1. Introduction

The goal of this project is to design, deploy and demonstrate a basic ISP in the lab environment. Our ISP (further called ISP2) would provide a few services which come in two parts: mandatory services and selective services.
The mandatory services are those services which provide almost the basic functionality of every ISP; e.g Dynamic IP Routing, DNS, DHCP, etc.
The selective services are those to provide additional functionalities which are mostly for special purposes and not ubiquitous in all ISPs; e.g. Voice over IP.

In this project we have been confined to using a limited set of equipments, and consequently our design is influenced by the amount and type of those equipments. The equipments we have had are as follows:
3 routers (1 cisco 2514 and 2 cisco 2501)
2 HP2524 switches
3 Dell Laptops
10 UTP straight cables
5 UTP crossed cables

of which we haven't used one switch in our design, and we have mostly used our laptops as for servers and clients.

The members of our team as as follows:

Hamidreza Khajeh Alizadeh Attar - hrkaa@kth.se
Gang Zhang - edgerz@gmail.com
Rickard Vestermark - rves@kth.se
M. Usman Minhas - mminhas@kth.se
Anshuman Aggarwal - anshuman.iitdelhi@gmail.com
Pablo Urribarri - p_urribarri@hotmail.com
Md. Safiqul Islam - islam3@kth.se

2. Network Design

2.1. Network Topology

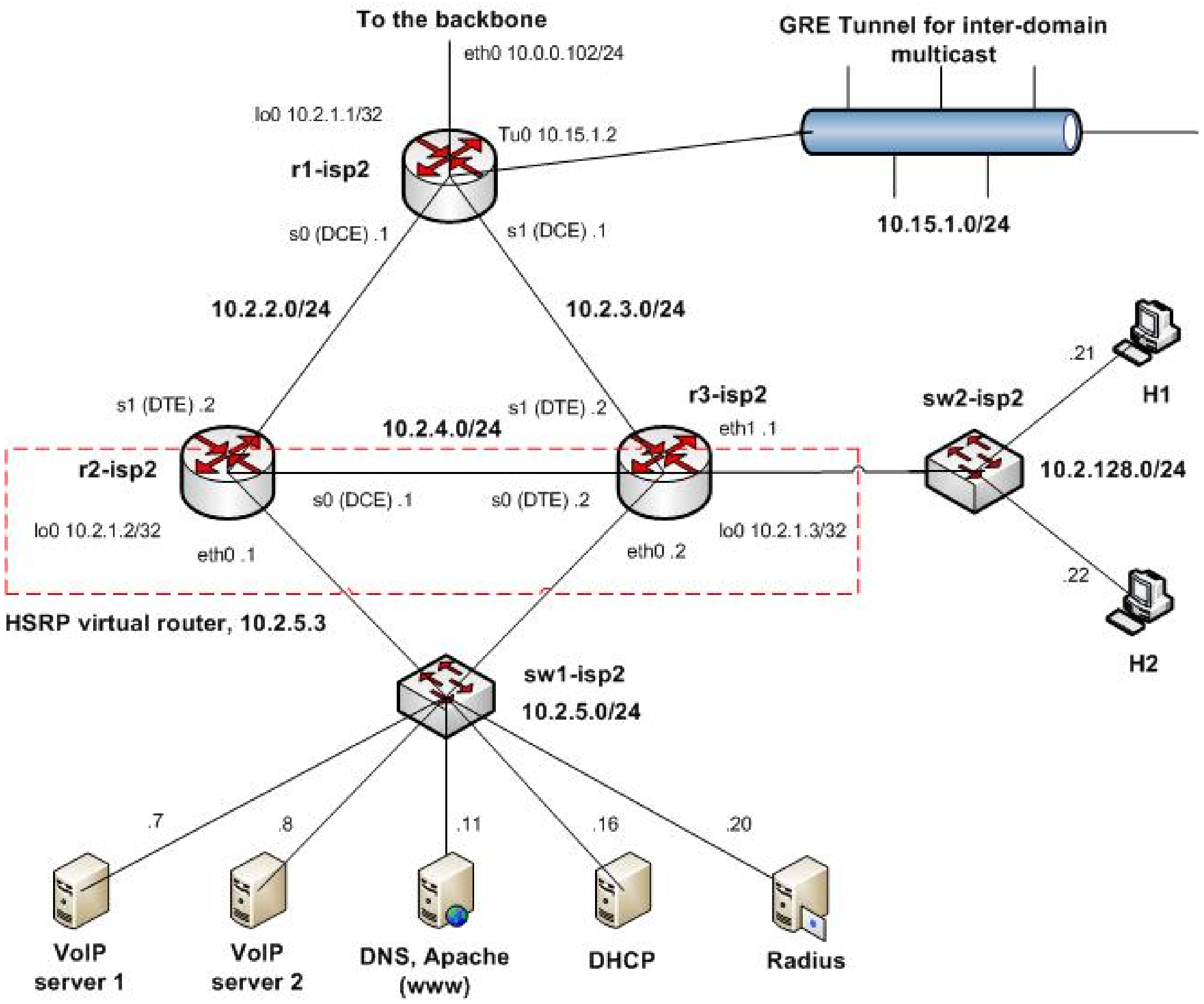


Figure 1: Network topology of ISP2

2.2. Explanation of Design and addressing scheme

The design of the topology is based on the proposed topology for the project. The ISP2 has a "internal" backbone of 3 routers, with two Ethernet LANS for a server and a client network respectively. The topology is shown in figure 1.
The server network is in a redundant Ethernet LAN conected to two routers, while the client network is in another Ethernet LAN connected to one router through one link.
For the tasks of this project it was enough to use 3 routers (two of them were used as a virtual router for the server park) and two switches, one for the client network and the other for the server network.
The addressing scheme was based on classless addressing. The whole network has the prefix 10.2.0.0/16. The server LAN is in the subnet 10.2.5.0/24, while the client network is in the subnet 10.2.128.0/24.
We are using loopback addresses for the routers (the subnet 10.2.1.0/24)
And finally the point-to-point links between routers and switches are also using /24 adresses (see figure 1 for the details). We didn't bother to spare adresses, because it was not an issue.

3. Services

3.1. Mandatory Services

3.1.1. Dynamic IP Routing

For dynamic routing we had no too much choice. IS-IS was not available on the routers and RIP will not be the best solution, thinking in redundancy. So the natural choice was OSPF.
Since we had limited the topology to 3 routers, it was enough to use one area, i.e. the backbone area (area 0).
We didn't care about forcing the routers to become DR or BDR because the 3 routers build a full mesh and the links are point to point, thus there was not profit in having a special DR and BDR.
The OSPF ASBR is the router r1-isp2, and following the requirements for this project this router is not exporting the internal topology to the backbone connecting to the other ISPs using dynamc routing. Instead it was used static routing for this task. On the contrary the router r1-isp2 has the task to inject (import) routes towards the other ISPs /16 networks.
The complete configuration of the routers is presented in the Appendix.

3.1.2. Fault-Tolerant IP Routing

For fault tolerance towards the server network it was used HSRP. Our attempt to build load-balancing based on HSRP failed because of the limitations of the available routers.
Fault tolerance consisted of simply assuring that only one of the routers will be able to be the active one at any time, under the condition that the Ethernet interface (and link to this router) was up and the priority of this interface was higher than the default priority of the other router's Ethernet interface towards the switched server network. See figure 1.
The configuration of the ethernet links of router r2-isp2 and r3-isp2 for enabling fault tolerance based on the protocol HSRP are presented in the Appendix.

Below is the required configuration for enabling fault tolerance with HSRP. In this particular case the router r3-isp2 becomes active, while the router r2-isp2 becomes passive, if everything is up and running towards the server switched network.

```
at r2-isp:
conf t
interface Ethernet0
 ip address 10.2.5.1 255.255.255.0
 standby 1 ip 10.2.5.3
 !standby 1 priority 90
 standby 1 preempt
 no shutdown
```

```
at r3-isp:
conf t
interface Ethernet0
 ip address 10.2.5.2 255.255.255.0
 standby 1 ip 10.2.5.3
 standby 1 priority 110
 standby 1 preempt
 no shutdwon
```

3.1.3. IP Multicast Routing

For the mandatory part, we've enabled PIM-SM with static RP on our three routers. Three commands are required:

```
    ip multicast-routing   // global config mode on all routers
    ip pim sparse-mode   // on all interfaces except r1 eth0 (backbone interface)
    ip pim rp-address 10.2.1.1    // global config mode on all routers
```

We configured r1 as the RP.

3.1.4. DHCP

DHCP: Dynamic address assignments (Stateful : We have a central DHCP server which keeps track of the addresses currently in use)
We have a dhcp server 10.2.5.16 which has to be manually assigned an ip address and also be provided with a gateway, which is 10.2.5.3 (virtual router) to communicate with the network.
The messages sent from the dhcp server can therefore reach any destination, as the virtual router guides it to the nearby routers, and then they hav the routing tables.
The dhcp.conf (located in /etc/dhcp3/) file is as follows..

```
option domain-name "isp2.lab";
option domain-name-servers 10.2.5.11;
log-facility local7;
ddns-update-style none;

subnet 10.2.5.0 netmask 255.255.255.0 {
option routers 10.2.5.3;
default-lease-time 144000;
max-lease-time 1728000;
host dns {
hardware ethernet 00:19:b9:6c:c0:db;
fixed-address 10.2.5.11;
}
host www {
hardware ethernet :::::;
fixed-address 10.2.5.21;
}
}
subnet 10.2.128.0 netmask 255.255.255.0 {
default-lease-time 14400;
max-lease-time 172800;
range 10.2.128.21 10.2.128.254
option routers 10.2.128.1;
}
```

- The ddns style can be set to interim for auto-updates to DNS.
- Lease times are kept different for different subnets. The client leases are shorter while the server side has longer leases to make it stable.
- We assign the server ip address from the DHCP, which is a better than designing them manually.
- We can include a set of ips on the server side, if we plan to connect some more machines on the server side.

Apart from this file, we need to do "touch /var/lib/dhcp3/dhcp.leases".
and then start the server by running the command /etc/init.d/dhcp3-server start.
The relay agent is the router, which are given helper addresses to convey DHCP discover messages to the server.
The clients can run dhclient or anything similar to get ip addresses.
DHCP messages captured on Ethereal include the standard DHCP messages, i.e. DHCP Discover, Offer, Request and Ack.

3.1.5. DNS

DNS was used to resolve ip addresses to assigned names and viceversa for all routers, servers and hosts of the ISP2 domain. DNS was used even for providing SRV record for the VoIP service.

To run DNS on a Linux machine, we need three files:
1. named.conf - configuration file for DNS service;
2. isp2.lab - zone record file for domain name -> ip address mapping and
3. 2.10.in-addr.arpa - reverse zone record file for ip address -> domain name mapping.

named.conf

```
options {
 # defines the name server's working directory
 directory "/var/lib/named";

 # local network interfaces to listen on
 listen-on port 53 { 10.2.5.11; };

 # upstream name servers to which queries should be forwarded
 forwarders { 10.0.0.11; 10.0.0.21; };
};

 # log queries and default information to syslog
logging {
 category queries { log_syslog; };
 category default { log_syslog; };
 channel log_syslog { syslog; };
};
```

```
 # zone records for domain name -> ip address mapping
zone "isp2.lab" in {
 # record file and its path relative to working directory
 file "master/isp2.lab";

 # zone type
 type master;
};

# reverse zone records for ip address -> domain name mapping
zone "2.10.in-addr.arpa" in {
 file "master/2.10.in-addr.arpa";
 type master;
};
```

isp2.lab

```
$TTL 2d
  # Start Of Authority record, specify domain serial number, and several timers relating
to refreshing
@  IN SOA  EdgeStation.isp2.lab. root.EdgeStation.isp2.lab. (
    2007101400 ; serial
    3h  ; refresh
    1h  ; retry
    1w  ; expiry
    1d )  ; minimum

  # Name Server record, specify domain name server
isp2.lab. IN NS  ns.isp2.lab.

  # Service record for VoIP
_sip._udp IN SRV  0 1 5060 sip1.isp2.lab.
             IN SRV  0 1 5060 sip1.isp2.lab.

  # Address record for domain name -> ip address mapping
ns  IN A  10.2.5.11  ; DNS server, 'ns' stands for ns.isp2.lab.
dhcp  IN A  10.2.5.16  ; DHCP server
www  IN A  10.2.5.11  ; Web server
sip1  IN A  10.2.5.7  ; VoIP servers
sip2  IN A  10.2.5.8
radius  IN A  10.2.5.20  ; Radius server
r1  IN A  10.2.1.1  ; Routers (loopback)
r2  IN A  10.2.1.2
r3  IN A  10.2.1.3
h21  IN A  10.2.128.21  ; Hosts
h22  IN A  10.2.128.22
h23  IN A  10.2.128.23
 .
 .
 .
h252  IN A  10.2.128.252
h253  IN A  10.2.128.253
h254  IN A  10.2.128.254
```

2.10.in-addr.arpa

```
 $TTL 2d
@  IN SOA  EdgeStation.isp2.lab. root.EdgeStation.isp2.lab. (
    2007101400 ; serial
    3h  ; refresh
    1h  ; retry
    1w  ; expiry
    1d )  ; minimum

2.10.in-addr.arpa. IN NS  ns.isp2.lab.

  # Pointer record for ip address -> domain name mapping
11.5  IN PTR  ns.isp2.lab.  ; '11.5' stands for 11.5.2.10.in-addr.arpa.
                            ; 11.5.2.10 is the reverse ip address, and in-addr.arpa. is a
fixed postfix
16.5  IN PTR  dhcp.isp2.lab.
11.5  IN PTR  www.isp2.lab.
20.5  IN PTR  radius.isp2.lab.
7.5  IN PTR  sip1.isp2.lab.
8.5  IN PTR  sip2.isp2.lab.
1.1  IN PTR  r1.isp2.lab.
2.1  IN PTR  r2.isp2.lab.
3.1  IN PTR  r3.isp2.lab.
21.128  IN PTR  h21.isp2.lab.
22.128  IN PTR  h22.isp2.lab.
 .
 .
 .
253.128 IN PTR  h253.isp2.lab.
254.128 IN PTR  h254.isp2.lab.
```

Place named.conf in /etc and isp2.lab, 2.10.in-addr.arpa in the DNS working directory according to the settings in named.conf. then start DNS service using the command "/etc/init.d/named start".

To test DNS service in a Linux machine, configure it use the DNS server you set up and run commands:

```
host www.isp2.lab               test domain name -> ip address mapping;
host 10.2.5.11                  test ip address -> domain name mapping;
dig SRV _sip._udp.isp2.lab       test SRV record
```

3.1.6. WWW - server

To implement a web server (apache) in Suse Linux, just goto yast2 -> Network Services -> HTTP Server, follow the wizard by pressing "next". In wizard (1/5), you can specify the interfaces and ports apache server listens to, in wizard (3/5), you can specify the path of html files by editing "Document Root" option, and that's all we need in this project.
To manage apache server in Linux, use the command "/etc/init.d/apache2 start/stop/ restart".

3.2. Selective Service

3.2.1. VoIP

As VoIP (Voice over Internet Protocol), also called IP Telephony or Internet Telephony. It is used for routing voice conversations over IP-based network. As Telecommunication industry is glooming nowadays we can have better opportunities to work with. By keeping this view in mind we decided to choose the VoIP service.

We choose "Brekeke SIP Server 2.0" as our VoIP server, SIP here stands for Session Initiation Protocol, it is an applicatioin-layer control protocol for creating, modifying and terminating sessions with one or more participants. SIP helps to route caller requests to callee's location, and manage the conversation, the actual voice or video content are carried by other session protocols, typically RTP.
Brekeke SIP Server is a commercial software, so we only have its academic version, which turns out to be rather unstable. We have installed a SIP server on Windows Platform and clients are from Windows based environment (Windows Messenger 5.1) and also from the Linux based environment (Twinkle 0.9). The basic setup of SIP server and client can be found in http://www.brekeke.com/products/products_sip_quickstart.php

VoIP with SRV records:
An SRV record (Service record) is a category of data in DNS, specifying information on available services. An typical SRV record looks like this:
_sip._udp.isp2.lab. 86400 IN SRV 0 1 5060 sip1.isp2.lab.

sip: service name;
udp: protocol, usually either TCP or UDP;
isp2.lab: domain name;
86400: DNS TTL (time to live) field;
IN: DNS class field, always IN;
0: priority of the target server, smaller is better;
1: weight for records with same priority, larger is better;
5060: service port number;
sip1.isp2.lab.: hostname of the machine providing this service.

Load balance and Backup with SRV, for example:
_sip._udp.isp2.lab. 86400 IN SRV **0 80** 5060 sip1.isp2.lab.
_sip._udp.isp2.lab. 86400 IN SRV **0 20** 5060 sip2.isp2.lab.
_sip._udp.isp2.lab. 86400 IN SRV **1 0** 5060 backup.isp2.lab.

Because of its priority (1 > 0), backup will never be used unless both sip1 and sip2 are down. For sip1 and sip2, when they are both up, sip1 will be used for 80% of requests, sip2 will be used for other 20%.
Load balance provided by SRV is essentially static, current load of servers is not taken into account.

To enable support for SRV record,
server: goto SIP server admin page, -> Configuration -> SIP(General) -> Authentication, set Realm = "isp2.lab"
client: From Windows Messenger menu, -> Tools -> Options -> Accounts -> SIP Communication Service enter "sip:201@isp2.lab" in Sign-in name field; -> Advanced, choose Auto Configuration.

Without SRV record, VoIP clients must be configured with VoIP server's ip address or FQDN (sip1.isp2.lab); with SRV record, clients only need to know the domain name (isp2.lab), DNS server will help them find out a specific server.

Interdomain VoIP:
Without dial plan, when VoIP user inside our isp2 domain wants to call users in other isp domain, say, user 333 in sip.isp3.lab, he has to call like this: "333@sip.isp3.lab"; with dial plan, user can just call "333", our VoIP server will examine the calling number

according to dial plan, aware it's a interdomain call, and find out the corresponding VoIP server in isp3.lab, then forward the calling request there.

3.2.2. Advanced IP Multicast

As one of our selective services, we've chosen to extend our multicast configuration with auto-RP and tunnel configuration for inter-domain multicasting.

For auto-RP multicast two roles need to be assigned - mapping agent(s) and RP(s). It's possible to have multiple mapping agents and RPs for redundancy and load balancing.

An RP announce its presence using 224.0.1.39. The mapping agent listens for RP announcements, and in turn sends RP-to-group mappings to 224.0.1.40 (in discovery messages). The remaining routers in the domain use these messages for their RP-to-group mapping.

```
   ip multicast-routing   // global config mode on all routers
   ip pim sparse-mode   // on all interfaces except r1 eth0 (backbone interface)

   ip pim send-rp-announce loopback0 scope 16   // on r1, RP ("scope 16" defines the
maximum hop count validity of the announcement)
   ip pim send-rp-discover scope 16  // on r1, mapping agent
```

For inter-domain multicast we configured a GRE tunnel to the other ISP (since we were not allowed to send any multicast traffic directly on the backbone network). The tunnel will encapsulate and unicast the multicast traffic over the backbone (transparently to the application).

```
   interface Tunnel0
   ip address 10.15.1.2 255.255.255.0
   ip pim sparse-mode
   tunnel source Loopback0
   tunnel destination 10.8.6.1   // the loopback interface address of the ISP border
router on the other end of the tunnel
```

To redirect the multicast traffic from our network to the tunnel, we simply added a multicast route entry on r1:

```
   ip mroute 0.0.0.0 0.0.0.0 Tunnel0
```

But if we want to send our multicast to two ISP. Then we have to create another tunnel. and we have to specify the mroute command for the ISPs.

Another tunnel for group 11..

```
interface tunnel1
ip address 10.16.1.2 255.255.255.0
ip pim sparse-mode
tunnel source looback0
tunnel destination 10.11.X.X
```

To redirect the multicast to isp 8 and isp 11

```
ip mroute 10.11.0.0 255.255.0.0 tunnel0
ip mroute 10.8.0.0 255.255.0.0 tunnel1
```

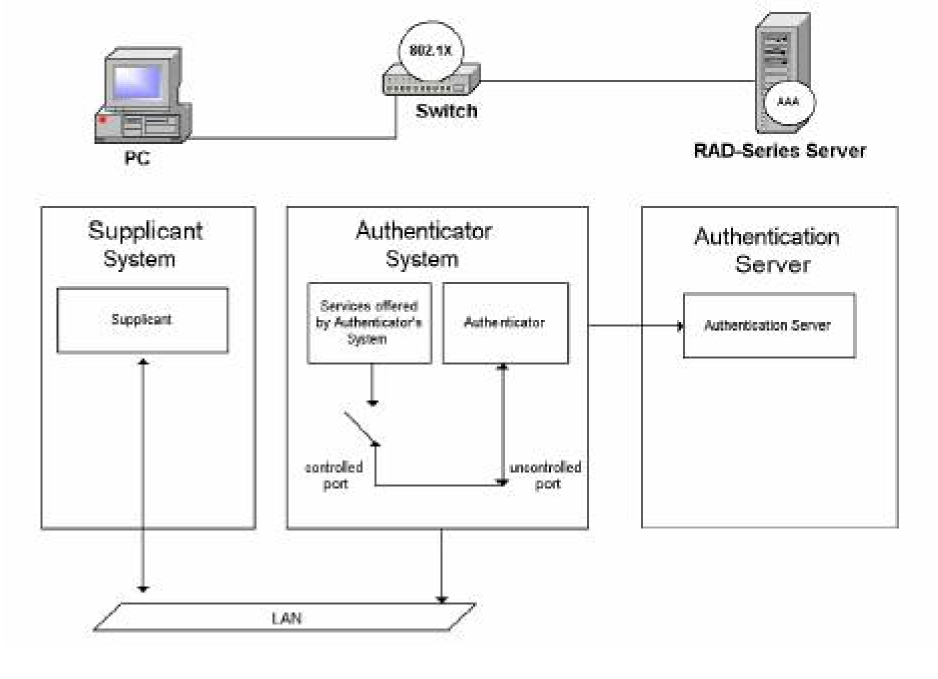
3.2.3. Port-Access Authentication

The IEEE 802.1X standard defines port-based, network access control that is used to provide authenticated network access for Ethernet networks. Port-based network access control uses the physical characteristics of a switched LAN infrastructure to authenticate devices that are attached to a switch port. The ability to send and receive frames using an Ethernet switch port is denied if the authentication process fails. This standard is designed for both wired Ethernet networks and IEEE 802.11 wireless LANs.

To provide a standard authentication mechanism for IEEE 802.1X, IEEE chose the Extensible Authentication Protocol (EAP). EAP is a Point-to-Point Protocol (PPP)-based authentication technology that was adapted for use on point-to-point LAN segments. Because EAP messages were originally defined to be sent as the payload of PPP frames, the IEEE 802.1X standard defines EAP over LAN (EAPOL), which is a method of encapsulating EAP messages so that they can be sent over Ethernet or wireless LAN segments.[1]

In the authentication process, based on 802.1X terminology, the Ethernet switch (NAS) acts as the *authenticator*, the device which is going to be connected to the network is known as *supplicant*, and the back-end server which has gotten a shared secret with the device and performs the actual authentication is named as *authentication server*.

As soon as the Ethernet network cable in connected to the Ethernet switch, the switch starts the authentication process with the device over EAPOL messages. On the other side, the switch translates and sends the messages back and forth to the back-end authentication server, which may be a central server on the network, or, in its simplest form, a table of username and passwords stored on the switch's memory. The following figure shows these relations [2]:



In our design we have decided to have a central server, because from the network administration point of view it simplifies the task of management and administration of the authentication credentials. It also enables the administrators to integrate their current authentication database to be used as for the port-access control mechanisms as well. On the other hand, having localy-stored username/passwords in switches means that every switch must be configured with the credentials table individually, that consequently brings to the administrators the workload of maintaining and protecting the integrity of these tables, and reconfiguring the switches individually in case of addition, deletion, or modification of any username/password. Also, from the security point of view, having localized credentials introduces weak points on the network which might be compromised more easily than a single highly-protected central server. In case of a central server, as for the single point of failure argument and if for a given network it is a concern to avoid single point of failures, the solution is to install another one or two authentication server(s) on the network and configure these servers to replicate their data, and then it would be possible to configure the UP2524 switches with up to three radius servers which would be consulted in order, in case the first one goes down.

Planning for central credential repository, it is very common for the authentication server to be a Radius server. In this case, the authenticator (switch) acts just as a translator, talking EAPOL to the supplicant and EAP-over-Radius to the Radius server, and eventually the final result of this interaction would be judged by the Radius server, which would be either Access-Accept or Access-Reject, and would be reported to the switch to either allow the access or deny it.

The authentication method (or methods) used for this purpose must be agreed upon in advanced, and must be configured in the server, switch and the client. There are a bunch of these methods defined, namely EAP-TLS, PEAP-TLS, PEAP-MS-CHAP v2, MD5-CHAP, etc. In this project, as for the demonstration purpose, we have used MD5-CHAP.

The Radius server we have used as our authentication server is FreeRADIUS implementation on Linux (Fedora Core 7). We had other alternatives such as BSDRadius, GNU RADIUS, JRadius, OpenRADIUS, Radiator, etc, among which FreeRADIUS comes as an optional package with Fedora Core 7, and also it seems to be recommended more often by the other users on the internet, and these were the reasons we preferred FreeRADIUS over the other alternatives.

We have tried both Linux and Windows clients as supplicant. The networking module of Windows XP SP1 (and higher) already has gotten 802.1X supplicant built-in. In Linux (Debian), we have used XSupplicant software which is a very common 802.1X client on Linux platforms; it is an open source software under GNU General Public License (GPL).

[1] http://www.microsoft.com/technet/community/columns/cableguy/cg0402.mspx
[2] http://www.interlinknetworks.com/whitepapers/Intro_802_1X_for_Wireless_LAN.htm

4. Appendix

4.1 Configuration of the routers

```
------------------------------------------------
Configuration of r1-isp2, the ASBR
------------------------------------------------
conf t
!
hostname r1-isp2
!
enable password gr2
!
!
clock timezone CET 1
```

```
ip subnet-zero
no ip domain-lookup
!
ip multicast-routing
!
!
interface Loopback0
 ip address 10.2.1.1 255.255.255.255
 ip pim sparse-mode
!
interface Tunnel0
 bandwidth 4096
 ip address 10.15.1.2 255.255.255.0
 ip pim sparse-mode
 tunnel source Loopback0
 tunnel destination 10.8.6.1
!
interface Tunnel1
 bandwidth 4096
 ip address 10.16.1.2 255.255.255.0
 ip pim sparse-mode
 tunnel source Loopback0
 tunnel destination 10.11.4.1
!
 interface Ethernet0
 bandwidth 10000000
 ip address 10.0.0.102 255.255.255.0
!
interface Serial0
 bandwidth 4096000
 ip address 10.2.2.1 255.255.255.0
 ip pim sparse-mode
 clockrate 800000
!
interface Serial1
 bandwidth 4096000
 ip address 10.2.3.1 255.255.255.0
 ip pim sparse-mode
 clockrate 800000
!
router ospf 100
 log-adjacency-changes
 passive-interface Ethernet0
 network 10.0.0.0 0.0.0.255 area 0
 network 10.2.1.1 0.0.0.0 area 0
 network 10.2.2.0 0.0.0.255 area 0
 network 10.2.3.0 0.0.0.255 area 0
 default-information originate always
!
ip classless
ip route 10.1.0.0 255.255.0.0 10.0.0.101
ip route 10.3.0.0 255.255.0.0 10.0.0.103ip route 10.4.0.0 255.255.0.0 10.0.0.104
ip route 10.5.0.0 255.255.0.0 10.0.0.105
ip route 10.6.0.0 255.255.0.0 10.0.0.106
ip route 10.7.0.0 255.255.0.0 10.0.0.107
ip route 10.8.0.0 255.255.0.0 10.0.0.108
ip route 10.9.0.0 255.255.0.0 10.0.0.109
ip route 10.10.0.0 255.255.0.0 10.0.0.110
ip route 10.11.0.0 255.255.0.0 10.0.0.111
ip route 10.12.0.0 255.255.0.0 10.0.0.112
ip http server
ip pim send-rp-announce Loopback0 scope 16
ip pim send-rp-discovery scope 16
ip mroute 10.11.0.0 255.255.0.0 Tunnel1
ip mroute 10.8.0.0 255.255.0.0 Tunnel0
!
!
line con 0
line aux 0
line vty 0 4
 password isp2
 login
!
ntp master
end

---------------------------------------
configuration of r2-isp2
-----------------------------------------
conf t
!
hostname r2-isp2
!
enable password gr2
!
!
ip subnet-zero
no ip domain-lookup
!
ip multicast-routing
!
!
interface Loopback0
 ip address 10.2.1.2 255.255.255.255
 ip pim sparse-mode
!
interface Ethernet0
 bandwidth 1000
 ip address 10.2.5.1 255.255.255.0
 no ip redirects
 ip pim sparse-mode
 standby 1 ip 10.2.5.3
 standby 1 priority 90
 standby 1 preempt
!
interface Serial0
 ip address 10.2.4.1 255.255.255.0
 ip pim sparse-mode
 no fair-queue
 clockrate 64000
!
interface Serial1
 ip address 10.2.2.2 255.255.255.0
 ip pim sparse-mode
!
router ospf 100
```

```
 log-adjacency-changes
 network 10.2.1.2 0.0.0.0 area 0
 network 10.2.2.0 0.0.0.255 area 0
 network 10.2.4.0 0.0.0.255 area 0
 network 10.2.5.0 0.0.0.255 area 0
!
ip classless
ip http server
!
!
line con 0
line aux 0
line vty 0 4
 password isp2
 login
!
ntp clock-period 17180038
ntp server 10.2.1.1
end

----------------------------------------
Configuration of r3-isp2
----------------------------------------
conf t
!
hostname r3-isp2
!
enable password gr2
!
!
ip subnet-zero
no ip domain-lookup
!
ip multicast-routing
!
!
interface Loopback0
 ip address 10.2.1.3 255.255.255.255
 ip pim sparse-mode
!
interface Ethernet0
 ip address 10.2.5.2 255.255.255.0
 no ip redirects
 ip pim sparse-mode
 standby 1 ip 10.2.5.3
 standby 1 priority 110
 standby 1 preempt
!
interface Ethernet1
 ip address 10.2.128.1 255.255.255.0
 ip helper-address 10.2.5.16
 ip pim sparse-mode
!
interface Serial0
 ip address 10.2.4.2 255.255.255.0
 ip pim sparse-mode
 no fair-queue
!
```

```
interface Serial1
 ip address 10.2.3.2 255.255.255.0
 ip pim sparse-mode
 no fair-queue
!
router ospf 100
 log-adjacency-changes
 network 10.2.1.3 0.0.0.0 area 0
 network 10.2.3.0 0.0.0.255 area 0
 network 10.2.4.0 0.0.0.255 area 0
 network 10.2.5.0 0.0.0.255 area 0
 network 10.2.128.0 0.0.0.255 area 0
!
ip classless
ip http server
!
!
line con 0
line aux 0
line vty 0 4
 password isp2
 login
!
ntp clock-period 17179884
ntp server 10.2.1.1
end
```

4.2 Configuration of the Radius server

```
-------------------------------------------------------------------------------------------------
----------------------
Port-Access Authentication
-------------------------------------------------------------------------------------------------
----------------------
********************** Radius Server (freeradius)*********************

# cat >> /etc/raddb/clients.conf <<isp2.lab
> client 10.2.128.0/24 {
>         secret          = secret
>         shortname       = isp2
> }
>
> client 10.2.5.0/24 {
>         secret          = secret
>         shortname       = isp2_s
> }
> isp2.lab
#

# ex /etc/raddb/users <<isp2.lab
> 1
> i
> bob     Cleartext-Password := "hello"
> .
> wq
> isp2.lab
#

--------radiusd.conf at least must have----------------------
authorize {

     preprocess

     files

     eap

}

authenticate {

     eap

}

--------eap.conf at least must have----------------------
eap {

     md5 {

     }

}

--------Starting radius server in debug mode-------------

# radiusd  -X
```

4.3 Configuration of the switch in the client network

```
************************HP2524 switch********************
HP ProCurve Switch 2524# config
HP ProCurve Switch 2524(config)# aaa port-access authenticator 18
HP ProCurve Switch 2524(config)# show port-access authenticator

 Port Access Authenticator Status

  Port-access authenticator activated [No] : Yes

            Access   Authenticator  Authenticator  Unauth   Auth     Current
  Port Status Control  State         Backend State  VLAN ID  VLAN ID  VLAN ID
  ---- ------ -------- -------------- -------------- -------- -------- --------
  18   Closed Auto     Connecting     Idle           0        0        1

HP ProCurve Switch 2524(config)# show authentication

 Status and Counters - Authentication Information

  Login Attempts : 3                              &nbsp;
```

```
                 | Login      Login          Enable     Enable
  Access Task | Primary    Secondary  Primary    Secondary
  ----------- + ---------- ---------- ---------- ----------
  Console       | Local      None       Local      None
  Telnet        | Local      None       Local      None
  Port-Access | Local
  SSH           | Local      None       Local      None


HP ProCurve Switch 2524(config)# aaa authentication port-access eap-radius
HP ProCurve Switch 2524(config)# show authentication

 Status and Counters - Authentication Information

  Login Attempts : 3

                 | Login      Login         Enable     Enable
  Access Task | Primary    Secondary  Primary    Secondary
  ----------- + ---------- ---------- ---------- ----------
  Console       | Local      None       Local      None
  Telnet        | Local      None       Local      None
  Port-Access | EapRadius
  SSH           | Local      None       Local      None

HP ProCurve Switch 2524(config)# aaa port-access authenticator active
HP ProCurve Switch 2524(config)# show radius

 Status and Counters - General RADIUS Information

  Deadtime(min) : 0
  Timeout(secs) : 5
  Retransmit Attempts : 3
  Global Encryption Key :

               Auth  Acct
  Server IP Addr  Port  Port  Encryption Key
  --------------- ----- ----- --------------------------------

HP ProCurve Switch 2524(config)# radius-server host 10.2.5.20 key secret
HP ProCurve Switch 2524(config)# show radius

 Status and Counters - General RADIUS Information

  Deadtime(min) : 0
  Timeout(secs) : 5
  Retransmit Attempts : 3
  Global Encryption Key :

               Auth  Acct
  Server IP Addr  Port  Port  Encryption Key
  --------------- ----- ----- --------------------------------
  10.2.5.20       1812  1813  secret


HP ProCurve Switch 2524(config)# ping 10.2.5.20
10.2.5.20 is alive, time = 20 ms
HP ProCurve Switch 2524(config)#
```