document.

5. User Interfaces

5.1 Structures

5.1.1 ST_ZOS_INET_ADDR

It is ZOS internet address structure used to hold the public IP address and port which are copied from a Binding Response received by the STUN client. *u.dwlIp* is used to hold the public IP address and *wPort* is used to hold the mapped port.

```

typedef struct tagZOS_INET_ADDR
{
    ZUCHAR ucType;                /* ZINET_IPV4... */
    ZUCHAR aucSpare[1];          /* for 32 bit alignment */
    ZUSHORT wPort;                /* not order[host or n/w] dependent */
    union
    {
        ZULONG dwIp;             /* not order[host or n/w] dependent */
        ZUCHAR aucIp[ZINET_IPV4_ADDR_SIZE]; /* ipv4 address */
        ZUCHAR aucIpv6[ZINET_IPV6_ADDR_SIZE]; /* ipv6 address */
    } u;
} ST_ZOS_INET_ADDR;

```

5.1.2 PFN_STUNCALLBACK

The prototype of the callback function used by Juphoon STUN Client when it gets the IP address and port mappings or the attempt fails.

```

/* stun query result callback */
typedef ZVOID (*PFN_STUNCALLBACK) (ZULONG dwPort,
    ST\_ZOS\_INET\_ADDR *pstMapAddr);

```

5.2 Config Interfaces

These interfaces are included in `stun_cfg.h`.

5.2.1 Stun_CfgGetTaskPriority

Gets the task priority. The default priority is `ZTASK_PRIORITY_NORMAL`.

```

ZINT Stun_CfgGetTaskPriority(ZINT *piPriority);

```

[Parameters]

Input parameters:

None.

Output parameters:

ZINT *piPriority
The task priority.

[Return value]

Returns ZOK.

5.2.2 Stun_CfgGetTaskStackSize

Gets the task stack size. The default size is 16K.

```
ZINT Stun_CfgGetTaskStackSize(ZULONG *pdwStackSize);
```

[Parameters]

Input parameters:

None.

Output parameters:

ZULONG *pdwStackSize
The task stack size.

[Return value]

Returns ZOK.

5.2.3 Stun_CfgGetTaskQueueSize

Gets the task queue size. The default size is 50.

```
ZINT Stun_CfgGetTaskQueueSize(ZULONG *pdwQueueSize);
```

[Parameters]

Input parameters:

None.

Output parameters:

ZULONG *pdwQueueSize
The task queue size.

[Return value]

Returns ZOK.

5.2.4 Stun_CfgGetTaskTimerNum

Gets the number of the task timers. The default number is 5.

```
ZINT Stun_CfgGetTaskTimerNum(ZULONG *pdwTimerNum);
```

[Parameters]

Input parameters:

None.

Output parameters:

ZULONG *pdwTimerNum
The number of timers.

[Return value]

Returns ZOK.

5.2.5 Stun_CfgGetLogLevel

Gets the log level.

```
ZINT Stun_CfgGetLogLevel (ZULONG *pdwLevel);
```

[Parameters]**Input parameters:**

None.

Output parameters:

ZULONG *pdwLevel
The log level.

[Return value]

Returns ZOK.

5.2.6 Stun_CfgGetLocalIpv4

Gets the local IPv4 address. The default address is "192.168.0.10"

```
ZINT Stun_CfgGetLocalIpv4 (ZULONG *pdwIpv4);
```

[Parameters]**Input parameters:**

None.

Output parameters:

ZULONG *pdwIpv4
The local IPv4 address string.

[Return value]

Returns ZOK.

5.2.7 Stun_CfgGetServName

Gets the STUN server name. The default name is “stun.fwdnet.net”.

```
ZINT Stun_CfgGetServName (ZCHAR **ppcName) ;
```

[Parameters]

Input parameters:

None.

Output parameters:

```
ZCHAR **ppcName  
The STUN server name.
```

[Return value]

Returns ZOK.

5.2.8 Stun_CfgGetServIpv4

Gets the STUN server IPv4 address. The default server IPv4 address is “192.168.0.250”.

```
ZINT Stun_CfgGetServIpv4 (ZULONG *pdwIpv4) ;
```

[Parameters]

Input parameters:

None.

Output parameters:

```
ZULONG *pdwIpv4  
The STUN server IPv4 address.
```

[Return value]

Returns ZOK.

5.2.9 Stun_CfgGetServUPort

Gets the server UDP port. The default port is 3478.

```
ZINT Stun_CfgGetServUPort (ZULONG *pdwPort);
```

[Parameters]

Input parameters:

None.

Output parameters:

ZULONG *pdwPort

The UDP port.

[Return value]

Returns ZOK.

5.2.10 Stun_CfgGetQryNum

Gets the number of the query blocks. The default number is

```
ZINT Stun_CfgGetQryNum (ZULONG *pdwNum);
```

[Parameters]

Input parameters:

None.

Output parameters:

ZULONG *pdwNum

The number of the query blocks.

[Return value]

Returns ZOK.

5.2.11 Stun_CfgSetTaskPriority

Sets the task priority.

```
ZINT Stun_CfgSetTaskPriority (ZINT iPriority);
```

[Parameters]

Input parameters:

ZINT iPriority

The task priority.

Output parameters:

None.

[Return value]

Returns ZOK.

5.2.12 Stun_CfgSetTaskStackSize

Sets the task stack size.

```
ZINT Stun_CfgSetTaskStackSize(ZULONG dwStackSize);
```

[Parameters]

Input parameters:

ZULONG dwStackSize
The task stack size.

Output parameters:

None.

[Return value]

Returns ZOK.

5.2.13 Stun_CfgSetTaskQueueSize

Sets the task queue size.

```
ZINT Stun_CfgSetTaskQueueSize(ZULONG dwQueueSize);
```

[Parameters]

Input parameters:

ZULONG dwQueueSize
The task queue size.

Output parameters:

None.

[Return value]

Returns ZOK.

5.2.14 Stun_CfgSetTaskTimerNum

Sets the number of task timers. The default number is 5.

```
ZINT Stun_CfgSetTaskTimerNum(ZULONG dwTimerNum);
```

[Parameters]

Input parameters:

```
ZULONG dwTimerNum  
The number of task timers.
```

Output parameters:

```
None.
```

[Return value]

```
Returns ZOK.
```

5.2.15 Stun_CfgSetLogLevel

Sets the log level.

```
ZINT Stun_CfgSetLogLevel(ZULONG dwLevel);
```

[Parameters]

Input parameters:

```
ZULONG dwLevel  
The log level.
```

Output parameters:

```
None.
```

[Return value]

```
Returns ZOK.
```

5.2.16 Stun_CfgSetLocalIpv4

Sets the local IPv4 address.

```
ZINT Stun_CfgSetLocalIpv4(ZULONG dwIpv4);
```

[Parameters]

Input parameters:

ZULONG dwIpv4
The local IPv4 address.

Output parameters:

None.

[Return value]

Returns ZOK.

5.2.17 Stun_CfgSetServName

Sets the name of the STUN server. It can be a string of its IP address or the domain name.

```
ZINT Stun_CfgSetServName(ZCHAR *pcName);
```

[Parameters]

Input parameters:

ZCHAR *pcName
The STUN server name.

Output parameters:

None.

[Return value]

Returns ZOK.

5.2.18 Stun_CfgSetServIpv4

Sets the IPv4 address of the STUN server.

```
ZINT Stun_CfgSetServIpv4(ZULONG dwIpv4);
```

[Parameters]

Input parameters:

ZULONG dwIpv4
The IPv4 address.

Output parameters:

None.

[Return value]

Returns ZOK.

5.2.19 Stun_CfgSetServUPort

Sets the UDP port of the STUN server.

```
ZINT Stun_CfgSetServUPort(ZULONG dwPort);
```

[Parameters]

Input parameters:

ZULONG dwPort

The UDP port.

Output parameters:

None.

[Return value]

Returns ZOK.

5.2.20 Stun_CfgSetQryNum

Sets the number of the query blocks.

```
ZINT Stun_CfgSetQryNum(ZULONG dwNum);
```

[Parameters]

Input parameters:

ZULONG dwNum

Number of the query blocks.

Output parameters:

None.

[Return value]

Returns ZOK.

5.3 Task Interfaces

5.3.1 Stun_Start

This function is used to start the STUN task.

```
ZINT Stun_Start();
```

[Parameters]

Input parameters:

None.

Output parameters:

None.

[Return value]

Returns ZOK on success, or ZFAILED on failure.

5.3.2 Stun_Stop

Stops the STUN task.

```
ZVOID Stun_Stop();
```

[Parameters]

Input parameters:

None.

Output parameters:

None.

[Return value]

None.

5.4 Useful Interfaces

Next details will be given on four user interfaces provided by Juphoon STUN Client to get the public IP address and port mappings. These interfaces are included in `stun_ui.h`.

[Stun Lookup](#)

[Stun LookupX](#)

5.4.1 Stun_Lookup

Like [Stun_LookupX](#), Stun_Lookup is also used to get the public IP address and port mappings. What is different from Stun_LookupX is that the user application process does not need to wait before Juphoon STUN Client gets the public IP address and port, or before the operation fails. The user application calls *Stun_Lookup*, and then resumes what it was doing. The user application can define a callback function to get the result --- the public IP address and port if Juphoon STUN Client has succeeded to get them or the information on failure. In this callback function, the user can define a series of operations that will be executed by Juphoon STUN Client after the Binding Response is returned or a timeout error occurs because no Binding Response is received by the client.

```
ZINT Stun_Lookup(ZULONG dwPort, ZULONG dwTryTimeLen, ZULONG dwTryCnt,  
                PFN\_STUNCALLBACK pfnCallback);
```

[Parameters]

Input parameters:

ZULONG dwPort

The port number, which should be designated by the user application. Its mapping will be returned by a STUN server in a Binding Response to the Binding request sent by Juphoon STUN Client on the order of the user application by calling *Stun_Lookup*.

ZULONG dwTryTimeLen

Indicates how long time the task should wait for a query try.

ZULONG dwTryCnt

Indicates how many times the task can try to query.

[PFN_STUNCALLBACK](#) pfnCallback

The callback function, used by the user application to get the result --- the public IP address and port if Juphoon STUN Client has succeeded to get them, or the information on failure.

Output parameters:

None.

[Return value]

Returns ZOK on success, or ZFAILED on failure.

[Example]

```

ZULONG dwPort;
ZULONG dwTryTimeLen;
ZULONG dwTryCnt;
ZINT iRet;
PFN\_STUNCALLBACK pfnCallback;

ZINT Stun_Callback (ZULONG dwPort, ST_ZOS_INET_ADDR *pstMapAddr)
{
    .....
    Return ZOK;
}
.....
pfnCallback = Stun_Callback;
iRet = Stun_Lookup(dwPort, dwTryTimeLen, dwTryCnt,pfnCallback);

```

5.4.2 Stun_LookupU

Like Stun_Lookup, this interface is also used to get public IP address. The difference is that the STUN message was sent through an opened UTAL channel.

```

ZINT Stun_LookupU(ZULONG dwTptId, ZULONG dwPort, ZULONG dwTryTimeLen,
                 ZULONG dwTryCnt, PFN\_STUNCALLBACK pfnCallback);

```

[Parameters]

Input parameters:

ZULONG dwTptId

The UTAL transport ID.

ZULONG dwPort

The port number, which should be designated by the user application. Its mapping will be returned by a STUN server in a Binding Response to the Binding request sent by Juphoon STUN Client on the order of the user application by calling *Stun_LookupX*.

ZULONG dwTryTimeLen

Indicates how long time the task should wait for a query try.

ZULONG dwTryCnt

Indicates how many times the task can try to query.

[PFN_STUNCALLBACK](#) pfnCallback

The callback function, used by the user application to get the result --- the public IP address and port if Juphoon STUN Client has succeeded to get them, or the information on failure.

Output parameters:

None.

[Return value]

Returns ZOK on success, or ZFAILED on failure.

[Example]

5.4.3 Stun_LookupX

A user application is able to ask Juphoon STUN Client to send a Binding Request to a STUN server in order to get the public IP address and port mappings by calling *Stun_LookupX*. Note that the user application process will wait until a Binding Response to the Binding Request sent by the client is received. The public address and port will be copied into the field pointed by *pstMpdAddr*.

```
ZINT Stun_LookupX(ZULONG dwPort, ZULONG dwTryTimeLen, ZULONG dwTryCnt,  
                 ST\_ZOS\_INET\_ADDR *pstMapAddr);
```

[Parameters]**Input parameters:**

ZULONG dwPort

The port number, which should be designated by the user application. Its mapping will be returned by a STUN server in a Binding Response to the Binding request sent by Juphoon STUN Client on the order of the user application by calling *Stun_LookupX*.

ZULONG dwTryTimeLen

Indicates how long time the task should wait for a query try.

ZULONG dwTryCnt

Indicates how many times the task can try to query.

Output parameters:

[ST_ZOS_INET_ADDR](#) *pstMapAddr

A pointer to the field which is used to hold the public IP address and port which are copied from the Binding Response received by Juphoon STUN Client.

[Return value]

Returns ZOK on success, or ZFAILED on failure.

[Example]

```

ZULONG dwPort;
ZINT iRet;
ZULONG dwTryTimeLen;
ZULONG dwTryCnt;
ST\_ZOS\_INET\_ADDR stMapAddr;
.....
iRet = Stun_LookupX(dwPort, dwTryTimeLen, dwTryCnt, &stMapAddr);

```

5.4.4 Stun_LookupUX

Like Stun_LookupX, this interface is also used to get public IP address. The difference is that the STUN message was sent through an opened UTAL channel.

```

ZINT Stun_LookupUX(ZULONG dwTptId, ZULONG dwPort, ZULONG dwTryTimeLen,
                  ZULONG dwTryCnt, ST\_ZOS\_INET\_ADDR *pstMapAddr);

```

[Parameters]

Input parameters:

ZULONG dwTptId
The UTAL transport ID.

ZULONG dwPort
The port number, which should be designated by the user application. Its mapping will be returned by a STUN server in a Binding Response to the Binding request sent by Juphoon STUN Client on the order of the user application by calling *Stun_LookupX*.

ZULONG dwTryTimeLen
Indicates how long time the task should wait for a query try.

ZULONG dwTryCnt
Indicates how many times the task can try to query.

Output parameters:

[ST_ZOS_INET_ADDR](#) *pstMapAddr
A pointer to the field which is used to hold the public IP address and port which are copied from the Binding Response received by Juphoon STUN Client.

[Return value]

Returns ZOK on success, or ZFAILED on failure.

[Example]