Project no. IST-033576

XtreemOS

Integrated Project
BUILDING AND PROMOTING A LINUX-BASED OPERATING SYSTEM TO SUPPORT VIRTUAL ORGANIZATIONS FOR NEXT GENERATION GRIDS

NAT Traversal Overview

XtreemOS Technical Report # 3

Marko Novak (XLAB), Gregor Pipan (XLAB), Luis Pablo (TID), Guillaume Pierre (VUA), Diego Puppin (CNR), Brian Matthews (CCLRC)

Report Registration Date: January 18th, 2008

Version 1.0 / Last edited by Marko Novak / January 18th, 2008

Project co-funded by the European Commission within the Sixth Framework Programme		
Dissemination Level		
PU	Public	√
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Revision history:

Version	Date	Authors	Institution	Section affected, comments

Contents

1 NAT traversal **3**

1.1 NAT traversal by using combination of STUN/TURN/ICE protocols 3

1.2 NAT traversal by using various tools for establishing Virtual Private Network (VPN) 4

1.3 NAT traversal by using Teredo protocol 4

Chapter 1

NAT traversal

There are a couple of ways to perform NAT traversal (for a description about NAT traversal, see [5]):

1.1 NAT traversal by using combination of STUN/TURN/ICE protocols

STUN (Simple Traversal of UPD over NATs) [10, 11] is a protocol for mapping private IP address to a corresponding <gateway's public IP, port number> pair. If a given computer publishes this pair in the web, remote computers can establish connection with this computer (ok, this is not that simple. Some additional actions have to be performed in order for remote computers to be able to establish connection successfully...).

Unfortunately, STUN doesn't work with all types of NAT. It will not connect two machines behind a "symmetric NAT" (you can educate yourself about various NAT methods at [4]). In such cases TURN protocol has to be used.

TURN (Traversal Using Relay NAT) [15, 16] allocates a public IP/port on a globally reachable server and uses it to relay data between communicating parties. The sender A which wants to send data to a receiver B (which is behind NAT) doesn't send it to B directly but sends it to a public relay server R instead. When R receives the data it forwards it to B. This protocol works for all the NAT types. However, to make it work, we need a set of publicly available TURN servers which are not behind NAT. All the communication data go through these servers, which means they can be very heavily loaded. Furthermore, these servers are single-point-of-failure: if they go down, the communication becomes impossible.

ICE (Interactive Connectivity Establishment) [1, 2] is a framework that defines how to use the STUN and TURN protocols to solve the NAT traversal problem, by choosing the best possible interconnection method between two users. Since ICE incorporates STUN and TURN methods, sometimes ICE is also used to refer to the complete STUN, TURN, and ICE solution.

A very good whitepaper which illustrate how to establish connection using STUN/TURN/ICE protocols can be found in [6]. There is also a paper which describes various ways of NAT Traversal [8].

There is also an open-source library which implements STUN, TURN and ICE protocols. Its name is PJNATH and can be found in [9].

1.2 NAT traversal by using various tools for establishing Virtual Private Network (VPN)

. There are a couple of tools for establishing VPN. These tools also include functionality for traversing NAT:

- OpenVPN [7]: in combination with TUN/TAP [14], OpenVPN provides a TCP connections over UDP, where the OS network stack is processing the traffic, which sums up to fully fledged TCP sockets.
- VTUN (Virtual Tunnels over TCP/IP networks) [17]

1.3 NAT traversal by using Teredo protocol

[Editor's note: Luis Pablo from TID suggested this technology as a possibility for doing NAT traversal, so perhaps he could be able to provide a more detailed description of Teredo protocol.]

Although Teredo [12, 13] was developed by Microsoft, it is open protocol. It provides NAT traversal (in the same vein as STUN) and IPv6 traffic over IPv4 networks. It is used by Microsoft in WinXP SP2 and Vista to provide IPv6 support. There is also a Linux implementation of Teredo protocol which is called Miredo [3]. It is already included in Debian Etch.

Teredo is also recommended by the IPv6 Task Force as a transition method to obtain IPv6 connectivity.

Advantages of Teredo (according to Luis Pablo):

- provides NAT traversal
- applications gain IPv6 connectivity in a transparent way (provided, of course, that the application already supports IPv6)
- theoretically, it would even allow for a server inside a NAT to receive connections without further configuration of the NAT device
- it's completely transparent for the applications (you do not need to link any additional library, nor recompile, as Teredo appears as just another network interface)

- implementation of Teredo protocol already exists, so no extra effort is required (except if we want to integrate mobile nodes, maybe, see below)

Disadvantages (according to Luis Pablo):

- requires certain amount of infrastructure in the network(s): Teredo servers (there are already servers deployed on the Internet), Relay servers (to communicate with pure IPv6 networks)
- Requires the Linux kernel to be compiled with certain options
- the interactions between Teredo and Mobile IPv6 still remain to be investigated
- IPv6 addresses are not really static (they are reassigned each time the NAT outgoing port changes), perhaps this could be an issue with certain services and applications
- if we want mobile nodes with MIPv6 and Teredo, we will probably have to modify not only Teredo implementation, but also the MIPv6 implementation

Bibliography

[1] Ice. http://en.wikipedia.org/wiki/Interactive_Connectivity_Establishment.

[2] Ice protocol specifications. http://www.ietf.org/internet-drafts/draft-ietf-mmusic-ice-19.txt.

[3] Miredo. http://www.remlab.net/miredo/.

[4] Nat. http://en.wikipedia.org/wiki/Network_address_translation.

[5] Nat traversal. http://en.wikipedia.org/wiki/NAT_traversal.

[6] Nat traversal for voip and internet communications using stun, turn and ice. http://www.eyeball.com/technology/whitepapers/EyeballAnyfirewallWhitePaper.pdf.

[7] Openvpn. http://openvpn.net/.

[8] Peer-to-peer communication across network address translators. http://pdos.csail.mit.edu/papers/p2pnat.pdf.

[9] Pjnath library. http://www.pjsip.org/pjnath/docs/html/index.htm.

[10] Stun. http://en.wikipedia.org/wiki/STUN.

[11] Stun protocol specifications, rfc 3489. http://tools.ietf.org/html/rfc3489.

[12] Teredo. http://en.wikipedia.org/wiki/Teredo_tunneling, http://www.microsoft.com/technet/network/ipv6/teredo.mspx.

[13] Teredo protocol specifications. http://tools.ietf.org/html/rfc4380.

[14] Tun/tap. http://en.wikipedia.org/wiki/Tuntap.

[15] Turn. http://en.wikipedia.org/wiki/Traversal_Using_Relay_NAT.

[16] Turn protocol specifications. http://www.jdrosen.net/papers/draft-rosenberg-midcom-turn-08.txt.

[17] Vtun. http://vtun.sourceforge.net/.