Establishing direct connection between two remote wireless clients

B. Almási*
* University of Debrecen/Faculty of Informatics, Debrecen, Hungary

Abstract— **Wireless clients are usually placed into private address realm (private network) in the Internet. The clients in the private address realm are able to communicate to each other in the same private network, and using special boxes (NAT boxes) the clients can also communicate to the public internet. This solution "hides" the wireless clients from the outside public internet world: they are unreachable from the outside world, or from other private networks.**

In special situations the requirement of the direct connection between two wireless clients (located in different private networks) may occur. One widely used solution for this kind of problem is the special configuration of the NAT boxes (called "port forwarding"), but also there are many cases, where the NAT box configuration is not allowed (e.g. the applied security policy does not allow it).

In this paper we would like to introduce a software based solution for the mentioned situation: The solution (named UDPTUN) establishes a direct tunnel connection between two clients located in different private networks (without changing or touching the configuration of the NAT boxes).

I. INTRODUCTION

Investigating the wireless communication technologies is a very popular research area today. Wireless clients and wireless sensors are usually located in private networks, i.e. these equipments use private addresses from the address range of 192.168.0.0/16, 172.16.0.0/12, 10.0.0.0/8 (see [1]). Wireless equipments in a private network are able to perform IP based communication to each other and using the "Network Address Translation" (NAT, see [2]) technology they are able to communicate to network nodes in the global address realm. The equipment performing the NAT technology (usually named as "NAT box") hides the wireless clients from the global internet, i.e. two wireless clients (located in different private networks) are unable to communicate to each other through the global internet. In this paper we investigate this kind of connection possibilities. Basically, considering the two widely used transport layer protocols TCP and UDP (see [3,4]), there are relatively easy solutions for the case using UDP protocol (see [5]). Unfortunately, the TCP connection of two remote wireless clients suffers from quite serious difficulties: The TCP connection establishment process (3 way handshake) can not be performed between the two remote clients.

One widely used solution for this kind of problem is the special configuration of the NAT box (called "static NAT table entry" or "port forwarding"), but also there are many cases, where the NAT box configuration is not allowed (e.g. the security policy of the organization does not allow it).

In this paper we introduce a software based solution (named as UDPTUN) which can be used to create direct TCP connection between remote clients without the configuration of the NAT box.

II. OVERVIEW OF THE NAT TECHNOLOGY

Network Address Translation (NAT, or sometimes referred as Network Address and Port Translation, NAPT) is a widely used technology to spare the globally used IP numbers. The basic idea of NAT was introduced in [6]: The IP address reusing idea allows organizations using the same private address range inside the organization, so giving a short-term solution for the IP address depletion problem. The private address ranges can be used for identifying and addressing the wireless clients inside an organization without outside address registration (see [1]).

A wireless client using a private IP address should be able to communicate with servers located in the global internetwork. A so called "NAT box" must be installed at the border of the private and global network. The NAT box changes the private addresses into global ones when the packets traverse through it into the direction of the global world. The data of the address change is stored in the NAT translation table. To spare the global IP addresses, the NAT box may change not only the IP addresses, but the port numbers too (overloaded NAT, or sometimes named as NAPT).

Basically, the clients located in the private address realm (i.e. behind the NAT box) can not be reached with a connection establishment request from the global network: concerning the situation represented on Figure 1. both "Node A" and "Node B" are able to establish connection to servers located in the Global Internet, but it is not possible in the opposite direction, and so "Node A" and "Node B" can not establish a TCP session to each other.

III. STATIC NAT ENTRY

The problem mentioned in the previous section is not new. The NAT box producers usually build the so called "virtual server" feature into the equipment, which is able to handle the introduced problem. The different vendors use different names for this technology; e.g. "port forwarding", "static NAT", "static PAT" (Port Address Translation) and "virtual server" expressions refer to the same solution idea.

In this solution a static NAT table entry is created by the system administrator of the NAT box, which opens the possibility of establishing TCP connection into a specified node of the private network. The static TCP NAT table

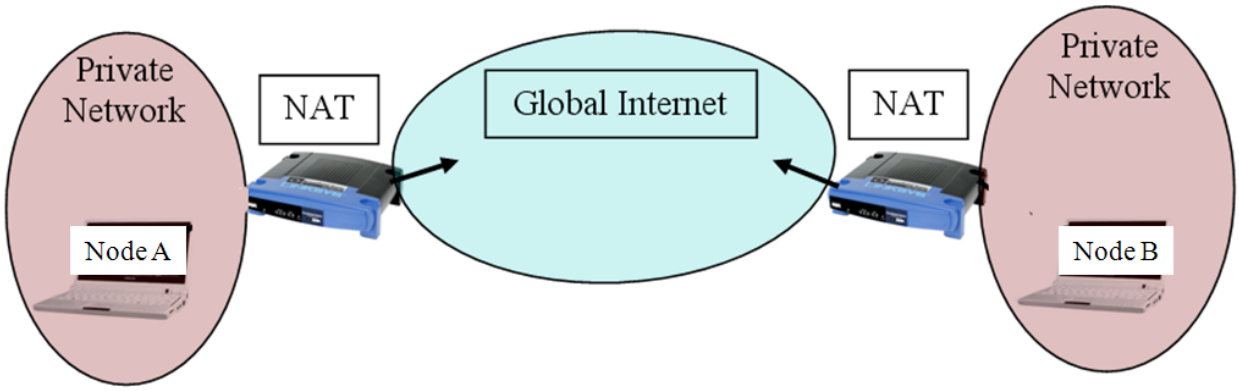

Figure 1. The NAT Technology

entry contains four important data fields: the global IP number (in the case when the NAT box has got more IP addresses, it identifies the actually used one), the global TCP port number, the private IP number (it identifies the node in the private address realm), and the private port number (which identifies the application).

A port forwarding example is shown in Figure 2. The working mechanism is the following: The administrator of the NAT box "B" creates a static NAT entry (see Table I.). When a packet arrives to the NAT box from the global world addressed to 125.6.7.8 containing the destination TCP port number of 13450, then the packet will be sent to the private network. The destination IP address is changed to 172.22.3.4 and the destination TCP port number is replaced with 2380. On the other hand, the dynamic NAT entries are applied as usually, i.e. all of the private nodes, and wireless clients are able to communicate to the servers located in the global address realm.

TABLE I.
ADDRESS TRANSLATION TABLE

Private IP & Port number	Global IP & Port number
172.22.3.4 : 2380 (TCP)	125.6.7.8 : 13450 (TCP)

The static NAT entry shown in Table I. opens the communication way for "Node A" to reach "Node B": "Node A" is able to open a communication session to any destination of the global internet. It means, that "Node A" can reach the IP address of 125.6.7.8 with the destination TCP port number 13450. When this packet arrives to the NAT box (shown at the right side of the picture), then the NAT box will apply the static NAT entry of Table 1 to direct this packet to "Node B". Port forwarding gives a good solution in many cases, but we may not forget, that it requires special configuration of the NAT box. There are many cases, when the NAT box configuration can not be performed (e.g. we do not have administration right over the Nat box).

The following section gives a solution example for these cases.

IV. THE THEORETICAL BACKGROUND OF UDPTUN

Reference [7] introduced the basic idea of establishing UDP communication between two nodes, located in different private networks:

Both wireless clients start flooding UDP packets to the peer's global address. These packets will create a dynamic NAT box entry, when they leave the sender's private network. When the peer's packet arrives, it will be directed to the right private address, according to the NAT box entry created before. Determining the peer's global address and port number is an interesting and sometimes difficult question (see [5]). In this paper we do not work on the global address determination process, we assume, that we could finish it, and the traversal of the UDP packet works.

The basic idea of the UDPTUN solution is the following: We reserve a UDP port number to create a

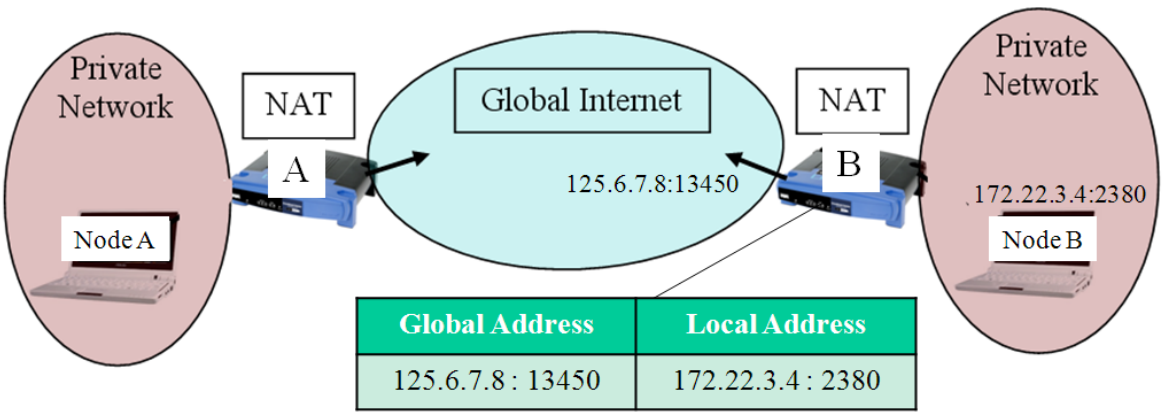

Figure 2. Port Forwarding

UDP connection between the wireless clients. A logical tunnel interface is created on both sides, and the packet tunneling mechanism will be set up according to the following rules: When an application sends an IP packet to the tunnel interface, then the whole IP packet will be encapsulated into a UDP segment using the port number of the created UDP connection. This UDP packet will be sent to the peer's global address. When a packet arrives to the reserved UDP port number (i.e. to the created UDP connection), then the data part of the incoming UDP segment (i.e. the IP packet which was encapsulated on the sender's side) will be sent to the logical tunnel interface. The tunnel interfaces communicate to each other as if they were directly connected. The necessary assumption of this solution is, that the UDP NAT traversal works and is active at both sides.

The UDP connection use dynamic NAT box entries, which are usually deleted from the address translation table, if no packet transfer occurs for a specified time. In order to keep the created UDP connection active (i.e. not to loose the dynamic NAT entries), we must send "empty" UDP packets to the peer regularly when there is no application activity through the tunnel interface.

V. THE UDPTUN APPLICATION

We implemented the UDPTUN mechanism in Linux and Microsoft Windows XP and Windows7. The Linux kernel 2.6 already contained the tunnel interface implementation, so Linux supports the tunnel interface natively.

In Microsoft Windows XP (and also in Windows7) basically there is no tunnel interface support, but a third party tunnel interface implementation can be found in the OpenVPN software package (see [8]). The tunnel interface is a well separated package in OpenVPN, and it can be installed alone, without the full installation of the OpenVPN software. Some older version of the WinTAP32 tunnel interface can be found independently of the OpenVPN package too.

In both operating systems there are two types of tunnel interfaces: the 'tap' and the 'tun' type. The 'tap' type tunnel interface represent an Ethernet-like interface, while the 'tun' type tunnel interface represent a Point-to-Point like interface. UDPTUN can be used with both types, but the same type must be used at the tunnel endpoints in order to get a successful connection.

The statically linked Windows XP/7 binary executables of the UDPTUN software can be downloaded from http://irh.inf.unideb.hu/user/almasi/udptun. We must have full administrator right to use it (root/administrator right is necessary in Linux too).

VI. THE USAGE OF THE SYSTEM

Starting the udptun executable program without parameters will show the usage syntax possibilities (see Figure 3.). The short syntax version of udptun contains only the name of the tunnel interface (e.g. tap1 or tun1). The name of the interface specifies also the type of it.

The second parameter is the IP address of the tunnel interface. Udptun will configure the tunnel interface with this IP address and netmask of 255.255.255.0.

The third parameter is the global IP address of the peer, i.e. the outgoing UDP packets will be sent to this destination address. Using the short syntax form we assume, that the UDP port numbers are the same. The default value of this common UDP port number is 33456, or it can be specified as the 4th command-line parameter.

The long syntax version of udptun allows to specify all the data related to the tunnel establishment (see Figure 3.).

After issuing the udptun command with the right parameters it configures the tunnel interface and begins to send UDP packets to the peer's global address (so creating the dynamic NAT table entry in the local NAT box). When the peer's UDP packet arrives, the tunnel is established, and the applications can use the tunnel interface for direct TCP connection to the peer.

If there is no application activity, then an empty

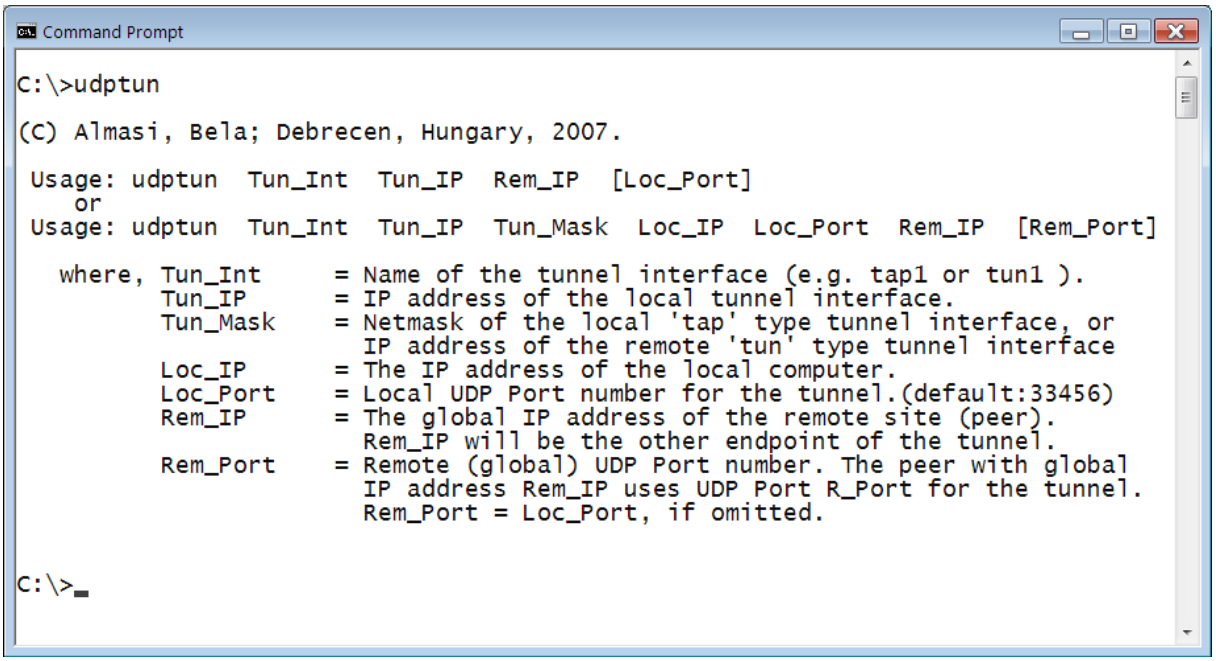

Figure 3. UDPTUN Syntax

'keepalive' message is sent to the peer in every 10 seconds, so ensuring that the dynamic NAT entries will not be deleted

VII. SUMMARY

In this paper we considered the problem of creating direct TCP connection between two remote wireless clients, located in different private networks. Although static NAT table entries can solve the problem, there are cases, when a solution must be created without NAT box configuration.

We introduced a software based solution: UDPTUN. It establishes a direct tunnel connection between the two remote clients, based on UDP NAT traversal.

The binary executables of UDPTUN can be downloaded from the URL:

http://irh.inf.unideb.hu/user/almasi/udptun

REFERENCES

[1] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot, E.Lear, "Address Allocation for Private Internets," *RFC 1918*, February 1996.

[2] P. Srisuresh, K. Egevang, "Traditional IP Network Address Translator (Traditional NAT*),*" *RFC 3022*, January 2001.

[3] J. Postel, "TRANSMISSION CONTROL PROTOCOL", *RFC 793,* September 1981.

[4] J. Postel, "User Datagram Protocol", *RFC 768,* August 1980.

[5] J. Rosenberg, R. Mahy, P. Matthews, D. Wing, "Session Traversal Utilities for NAT (STUN)," *RFC 5389,* October 2008.

[6] K. Egevang, P. Francis, "The IP Network Address Translator (NAT)," *RFC 1631,* May 1994.

[7] J. Weinberger, C. Huitema, R. Mahy, "STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs)," *RFC 3489,* March 2003.

[8] The OpenVPN Project, *http://openvpn.net/index.php/open-source.html*, OpenVPN Technologies Inc., 2011.