NTCP: NAT traversal TCP HOWTO

Luca Gaballo

September 13, 2005

Contents

1 Product name and version number

Name: NTCP (NAT traversal TCP)
version: 0.2 (beta version)

2 Company name

Company: France Telecom R&D
Department: MAPS/MMC
Project: SOLIPSIS
http://solipsis.netofpeers.net/wiki2/index.php/Main_Page

3 New and special in this release

- Implementation of various methods for NAT traversal
- STUNT protocol implementation (client and server)
- Remake of UDP Hole punching (Protocol, Puncher and Connection Broker)
- Integration in SOLIPSIS project
- other tests for TCP NAT traversal

4 Download source code and documentation

Anonymous SVN Access

The project's BerliOS Developer SVN repository can be checked out through anonymous (svnserver) SVN with the following instruction set. To see the list of the subdirectories available in the repository use the web-based SVN repository access with ViewCVS or WebSVN.

```
svn checkout svn://svn.berlios.de/ntcp/trunk
```

Developer SVN Access via SSH

Only project developers can access the SVN tree via this method. SSH2 must be installed on your client machine. Substitute developername with the proper value. Enter your site password when prompted.

```
svn checkout \
svn+ssh://developername@svn.berlios.de/svnroot/repos/ntcp/trunk
```

The subdirectories for documentation or source code only are respectively:

- /ntcp/trunk/doc
- /ntcp/trunk/p2pNetwork

5 Hardware and software requirements

5.1 General Requirements

- **Python**
  http://www.python.org/
- **Twisted** (at least 1.3)
  http://twistedmatrix.com/projects/core/

5.2 Special Unix Requirements

Connection Broker side:

- **UNIX:** the Connection broker require UNIX system for spoofing section
- **libpcap** (at least 0.9.1)
- **Pcapy**: python library for spoofing (modification of address, SYN, ACK, TTL, ...) (http://oss.coresecurity.com/projects/pcapy.html)
- **Impacket**: python library to sniff and listen for outgoing packets
  http://oss.coresecurity.com/projects

Peer side: if you want to implement spoofing method you need this package. Otherwise the NTCP operations will be reduced to the methods without spoofing. (PS: Spoffing methods require administrator privilege on the machine too!)

- **libpcap** (at least 0.9.1)
- **Pcapy**: python library for spoofing (modification of address, SYN, ACK, TTL, ...) (http://oss.coresecurity.com/projects/pcapy.html)
- **Impacket**: python library to sniff and listen for outgoing packets
  http://oss.coresecurity.com/projects

5.3 Special Windows Requirements

- **Python** (at least 2.3): some socket options are not available in the previous version
- **winpcap** (at least 3.0): the packet capture library for Windows
  http://www.winpcap.org
- **Impacket**: python library to sniff and listen for outgoing packets
  http://oss.coresecurity.com/projects

6 Installation instructions, getting started tips, and documentation

6.1 Installation

There is no installation procedure at now. So download the source code (see download section), put it to the desired folder, and make the PYTHONPATH environment variable to point it. Execute this command or put it in your '*profile*' file.

```
export PYTHONPATH=\$PYTHONPATH:/directory_to_ntcp/ntcp/
```

6.2 Modules' structure

The source code in the /ntcp/branches/beta/ntcp is organized like exposend

```
/ntcp/branches/beta/ntcp
 |
 |__ connection: some usefull class to wrap twisted structure
 |
 |__ punch: Hole Punching protocol implementation (Connection Broker and Puncher)
 |
 |__ stunt: STUNT porotocol implementation (client and server)
 |
 |__ test: tests for NTCP connectivity. Look here if you are loking for some example
 |
 |__ NatConnectivity.py: the most important file.
                         It gives you the acces to the NTCP functions
```

6.3 Getting started

To know how to use *ntcp* library in your code look at the examples in the ntcp/test/ directory; the file simulator can help you.

6.4 Start server and connection broker

For particular settings see the Configuration files section.

- STUNT server: start the python file in your branche *ntcp/stunt/StuntServer.py*
- Super Node Connection Broker: start the python file in your branche *ntcp/punch/SNConnectionBroker.py*

6.5 Configuration files

Before to start use the ntcp library, you have to configure the configuration file.

Endpoint side:

- *ntcp/stunt/StuntClient.py*

Listing 1: ntcp/stunt/StuntClient.py

```
DefaultServers = [
    ('p-maul.rd.francetelecom.fr', 3478),
    ('new-host-2.mmcbill2', 3478),
]
```

- *ntcp/p2pNetwork.conf*

Listing 2: ntcp/p2pNetwork.conf

```
# ConnectionBroker for peer configuration
[holePunch]
ConnectionBroker: p-maul.rd.francetelecom.fr:6060
```

STUNT Server side:
For STUNT server you need two different Ip addresses. These addresses can be on the same machine or on two different machine. On UNIX system, infact, you can configure two different Ip address on the same ethernet device.

- *ntcp/p2pNetwork.conf*

Listing 3: test.conf

```
[stunt]
# STUNT Server configuration
stuntIp: 192.168.1.201
stuntPort: 3478
# If the two server are on the same machine
# it needs of the second interface's name
otherStuntIp: 192.168.1.203
otherStuntPort: 3479
```

6.6 API Documentation

For the complete API documentation and a report on NAT traversal see the documentation in /ntcp/branches/beta/doc/ directory

7 Important known problems

The *ntcp* is still in an initial phase, it isn't stable and dynamically configurable.

8 Version history

Release 0.2: A real NTCP library. Tested and integrated in applications.

- Implementation of various methods for NAT traversal
- STUNT protocol implementation (client and server)
- Remake of UDP Hole punching (Protocol, Puncher and Connection Broker)
- Integration in SOLIPSIS project
- other tests for TCP NAT traversal

Release 0.1: First release of NTCP

- STUN implementation (client and server)
- UDP Hole punching (Puncher and Connection Broker)
- test for TCP NAT traversal

9 Contact information

Gaballo Luca
mail: gaballo@enst.fr

10 Legal information

...