Technological Educational Institute of Crete

School of Technological Appliances

Department of Informational Technology Engineering

Thesis Report

Unity Network ~ resilient virtual networking platform built for direct host-to-host data exchange with user identification and security properties

Kagiampakis Konstantinos (ID: 2770)

Supervising Professor: Charalampos Manifavas

Evaluation Committee: G. Kornaros, N. Papadakis, Ch. Manifavas

Presentation Date: June 2014

The report's translation in English was made by its author under a loose sense were more emphasis was given to the idea's logical representation rather than a linguistic resemblance to the original text.

Acknowledgments

The present work is awarded to the people:

Who understand and value what I am trying to achieve and who are making a beneficial impact of their own.

To the people who are fighters, who do not give up their weapons or sellout their beliefs when circumstances are getting tight.

To the free and creative spirits as the world without them would not have a breath and motion!

Abstract

Unity network is a virtual networking platform, capable to be deployed in any kind of IP network as a Local Area Network or over the Internet. The network is centered around three pillar key-principles: **Attributing a unique network identity to each connected host**: The ability for a user to have many personal host-devices connected to the same network where each one is identified by a unique IP address. Feature which enhances host-to-host connections and lets the users keep contacts of other friendly hosts in their contact list. **Non-restraining host-to-host connections**: the ability for a user's host to non-restrictively and conveniently exchange data of any type of network service he wishes with any other host. **Secure and private host-to-host connections**: the ability for each host to exchange encrypted information with any other by default and non-encrypted by selection by making use of RSA certificates and public key distribution features in order to provide authentication and confidentiality between the connected nodes. From a technical aspect, the network is built for enhanced resilience as, from the one hand, it is based on a divide and conquer logic demonstrating distributed node roles and decupled network traffic from network logic, features which allow the platform to resist death and dynamically expand to serve many host-systems. From the other, it is based solemnly on software written in Java without the need of any dedicated device or hardware which allows its applications to be hosted under many different devices and Operating Systems. For the clients behind a NAT/firewall, the network offers NAT traversal techniques to let them pass their network limitations and connect to the virtual network without compromising their ability to exchange any kind of data. To conclude, the network’s higher intention is to act as a live and tangible example of a better version of today’s Internet.

Table of contents

Table of graphs and pictures

Introduction

1.1 What is Unity Network? Which is its purpose?

Unity network is a distributed virtual networking platform capable to be deployed and used over the Internet. The network is centered around three pillar key-principles:

- Attributing a unique network identity to each connected host.
- Non-restraining host-to-host connections.
- Secure and private host-to-host connections.

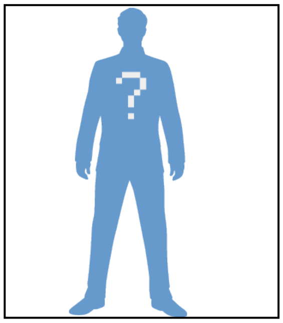
Its main purpose is to solve problems which, despite the rapid technological evolution, the modern Internet still faces by providing to the connected hosts additional features and services centered around communication, security and privacy. The network's higher intention is to act as a live and tangible example of a better version of today's Internet.

1.2 Which are the problems the modern Internet faces?

Unity Network's idea was inspired from the need to provide robust solutions for certain existing problems which, up to date, torment the users and limit the Internet's true potential. The most important problem cases are noted below:

IPv4/NAT

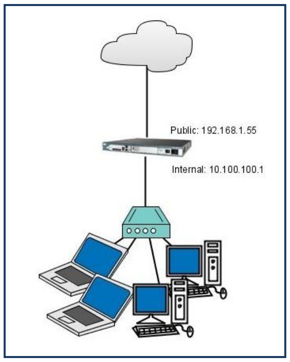
One of the most significant problems the modern Internet faces has to do with the identity it attributes to its connected hosts. The Internet is currently based in the IPv4 technology, where, due to the Internet's rapid spread and the enormous number of the connected devices, it has encountered a lack of available IP addresses to attribute to them. The reason is that the number of available IP addresses is far smaller than the number of the connected devices which makes it impossible for each device to carry a unique IP address, although that was the initial intention for the Internet's hosts. In order for the Internet to work around this setback and retain its functionality, the IP addresses are being treated dynamically **by default** by appointing each one to a host, usually if it is available and thus, resulting for most of the hosts to obtain a **dynamic** IP address as it is called, unless if a host has specifically requested to own a **static** address, if it happens to be a service provider, which in most cases may require additional expenses.



Picture 1 whoami?

The Network Address Translation and its setbacks

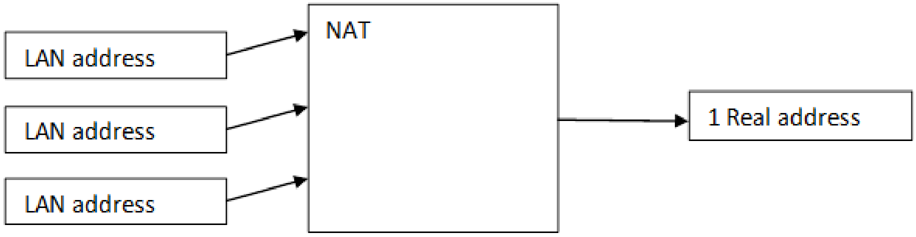
Moreover, to squeeze things even more, the **Network Address Translation, NAT** technology was introduced in order for a group of hosts inside the same Local Area Networks to share the same address and thus, successfully but not efficiently work around the problem to fit into the available number of addresses. In more detail, the NAT protocol, and more specifically the NAT (n to 1) which is mainly used for home appliances, is a solution capable to share one Internet public IP address to a series of inner network hosts. As it was mentioned, it was discovered as a means to work around the problem of making economy towards the small number of IP addresses in order to suffice for the greater number of the connected devices and thus, to identify the hosts online. NAT as a technology has been described by many as a work-around solution [1-3]. The reason is that it treats many local network computers as one over the internet rather than uniquely identifying each one. Some major negative outcomes of this routing are:



Picture 2 Many hosts behind a NAT

- Because the NAT (N to 1) shares one address to many hosts these have to share the available transmission ports like if they were one host. A typical negative example of this case may be that under this manner a LAN may not host two or more web servers as each one would need to occupy the same port, '80', as described in the HTTP application protocol which makes use of the TCP transmission protocol. Therefore, only one host, from the given example, may be allowed to serve properly the HTTP service without workarounds.

- Another setback lies in the available transmission port number range. Typically, in a transmission protocol such as TCP and UDP where two bytes are being used in order to define either the source or destination port, a port's address ranges from $2^{16} – 1 = 65535$. In other words, each host normally owns this port range for its own usage. However, since the NAT treats many computers as one, this range has to be shared among all the inner hosts. The more hosts participate the smaller the port range each one may own.
- NAT examines the transmission header to forward the packets back and forth the inner network where it needs to be preconfigured to analyze specific transmission headers. Due to this fact, most of home routers recognize only two available transmission protocols from all the available spectrum, the TCP and UDP. This workaround practically kills either the rest of the available protocols to use such as RTP and results to the prevention of the research and engineering of new transmission protocols for common usage.



Picture 3 NAT (N to 1) model

A blow in a host's ability to be uniquely identified

The above phenomena had the significant impact for the majority of hosts connected to the Internet to end up with a dynamic IP address which in many cases is sharable with other hosts from the same network. This is the reason that the address cannot serve its purpose and uniquely identify a host, which in its turn, makes the hosts undiscoverable from each other. The successive technology of IPv4, the IPv6, will take a while to be applied as it is not an easy task to do and still, it may not be certain whether it will be applied under the right approach which is to uniquely identify the connected hosts and set aside the NAT as commercial and social motivations have been established like for ex. the static IPs to have an extra cost for a client or pay extra to host a service. Alas, our biggest strength and weakness altogether may be working around problems, however, it is clearly a mistake to learn to live with short fixes. Finally, by a chronological distance we are moving away from the Internet's birth and although its usage is known we tend to forget its intended usage and get settled with its present functionality.

What negative outcomes may result from this phenomenon

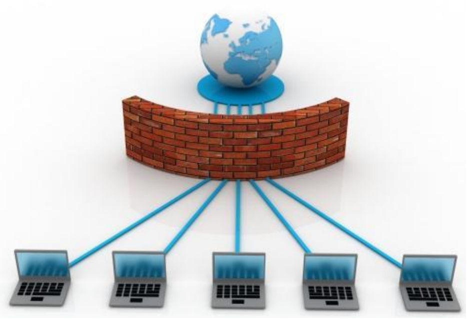
Each user's device is appointed a dynamic and shared with others IP address each time it connects to the Internet, resulting to the direct impact for the other users to not be in a position to locate the user's device. A similar example of this situation may be derived from the phone network. If we had to think about the mobile phones, each user has his own. Each mobile phone has a unique and identifiable number which, in other words, acts as the mobile's name. The reason that we can call directly someone in his mobile is that every time his mobile connects to the mobile network it will get the same name (phone number). This is the reason that we are able to keep contact lists with other friendly mobile numbers. Now, think about the situation where each mobile phone had to collect a different number each time it connected to the mobile network and you get the Internet! Therefore, in the Internet, where hosts may not be uniquely identified, their users have to make use of intermediate services like Facebook, Instagram, etc. in order to be in a position to communicate with others. This phenomenon has a direct impact to the users' communication as they are compelled to join commercial services without any other default alternative offered by the network. A commercial service from the other hand, introduces legal terms which a user has to accept in order to use it. This allows many commercial services to be allowed to mine personal data from their client-users and to commercially exploit those data for their own companies' benefit. Finally, the users are being fragmented into a variety of services as some of them may be members of Facebook, some of them members of Skype where the users from group A may not be allowed to communicate with the users from group B.

New trends are being introduced

Over the Internet's first years, a user would normally use one or no device to connect to it, thus, he would only need one IP address. The relation, therefore, between a user and his device was 1-1. In today's world one user may own multiple devices such as a desktop/laptop, a mobile, a tablet, a smart device where all of those may be connected to the Internet simultaneously, therefore the model is changed from 1-1 towards 1-N where one user may own multiple devices connected to the Internet at once where the latter is in need to cope with the new model. Moreover, one of the latest trends for a user's devices over the Internet, is to have the option to synchronize and distribute data and tasks like common user interfaces, music/video streaming between his devices. This situation encourages the devices to be able to have a unique name in order to properly create direct connectivity towards them and thus, to properly serve such services.

Connectivity restrictions

Another sum of setbacks for the Internet is related towards the **level-of-difficulty** and **opposed-restrictions** it provides towards each connected user regarding its ability to provide network services and to exchange data between others. In more detail, due to the internet's distributed nature, the hosts' location is highly related towards opposed-restrictions where these may vary from ISPs, firewalls, country restrictions and other. An example may be a host who connects to the Internet via 3G may not be allowed to host a web-service on port '80' as opposed to a host who connects from a cable connection. Regarding the level of difficulty, users may be able to know which service they would prefer however they may not be in a position to properly use the tools as software to complete their task. To a typical user, many services are considered a demanding task for more advanced users, they require time and energy to build and in many cases, there is not even the available software to use them. An example of this situation may be a user who would like to host his own file sharing room but he is faced with a list of complex and time-consuming tasks to perform.



Picture 4 A firewall which stands in between the connecting hosts and the Internet

Some examples:

- Regarding the **difficulty of the task**:
    - How easy is for a user host a web server?
    - Is it an everyday task?
    - Does he need a series of clicks or days of work?
- Regarding the **opposed restrictions**:
    - Is the user's ISP blocking him from hosting a web server?
    - Does his ISP block Twitter in his country?
    - Does the ISP prevent him from using the SSH service?
    - Does the NAT prevent the user host a web server as the port is registered to another local server?
    - Are there intermediate firewalls that block the user's web server?

Last but not least, the Internet as a network is distributed both in hardware as in country and provider policies where each host may faces a different set of different opposed-restrictions, on the other side, the user as a person, a different level of difficulty. Both factors lead to **non-uniformity** among its connected hosts.

In this point, it should be questioned whether it is ethical for a network administrator, an ISP, or any other intermediate party, to be able to limit a host's data sharing options over the Internet. If it is considered that there may be many hosts connected from different places, where in each one, different policies are being applied, then the connected nodes in their overall may demonstrate a different behavior and capabilities from each other. Ultimately, the connected devices-members should "reach" the Internet with the same sharing rights as only under this approach the hosts may experience a true and limitless sharing experience.

Lack of privacy

One of Internet's major setbacks is its limited security and privacy it attributes to its connected users. Every day, in the Internet, personal files, passwords and money may be stolen from third parties, while companies and organizations may mine personal data for their own benefit. The root of this problem lies in the Internet's nature as it is established today where the users do not sufficiently make use of encryption processes as they are either not informed about them, their use is complicated or in many cases a user needs to pay something extra to use them like earning a digital certificate. Things get worse as usually a user's choice to not use encryption comes as a combination of the above reasons ex. A user may think that there is no point to pay something extra for a technology he is not either experienced to use or there is no mature and easy to use software. Moreover, the ownership of a key-pair per host is not something necessary but an optional feature instead. To make things even worse, processes as signing digital documents, sending encrypted emails, initiating confidential connection streams and others are significantly difficult to use, they are often solemnly based on terminal applications, and eventually not socially accepted and established. The result of the above conditions is a significant percentage of the Internet's traffic to not be encrypted, and therefore to not be confidential, something which makes the job of MITMs and data miners exceptionally easy! A beneficial approach for the matter would be for each user's host to own a key-pair by **default**. Under this approach, ISPs could act as Public Key Authorities whereas the device manufacturers could equip their devices with friendly to use software for generating and managing keys. By doing so, the produced connections in their overall would eventually create a cloud of encrypted data where it would be technically impossible to intercept each one of those streams. In other words, the more the users who make use of encryption methods, the harder is for the miners to intercept personal data in their total.

Picture 5 A third party who intercepts a data exchange between two hosts.

If we had to go into a bit of technical detail, we would observe that the Internet is a network which forwards data in the form of packets from the one node to the other until each packet reaches its destination. Since the network's routing nodes accept packets to forward them, where the latter ones are scarcely encrypted, the process of mining data is facilitated. Furthermore, the process of data mining, with the combination of commercial services provided from Facebook, Twitter and other services who reroute personal, semi-personal or public data, create a fertile ground for this process. Due to this background, practices directly related with network pattern recognition and data mining have become exceptionally popular both in the enterprise as in science. Today 'data mining' is considered a scientific domain, where ethical practices should be systematically considered as it should be reminded that being able to mine personal user data is in a direct breach with the established modern world's civil-rights.

Conclusion

To conclude, despite the demonstrated rapid evolution of the past years, Internet still faces significant lacks and problems not only in the convenience of communication for its connected users but also in matters of safety and privacy for them. Such problems degrade its functionality and end it semi-capable to represent its basic purpose, which is to let the users communicate between them in an easy, non-restrictive, uniform, secure, private and non-commercialized manner on its basis. Finally, instead of the core problems to be solved we tend to build onto them by introducing new shortcuts and workarounds. Under this approach it should be reconsidered what kind of evolution we would like for it and whether should we actively solve the introduced setbacks to make the best out of the present situation or to devise workaround solutions resulting in a future's crisis.

1.3 How Unity Network encounters these problems

Unity Network, as it was introduced in section 1.1, has the purpose to counter all the formerly posed problems by offering enhanced features to its connected users. Where, as noted, it follows three pillar key-principles which are described in detail ahead.

Attributing a unique network identity to each connected host

The network appoints a different and **unique IP address**, not towards each user, but to each one of his devices, as a user in Unity may own multiple devices on the network. Each device collects the same appointed IP address every time it connects to the network regardless its location from where it is connected. The IP addresses are owned by the hosts and do not change, or in other words, they are **static** and **unique** for each host.

Non-restraining host-to-host connections

The network provides the option for each user's device to be allowed to share any kind of service without the consent of any other third party, **be-a-full-host**. In network terms, each device may act both as a client or as a server in any network service. Since each host can use the full spectrum of services, **uniformity** is established while workaround protocols like UPnP IGD which are not secure are not needed anymore.

Secure and private host-to-host connections

The network provides to each users' host an RSA keypair by default and a contact list for other hosts and their public keys. Under this approach, each user's host upon its login to the network, is authenticated by a two-factor combination of its private key and its user's credentials. In addition, a host is able to use its keys for any task inside the network as to be validated by any other host member or to start a secure connection with them.

Each host is appointed its personal integrated firewall, its panel of services and its network monitor. To counter the problems mentioned around the **level-of-difficulty** notion, the connected hosts follow a model of an approachable interaction towards the network by introducing a pyramid layered system that moves from set of abstractive operations to a more detailed set of actions for advanced users. The software easily allows a user to select with which group of hosts is allowed to exchange information and in which way, as there are three groups of hosts a **whitelist**, a **blacklist** and a **neutral host list**. Moreover, from the panel of services, it allows to easily start and stop ready to use services such as chat and file transfer while giving the user the tools to go deeper if he desires into defining more complex firewall policies and host other kinds of services. Due to these features, users of Unity Network may experience a truly secure and private communication towards each other.

1.4 How can Unity Network be described as a technology in brief?

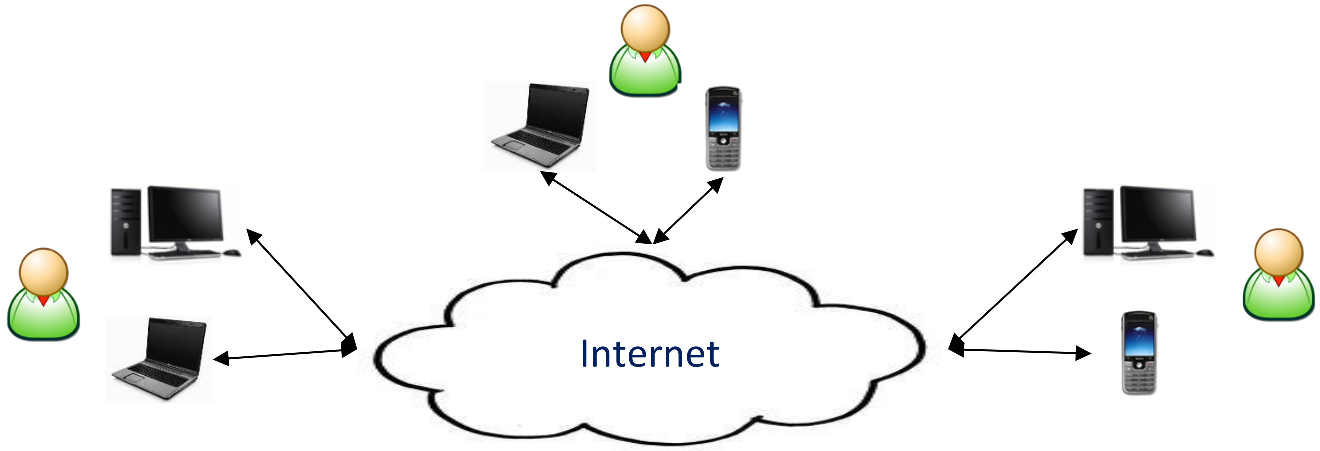
In general, Unity Network may be defined to be a virtual network as it carries many characteristics from the **Virtual Private Network, VPN** family. Its first and major difference compared to other VPNs is that not many of them make use of a distributed behavior to route their traffic as the norm is for a centralized server to exist to serve such a task. Unity, however, is built on a distributed manner by making use of three different node types which are explained ahead under section 1.6, were each one has a different role in the network.

Unity's second and very significant feature compared to VPNs is that is does not make use of a standardized VPN traffic protocol such as the **L2TP** and **PPTP** protocols do, as their network data streams, from the analysis ahead under **section 3**, are being recognized and policed quite easily over the Internet and in some cases, semi-private data may be exposed as well. In contrast, it uses UDP datagrams with an encrypted payload which may be harder to be monitored or policed.

Moreover, another of Unity's architectural features is that the virtual network uses NAT Traversing techniques in order to let host-clients behind a strong network policing as firewalls in a work environment or Internet Service Providers and Intermediates, to be able to connect to the network and be allowed the full range of features the network has to offer.

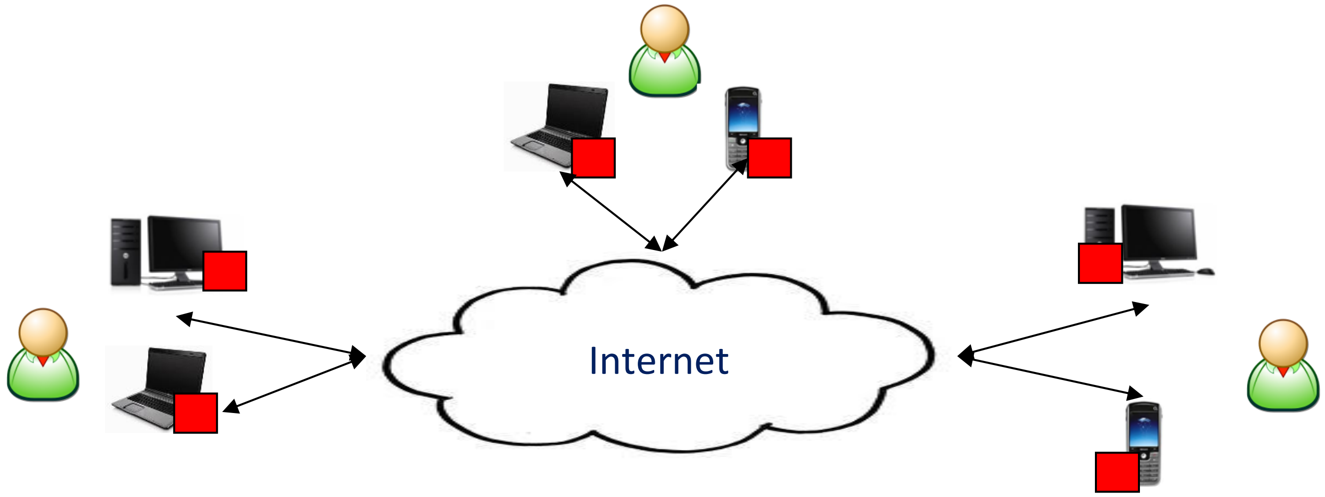
1.5 The network's abstract feeling

In the following graph, many users who own network devices such as laptops, mobiles, smart devices where their intention is to directly communicate towards each other, are being observed and limited to mainly act only as clients. In the beginning, their devices are connected with the Internet where its nearly impossible for them to directly communicate as that would mean to exchange IP addresses, work around the network firewalls and routers in order to forward their hosted services.



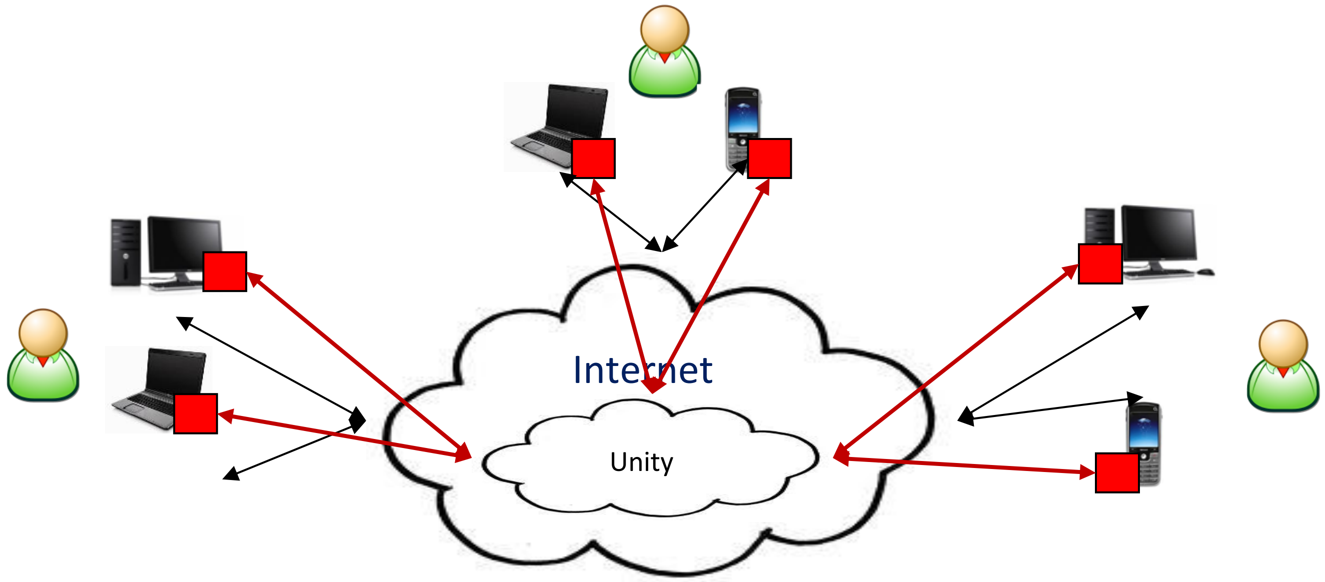
Picture 6 network feeling 1

Now, each host owned by its respective user executes the RedNode application, provides his credentials and logins. After the hosts have been authenticated to join the network by validating their private key and the given user credentials, they are allowed and connected to Unity Network.



Picture 7 network feeling 2

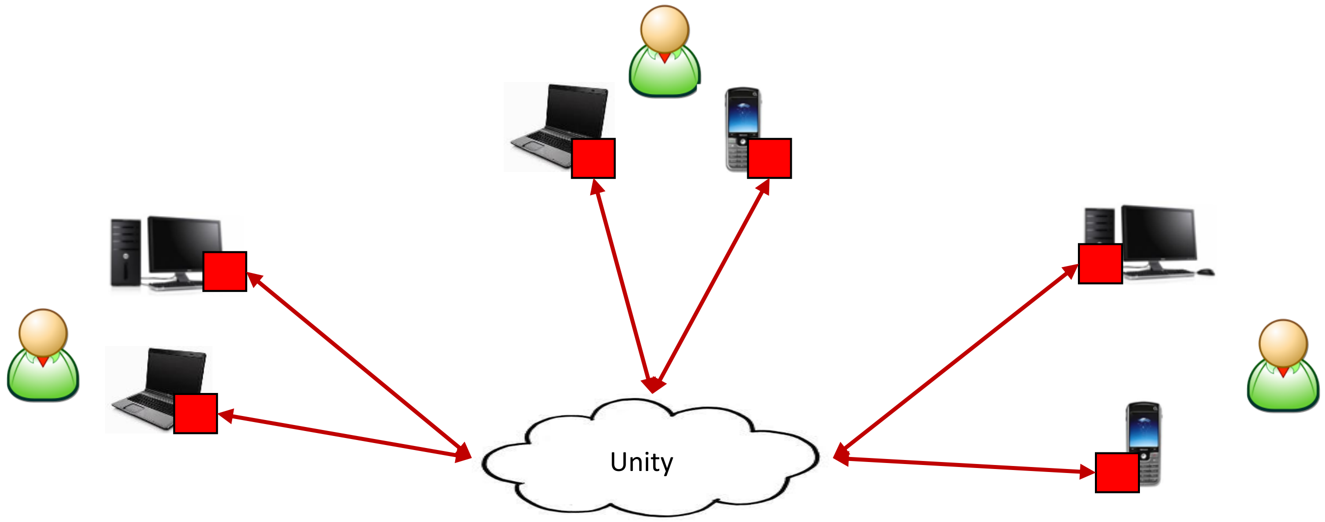
After joining the network each host is **a-full-host** and is allowed to share any kind of service with each other. Moreover, each user may have a contact list with other friendly hosts as their address does not change. If the hosts-clients log-out and then log back in they will get the same address.



Picture 8 network feeling 3 – The black arrow-lines signify the physical connections although the red ones the virtual.

Some direct host-to-host communication scenarios may be the following:

- Bob may directly send a file from his laptop to David's Laptop.
- Steve may video-call Jenny from his computer to her mobile phone.
- May leaves a message from her laptop to her home's noticeboard.
- Bob streams a song form his mobile to his computer
- Jenny sends a file from her laptop to her mobile phone.
- Bill voice-calls Dave from his mobile to Dave's mobile.



Picture 9 network feeling 4 - The virtual network which is now formed

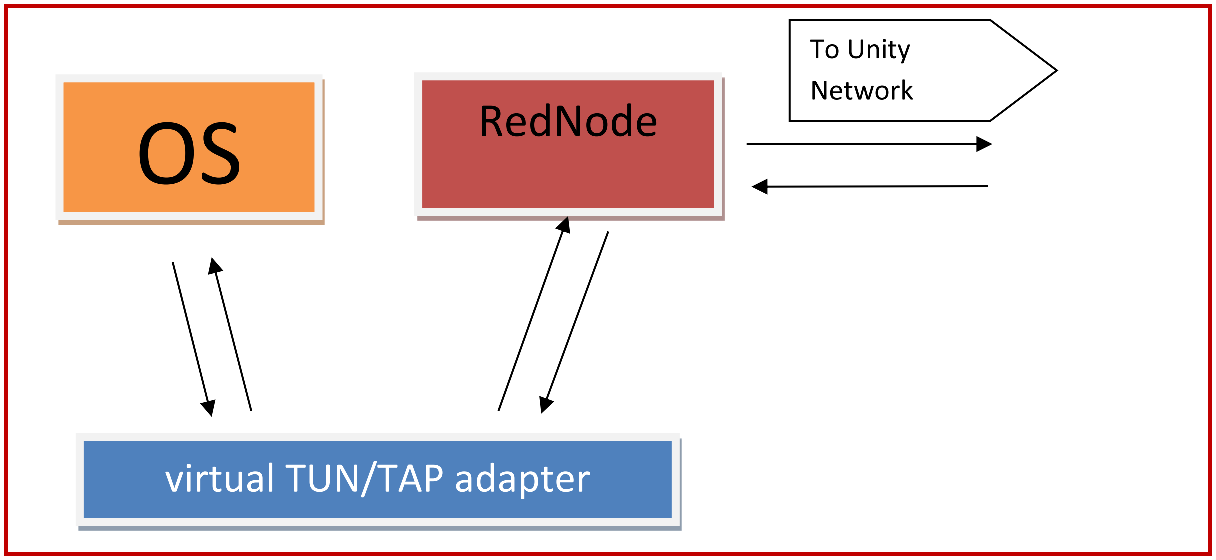
1.6 The platform's architecture

Unity's operational purpose as a platform is to provide a large capacity's **virtual networking space** to its connected hosts. In contrast to classic VPN rooms where, as mentioned, they usually follow a centralized approach with a main server to both route the traffic and make administrative decisions, in Unity a far greater number of hosts can connect as the network makes use of a distributed manner where the **virtual networking space** may be supported by a more than one routing node. Due to this approach the network in its overall demonstrates resistance to death as there are proper routines and co-ordination messages for when a routing server is killed, the network to take appropriate action and move the network's load to the other routing nodes instead. Its operation is as independent as possible from hardware and from a host's operating system. This is because the platform is entirely build on software written in Java which allows it to run under a variety of operating systems. This feature makes the platform more **logical** and **object-oriented** since it is occupied only with the higher logic levels. Finally, the platform functions as a complete solution where it offers to its users, ready processes as: register, manage hosts, generate a keypairs for each host and share public keys. The platform is composed by three node types:

- **RedNode** or **RN**: The client's application or, in general, the node who runs the application. As an application, it allows a host device to be connected on the virtual network. The connected host apart from **RN** may otherwise be known as a **host-client** as it is a network host that is a client towards the virtual network.
- **BlueNode** or **BN**: The application's name or, in general, the node who hosts the application, which is responsible to route the produced traffic from other RNs or forward traffic towards other BNs.
- **Tracker**: The node which is responsible for the whole network's logical behavior. As a node it does not carry or route network traffic, however, its task is to keep track of which RN is connected to which BN, distribute the load evenly and authenticate connected BNs and RNs to the network. In its turn, a tracker may be hooked to a database or use a local solution as SQLite. The database, in its turn, may be connected to a web interface to let the platform's users create accounts and modify their content.

The client's host-device & the RedNode client

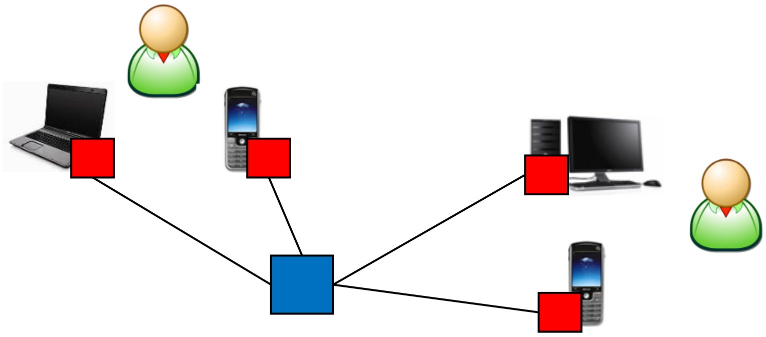
A client's host has two features which allow it to connect to the network. The first is, a virtual TUN/TAP network adapter and the second, a RedNode application. The virtual TUN/TAP adapter has become significantly popular in network engineering as despite that it is a software solution, it is being treated by the OS as a hardware network adapter with a full set of features one may have as an IP address while being fully capable to route network traffic back and forth the OS to an application. Then, the adapter allows the controlling application to read/write byte arrays from/to the medium which are being treated as network packets. The RedNode application, which makes use of the TUN/TAP adapter, after it is being authenticated by the network, starts an instance of it with the acquired IP address and routes network data between the OS and Unity Network. On its turn, the RedNode application connects as a client towards a defined BlueNode from the network and forwards the packets to be sent, encapsulated under a UDP stream to the target BlueNode. Moreover, a RedNode is able to co-ordinate the virtual network adapter by sending network packets such as ARPS and DHCP offers in order to effectively manage the adapter, traffic and the other hosts.



Picture 10 A client's host system

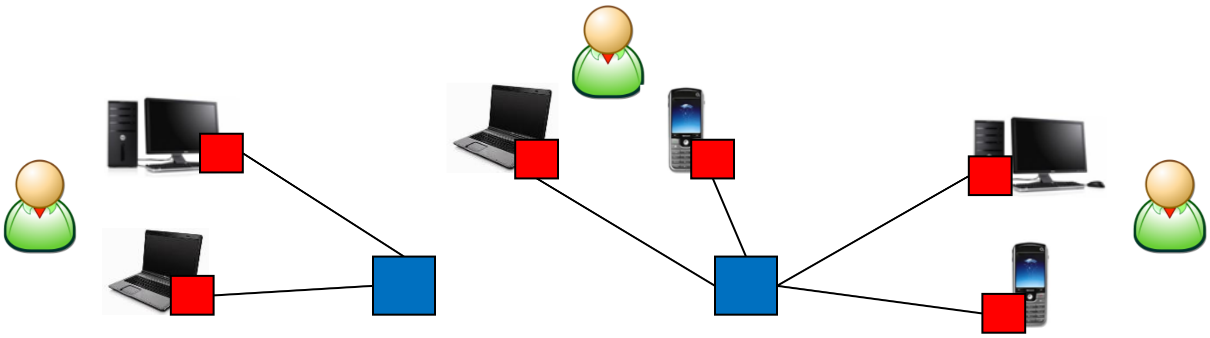
A BlueNode

In its simplest use, a BN acts as a basic VPN server where it may allow RNs from a near area regarding latency (with a small ping towards the BN) to connect to it and become members of the virtual network. Initially, one BN authenticates a connecting RN by its login. Then, it may receive UDP datagrams from it which in their turn, may contain among other synchronization messages, the encapsulated IPv4 packets. The BN examines each encapsulated packet's header and identifies the destination IP address, where, by making use of an integrated router, forwards the packet to the RN associated with the given address, under the condition that it is online. The target RN, upon receiving the packet, loads it to its virtual network adapter to make it an operational network packet.

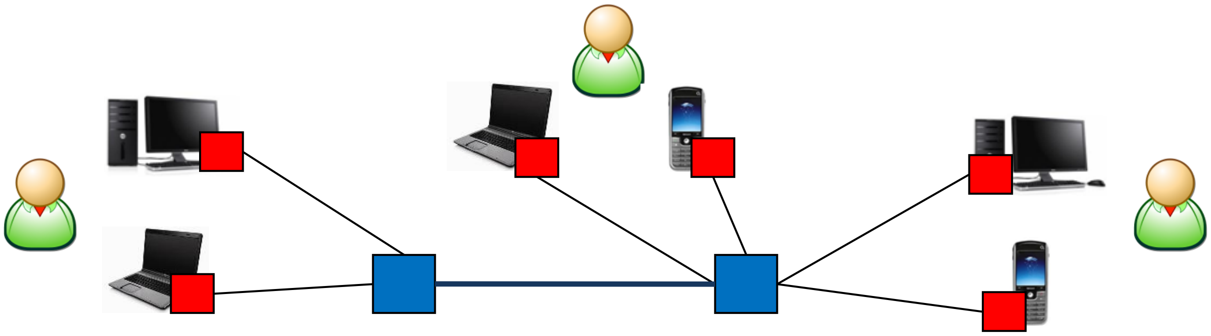


Picture 11 Unity Network architecture 1 – One BN, No Tracker

In its next architecture, the network varies from a classic VPN network in the sense that this time a BN, on its turn, may keep a friendly BN list. As compared to the previous example where a BN was discarding a virtual packet if there was no entry for the target RN's destination IP, in this architecture, it requests from the associated BN whether the given target RN is their client. If another BN is to be found to keep the requested RN, the BNs dynamically create a data stream to let the remote RNs exchange data.



Picture 12 two BNs where each one serves different host-clients



Picture 13 two BNs which have been paired so that the distributed RNs are able to communicate

BNs run on hosts who have selected to serve the network by sharing some of their Internet bandwidth. In contrast to RNs that are solemnly client applications and do not demand network privileges, the BNs require to be allowed to forward to a specific port range, therefore, for their host system to have admin rights. The BNs application is written in Java with the option to either present a GUI or work in terminal. It should be noted that hosting a BN does not make the host who runs it a member of the network as the application is dedicated for rerouting traffic among RNs, therefore no TUN/TAP adapter is needed.

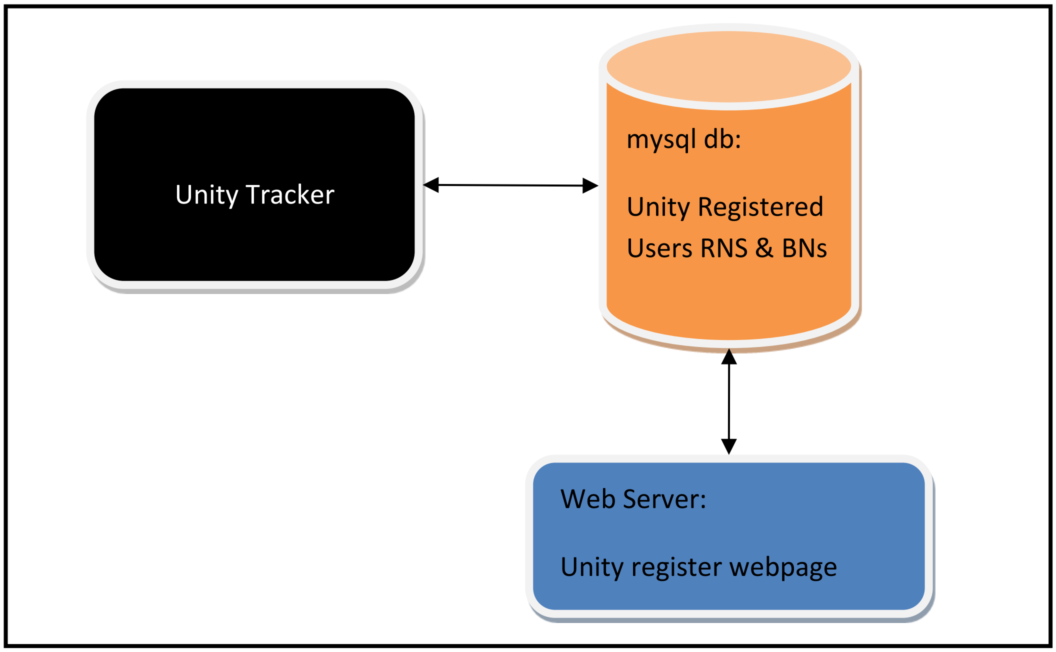
Unity Tracker & Registry & Web Interface

Since BNs may start or stop in an unpredicted time and due to the fact that there may be many of them, all with different clients who may similarly login and logout anytime, the network makes use of a central Tracker who is responsible to co-ordinate the platform and let the multiple BNs be able to identify one another.

- A tracker is aware of the destination IP address of each connected RN on the virtual network but not aware of their real IP address
- A tracker is aware of all the connected BNs and their real IP addresses
- A tracker is aware of which RN is connected to which BN
- A tracker acts as a public key authority for both the BNs and RNs while it uses a personal key-pair to be identified from all of them

- No virtual network traffic is being routed through the tracker
- A tracker cannot choose into which BN, an RN may be connected to but it may provide the best option instead
- A tracker may respond to specific messages regarding the sender's type whether it's a BN or an RN.

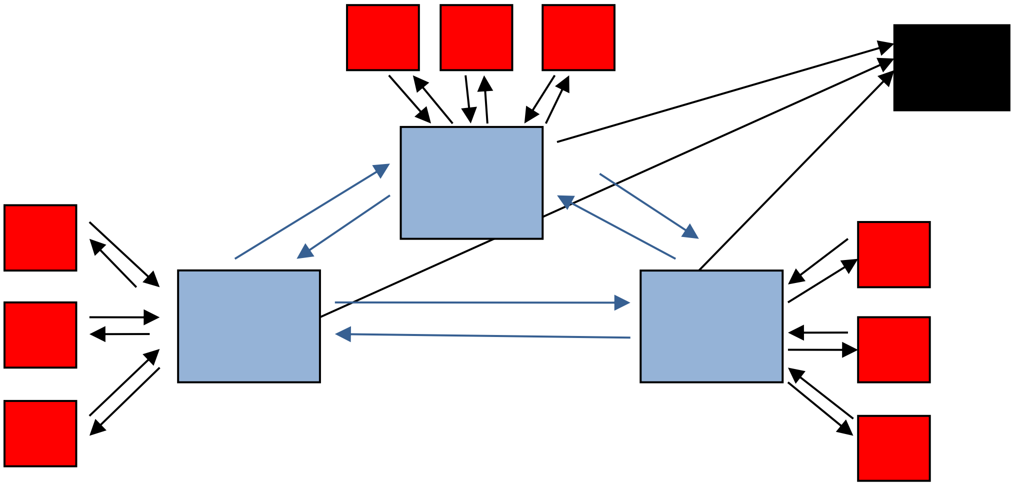
A tracker's host includes the Tracker's application in Java. The tracker's application makes use of an integrated database scheme. However, for a larger number of clients and BNs, the tracker's app may be linked to an external database instead. Moreover, the host may make use of a web interface to let users manage their accounts via the web.



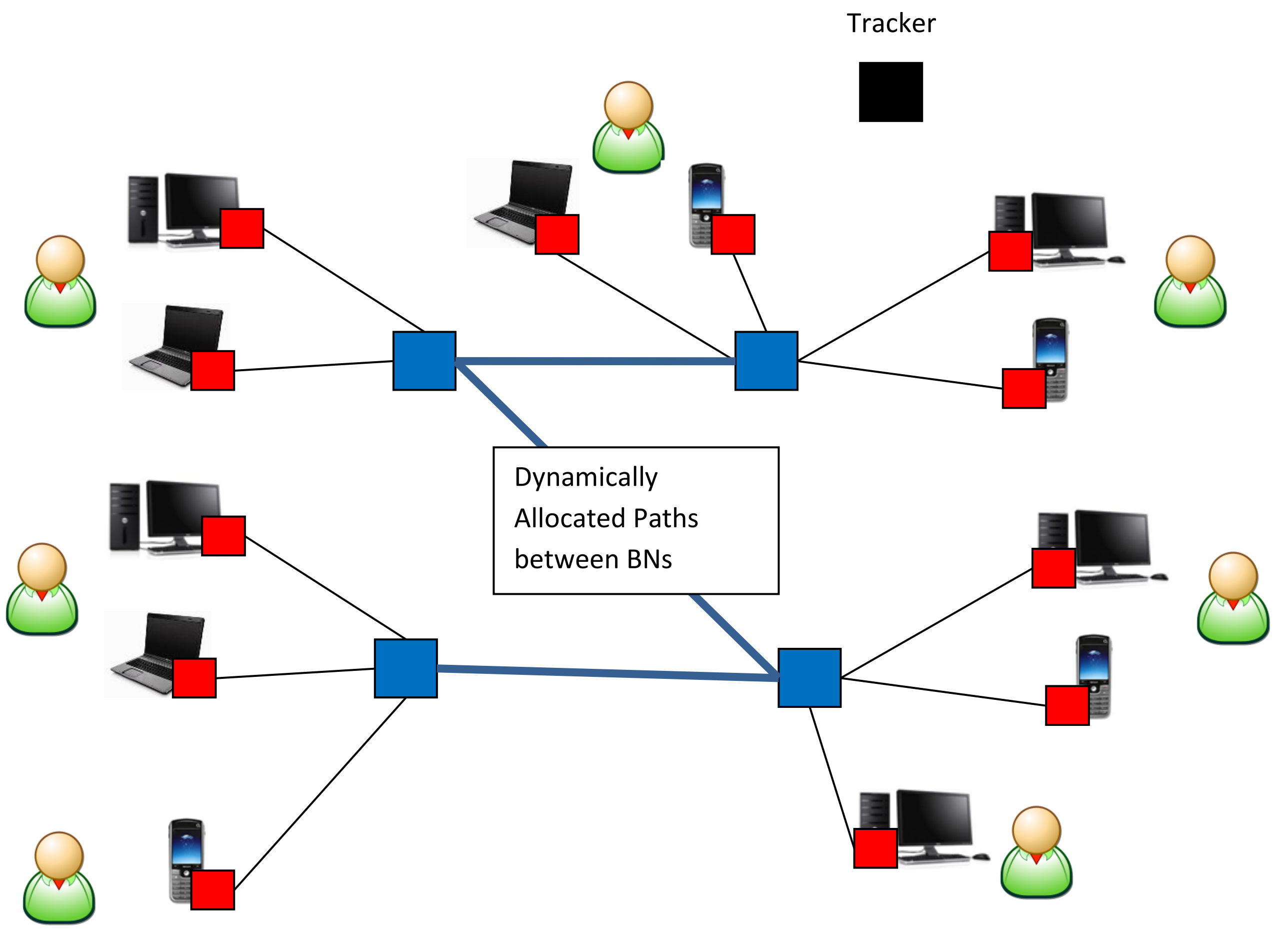
Picture 14 Tracker Node

1.7 The network's complete architecture

Ultimately, the virtual network is demonstrated in its full form in the following model.



Picture 15 The virtual network's overview



Picture 16 A random instance of the network's complete architecture

- The RNs connect to close BNs
- The tracker keeps log of the connections
- Through the tracker, BNs can locate one another
- Since BNs can be inter-located, all RNs are able to communicate with each other despite from which BN they are connected from.

Role distribution

In Unity Network, a privilege hierarchy is established so that no node has the total control or the total intelligence over the virtual network, but instead, each node has the necessary insight to fulfil its task.

- The tracker coordinated but does not route network traffic and does not now the RNs physical address
- BNs route traffic but they cannot control the platform or other BNs
- RNs have the total control of themselves but not anyone else. The are allowed to connect and disconnect wherever they fit to. In addition, they may select a BN to connect to.

2. The usage of Unity Network

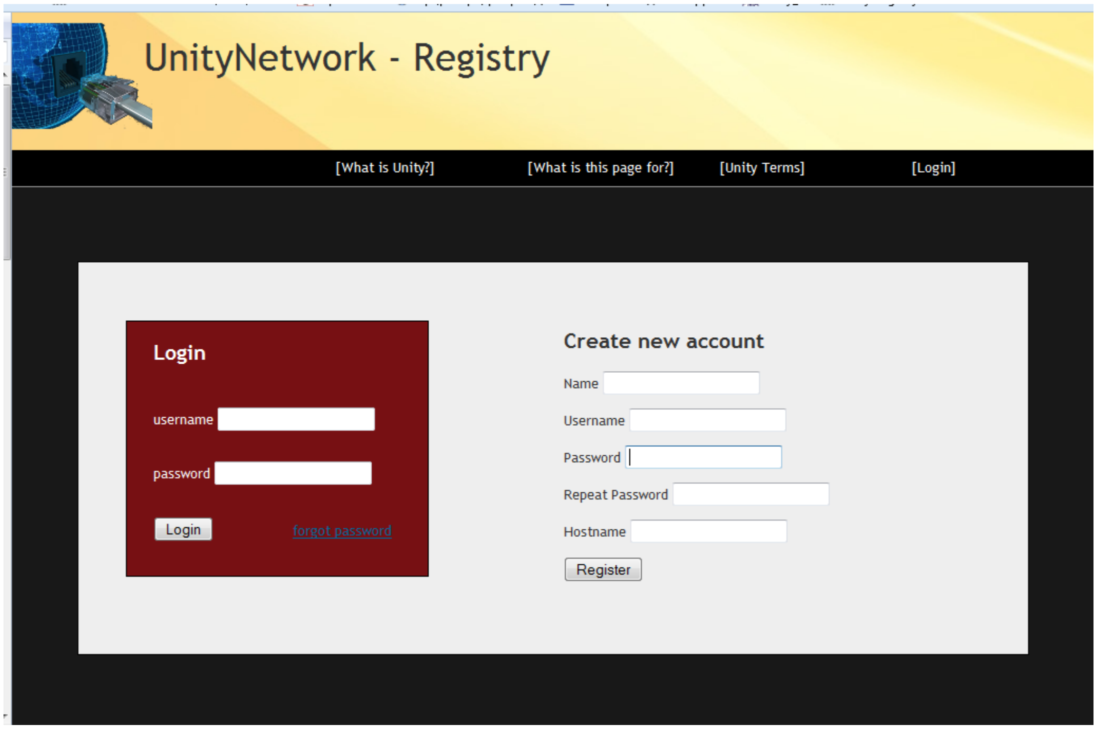
In this chapter, all the possible operations as the applications' GUI (Graphical User Interface) for all three node types are being introduced. Moreover, all the necessary processes, from a user to register and connect to the network to the procedures an admin should perform to set up the platform or a part of it are being explained.

2.1 The User

The user's software is made under the approach to be simple and effective to use so that a user may not be faced with complex installation and configuration processes but rather be focused on managing his shared content.

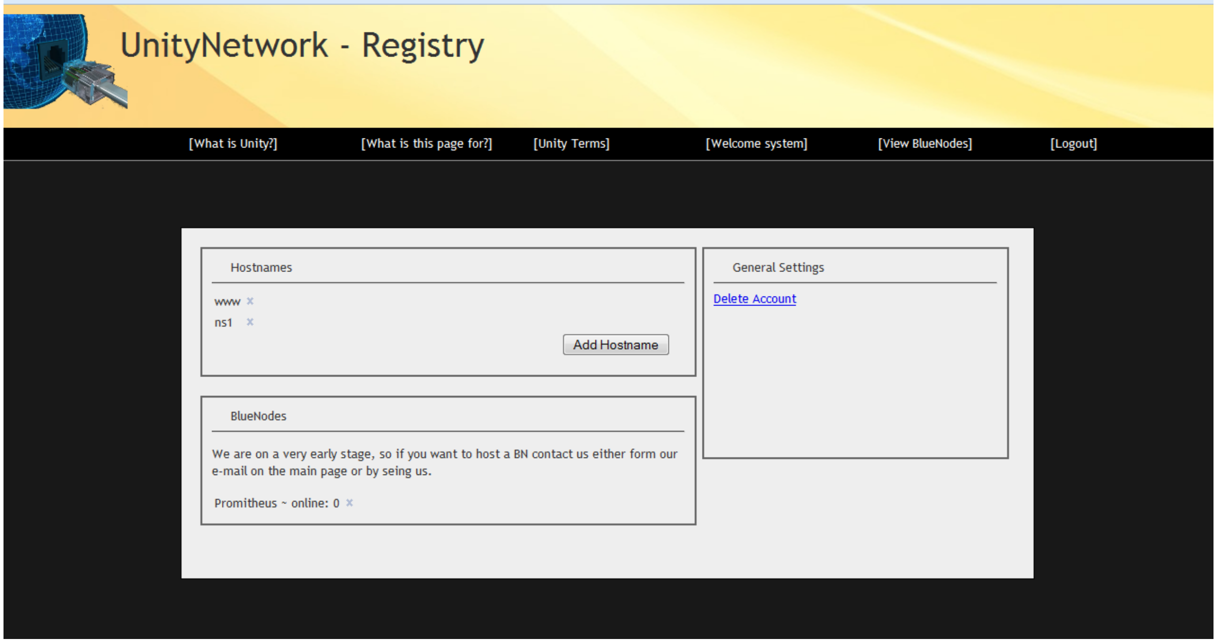
Registering to the platform

Upon revising section 1.6, a tracker may keep a web interface which users may be able to use via a web browser. A user may access the web UI by simply visiting the main web-page of the tracker's registry. Then, as in every other web-service he may create a web account and login/logout from his control panel.



Picture 16 Registry page

After the registration process, he will be asked to define his first hostname which matches the system which he may wish to connect to the network. Then, he may add or remove further hostnames from the control panel. A user's page keeps additional options as account deletion or, optionally, hosting BNs should the user decide to share some of his traffic with the platform.



www ×
ns1 ×

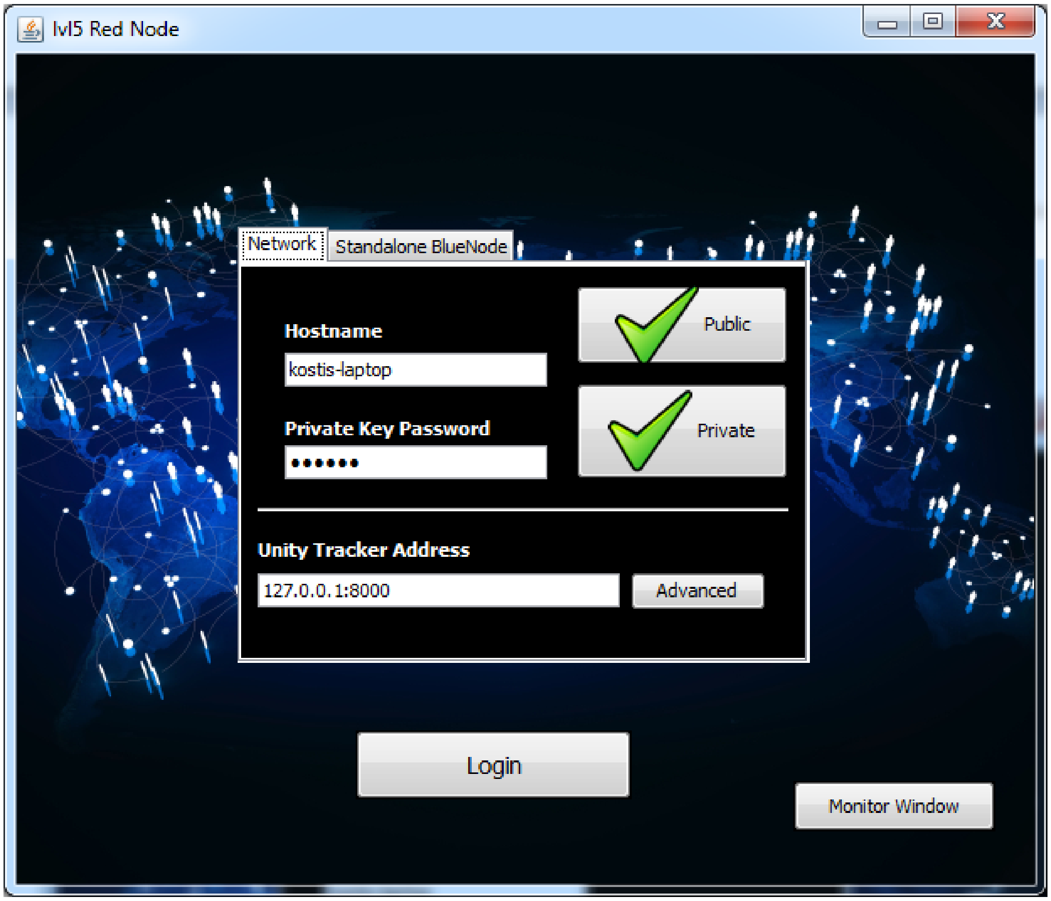
We are on a very early stage, so if you want to host a BN contact us either form our e-mail on the main page or by seing us.

Promitheus ~ online: 0 ×

Picture 17 registry page settings

The Red Node application

This is the RN application where it is developed in Java and a user may run an instance of it in each of his owned hosts he wishes to connect to the network. Initially, a user has two options:



Hostname: kostis-laptop

Private Key Password: ••••••

Unity Tracker Address: 127.0.0.1:8000

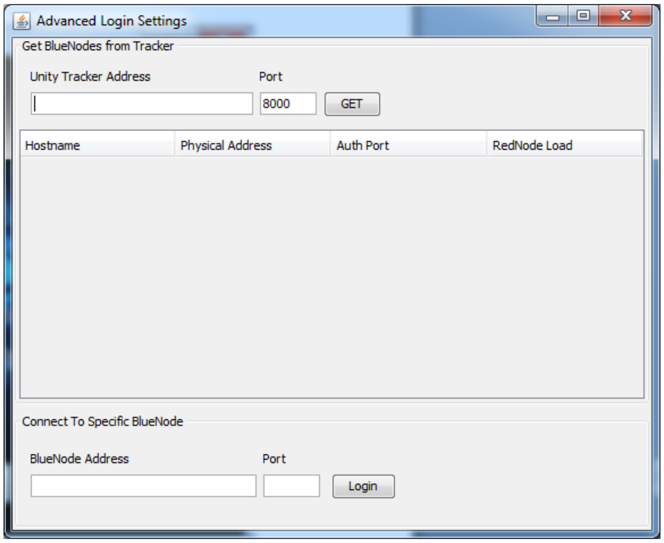
Picture 18 A RedNode’s welcome window

The first option is to let a host connect to a full instance of Unity Network while the second is to let the host connect to a standalone Blue Node which acts as a centralized VPN server without any traffic forwarding to other BNs.

Unity Network option

Should the user wish to connect his host to an instance of Unity Network, he needs to own a public/private keypair for the given host, which on its turn, should be registered with the target network. In addition, he needs provide the tracker's address, his username, password and the host's name.

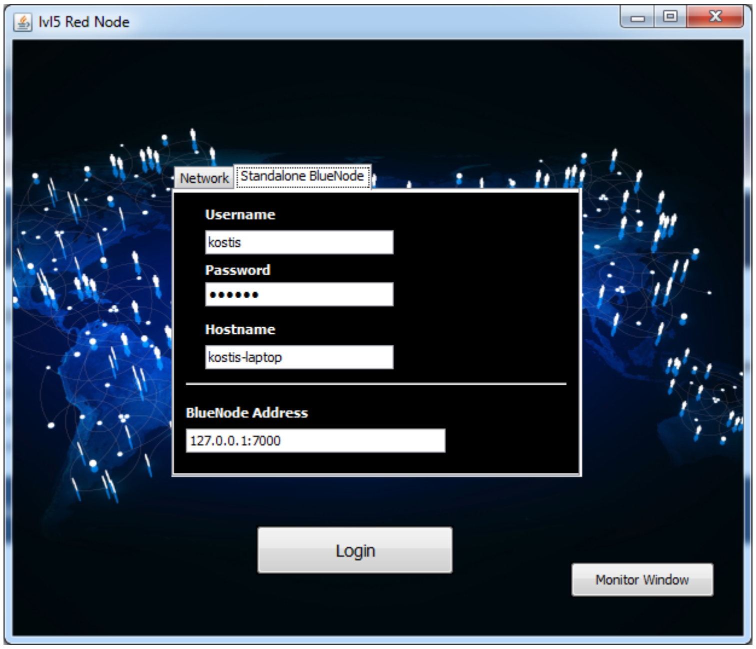
Then, he may either click login where the network's tracker will automatically respond with the address of the closest BN or manually he may open the advanced window to select a specific BN to connect to.



Picture 19 Advanced Window

Standalone Blue Node option

The second option is for the user's host to be connected in a standalone BN which acts as a centralized VPN server without any traffic forwarding towards other BNs. In this case, a user should provide the BN's network address and he may optionally provide a username, a password and a hostname or leave the fields blank to be assigned a random hostname when connected to a BN without authentication.



Username: kostis

Password: ••••••

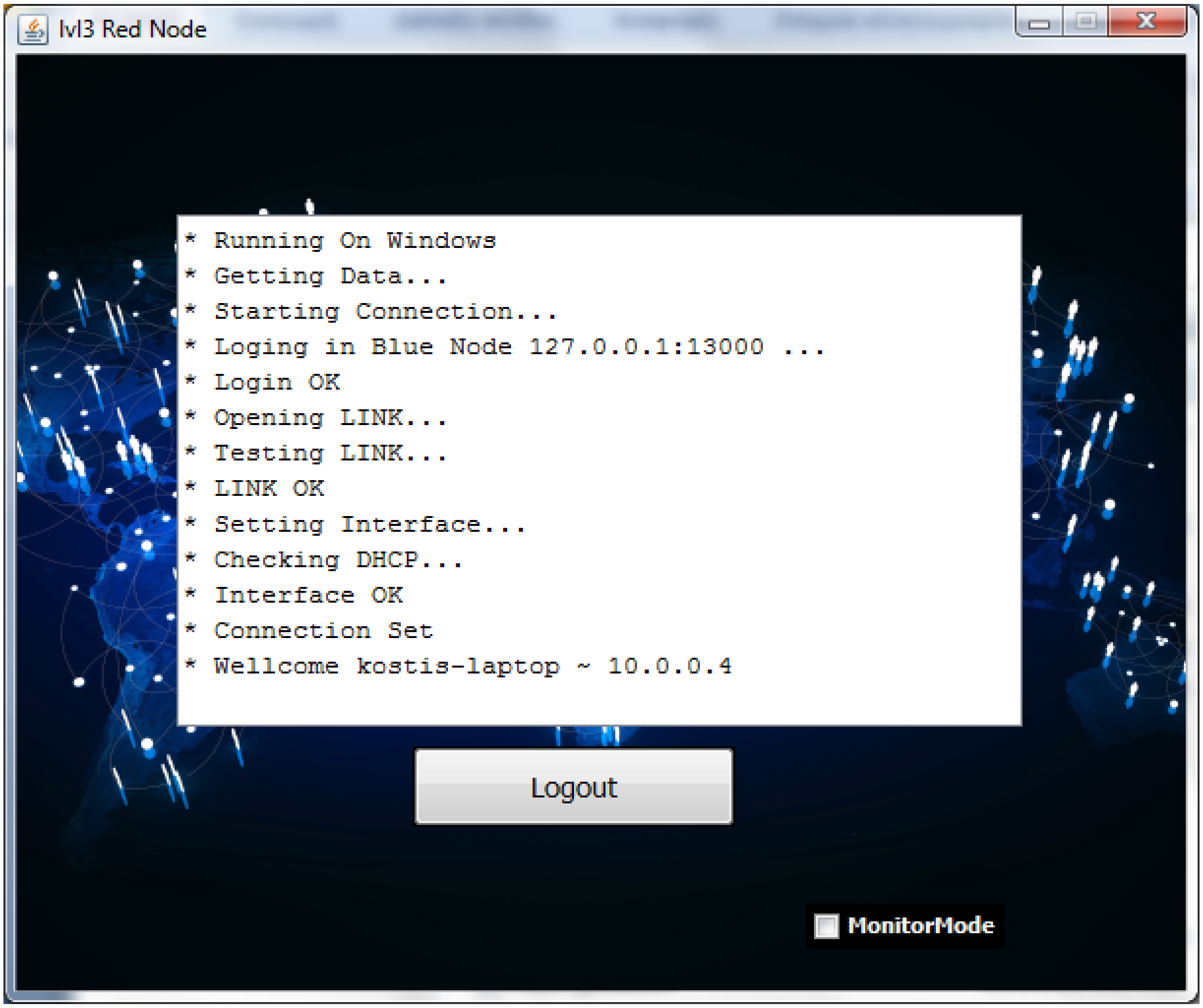
Hostname: kostis-laptop

BlueNode Address: 127.0.0.1:7000

Picture 20 Figure 26 RedNode main window – Standalone BlueNode option

Connecting

In either option, as soon as the user clicks login, the connection process may begin towards the selected or provided BN. The user may view the connection process from the information messages shown in the main window. In a successful connection, the window will present the host's virtual address and the button will be changed to a logout option.



```
* Running On Windows
* Getting Data...
* Starting Connection...
* Loging in Blue Node 127.0.0.1:13000 ...
* Login OK
* Opening LINK...
* Testing LINK...
* LINK OK
* Setting Interface...
* Checking DHCP...
* Interface OK
* Connection Set
* Wellcome kostis-laptop ~ 10.0.0.4
```

Picture 21 RedNode - logged in

After the login process is finished, he may open any kind of server, client or peer network application he wishes.

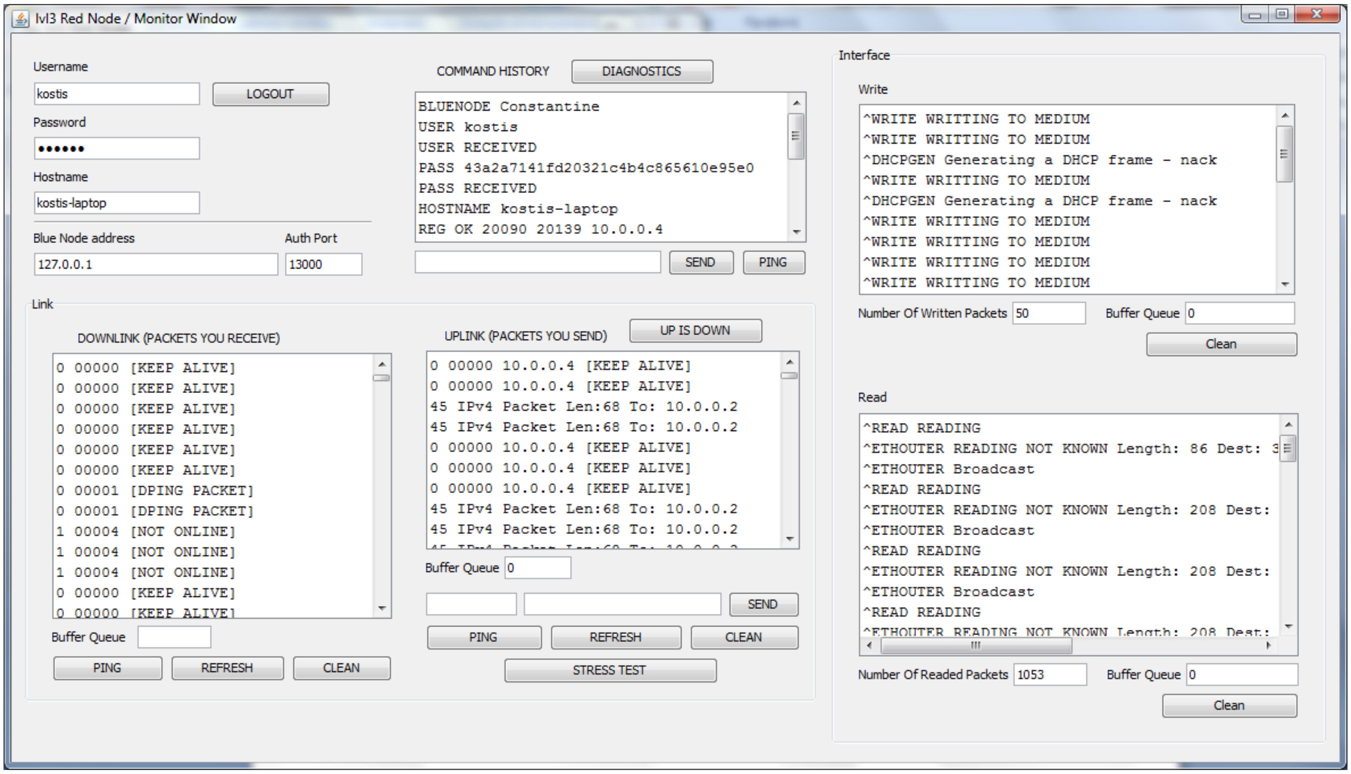
Some examples of network applications are:

- ftp server or client
- Http server or client
- Torrents, p2p networks
- Game servers
- Chat servers like IRC
- Real VoIP
- Any other network service

A user is not in need to forward any transmission ports from his home's or work's router and firewall and neither to share his host's ports with any other host over the LAN. Moreover, he may define his personal firewall, integrated in the application, to share what he desires and with whom in Unity Network.

Connection Debugging

A user may open the monitor window by clicking the button on the bottom right to view the network's traffic. From the monitor window he may collect live information and make some debugg tests.



Username: kostis

Password: ••••••

Hostname: kostis-laptop

Blue Node address: 127.0.0.1

Auth Port: 13000

```
BLUENODE Constantine
USER kostis
USER RECEIVED
PASS 43a2a7141fd20321c4b4c865610e95e0
PASS RECEIVED
HOSTNAME kostis-laptop
REG OK 20090 20139 10.0.0.4
```

```
0 00000 [KEEP ALIVE]
0 00000 [KEEP ALIVE]
0 00000 [KEEP ALIVE]
0 00000 [KEEP ALIVE]
0 00000 [KEEP ALIVE]
0 00000 [KEEP ALIVE]
0 00001 [DPING PACKET]
0 00001 [DPING PACKET]
1 00004 [NOT ONLINE]
1 00004 [NOT ONLINE]
1 00004 [NOT ONLINE]
0 00000 [KEEP ALIVE]
0 00000 [KEEP ALIVE]
```

Buffer Queue:

```
0 00000 10.0.0.4 [KEEP ALIVE]
0 00000 10.0.0.4 [KEEP ALIVE]
45 IPv4 Packet Len:68 To: 10.0.0.2
45 IPv4 Packet Len:68 To: 10.0.0.2
0 00000 10.0.0.4 [KEEP ALIVE]
0 00000 10.0.0.4 [KEEP ALIVE]
0 00000 10.0.0.4 [KEEP ALIVE]
45 IPv4 Packet Len:68 To: 10.0.0.2
45 IPv4 Packet Len:68 To: 10.0.0.2
45 IPv4 Packet Len:68 To: 10.0.0.2
```

Buffer Queue: 0

```
^WRITE WRITTING TO MEDIUM
^WRITE WRITTING TO MEDIUM
^DHCPGEN Generating a DHCP frame - nack
^WRITE WRITTING TO MEDIUM
^DHCPGEN Generating a DHCP frame - nack
^WRITE WRITTING TO MEDIUM
^WRITE WRITTING TO MEDIUM
^WRITE WRITTING TO MEDIUM
^WRITE WRITTING TO MEDIUM
```

Number Of Written Packets: 50 Buffer Queue: 0

```
^READ READING
^ETHOUTER READING NOT KNOWN Length: 86 Dest: 3
^ETHOUTER Broadcast
^READ READING
^ETHOUTER READING NOT KNOWN Length: 208 Dest:
^ETHOUTER Broadcast
^READ READING
^ETHOUTER READING NOT KNOWN Length: 208 Dest:
^ETHOUTER Broadcast
^READ READING
^ETHOUTER READING NOT KNOWN Length: 208 Dest:
```

Number Of Readed Packets: 1053 Buffer Queue: 0

Picture 22 RedNode Monitor View

Red Node Application Installation and Running

Since the application is continuously updated the application's installation and running process are explained in detail under the application's page on GitHub, accessible from the following url:

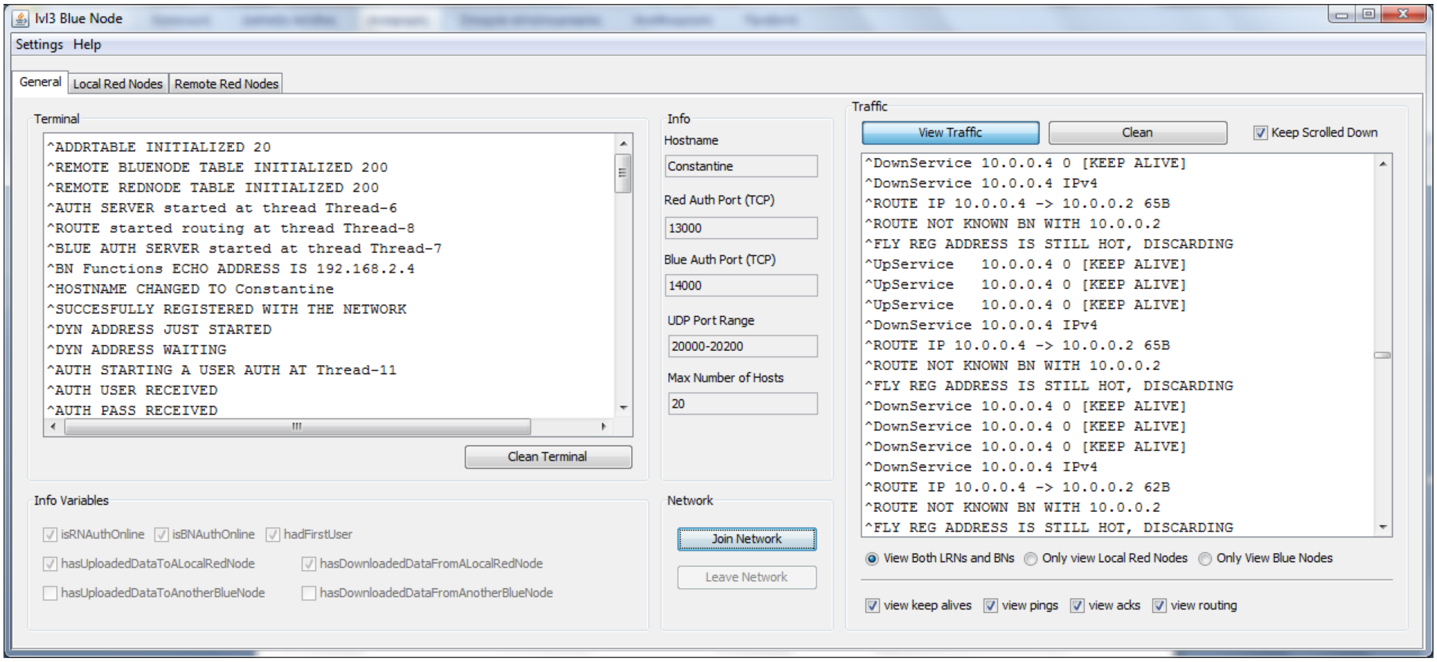
https://github.com/kostiskag/unitynetwork-rednode

2.2 The Blue Node application

A Blue Node host is a system which may run the Blue Node application. This system may either choose to host a standalone BN which is a centralized mode for a VPN server or to run the BN registered with a Unity Network instance to expand the given network. The more BNs are added to a network instance, the more effectively it may route and distribute the network load. A BN may be run either as a GUI window or as a terminal application while the level of verbose information may be controlled to better suit the admin. The application does not require any terminal arguments as it was considered a good practice to be defined solemnly by the configuration files under its directory. Finally, for a host to run an instance of a BN, the host should be allowed to forward its port range on the Internet and set his router or firewall accordingly.

2.2.3 A Blue Node's GUI

This is a BN's main window. It is separated under three tabs where in each one there is categorized information about the node's operational data. The first tab depicts general info as the BN's name, the BN's state, terminal and traffic logs and other. The second tab presents all the connected RNs directly with the BN under a table where each entry represents one's data. The last tab is related towards external connections. It shows two tables, the table on the right shows the connected BNs towards the given BN while in the left window shows the Remotely Associated RNs - RRNs which are clients to the connected BNs.



```
^ADDRTABLE INITIALIZED 20
^REMOTE BLUENODE TABLE INITIALIZED 200
^REMOTE REDNODE TABLE INITIALIZED 200
^AUTH SERVER started at thread Thread-6
^ROUTE started routing at thread Thread-8
^BLUE AUTH SERVER started at thread Thread-7
^BN Functions ECHO ADDRESS IS 192.168.2.4
^HOSTNAME CHANGED TO Constantine
^SUCCESFULLY REGISTERED WITH THE NETWORK
^DYN ADDRESS JUST STARTED
^DYN ADDRESS WAITING
^AUTH STARTING A USER AUTH AT Thread-11
^AUTH USER RECEIVED
^AUTH PASS RECEIVED
```

Hostname: Constantine

Red Auth Port (TCP): 13000

Blue Auth Port (TCP): 14000

UDP Port Range: 20000-20200

Max Number of Hosts: 20

☑ isRNAuthOnline ☑ isBNAuthOnline ☑ hadFirstUser

☑ hasUploadedDataToALocalRedNode ☑ hasDownloadedDataFromALocalRedNode

□ hasUploadedDataToAnotherBlueNode □ hasDownloadedDataFromAnotherBlueNode

☑ Keep Scrolled Down

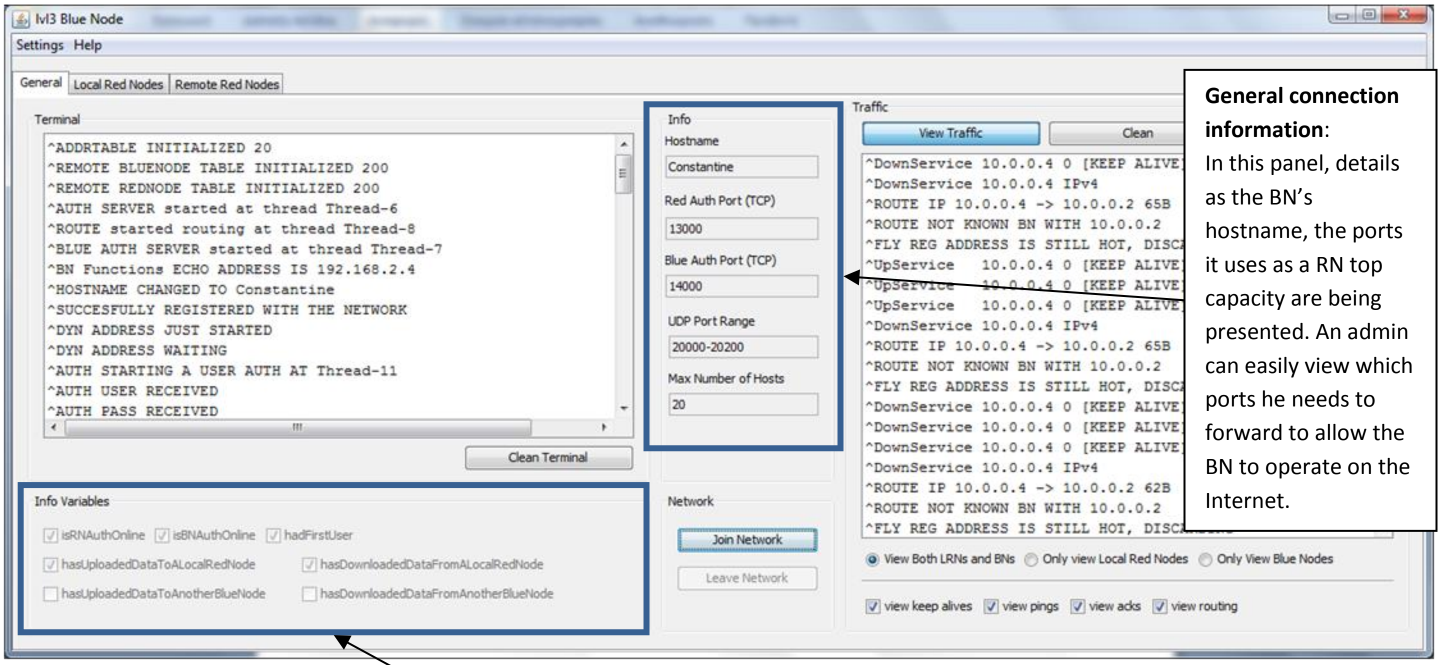
```
^DownService 10.0.0.4 0 [KEEP ALIVE]
^DownService 10.0.0.4 IPv4
^ROUTE IP 10.0.0.4 -> 10.0.0.2 65B
^ROUTE NOT KNOWN BN WITH 10.0.0.2
^FLY REG ADDRESS IS STILL HOT, DISCARDING
^UpService   10.0.0.4 0 [KEEP ALIVE]
^UpService   10.0.0.4 0 [KEEP ALIVE]
^UpService   10.0.0.4 0 [KEEP ALIVE]
^DownService 10.0.0.4 IPv4
^ROUTE IP 10.0.0.4 -> 10.0.0.2 65B
^ROUTE NOT KNOWN BN WITH 10.0.0.2
^FLY REG ADDRESS IS STILL HOT, DISCARDING
^DownService 10.0.0.4 0 [KEEP ALIVE]
^DownService 10.0.0.4 0 [KEEP ALIVE]
^DownService 10.0.0.4 0 [KEEP ALIVE]
^DownService 10.0.0.4 IPv4
^ROUTE IP 10.0.0.4 -> 10.0.0.2 62B
^ROUTE NOT KNOWN BN WITH 10.0.0.2
^FLY REG ADDRESS IS STILL HOT, DISCARDING
```

● View Both LRNs and BNs ○ Only view Local Red Nodes ○ Only View Blue Nodes

☑ view keep alives ☑ view pings ☑ view acks ☑ view routing

Εικόνα 23 BlueNode main window

1st Tab General Information



```
^ADDRTABLE INITIALIZED 20
^REMOTE BLUENODE TABLE INITIALIZED 200
^REMOTE REDNODE TABLE INITIALIZED 200
^AUTH SERVER started at thread Thread-6
^ROUTE started routing at thread Thread-8
^BLUE AUTH SERVER started at thread Thread-7
^BN Functions ECHO ADDRESS IS 192.168.2.4
^HOSTNAME CHANGED TO Constantine
^SUCCESFULLY REGISTERED WITH THE NETWORK
^DYN ADDRESS JUST STARTED
^DYN ADDRESS WAITING
^AUTH STARTING A USER AUTH AT Thread-11
^AUTH USER RECEIVED
^AUTH PASS RECEIVED
```

Hostname: Constantine
Red Auth Port (TCP): 13000
Blue Auth Port (TCP): 14000
UDP Port Range: 20000-20200
Max Number of Hosts: 20

```
^DownService 10.0.0.4 0 [KEEP ALIVE]
^DownService 10.0.0.4 IPv4
^ROUTE IP 10.0.0.4 -> 10.0.0.2 65B
^ROUTE NOT KNOWN BN WITH 10.0.0.2
^FLY REG ADDRESS IS STILL HOT, DISCA
^UpService   10.0.0.4 0 [KEEP ALIVE]
^UpService   10.0.0.4 0 [KEEP ALIVE]
^UpService   10.0.0.4 0 [KEEP ALIVE]
^DownService 10.0.0.4 IPv4
^ROUTE IP 10.0.0.4 -> 10.0.0.2 65B
^ROUTE NOT KNOWN BN WITH 10.0.0.2
^FLY REG ADDRESS IS STILL HOT, DISCA
^DownService 10.0.0.4 0 [KEEP ALIVE]
^DownService 10.0.0.4 0 [KEEP ALIVE]
^DownService 10.0.0.4 0 [KEEP ALIVE]
^DownService 10.0.0.4 IPv4
^ROUTE IP 10.0.0.4 -> 10.0.0.2 62B
^ROUTE NOT KNOWN BN WITH 10.0.0.2
^FLY REG ADDRESS IS STILL HOT, DISCA
```

● View Both LRNs and BNs ○ Only view Local Red Nodes ○ Only View Blue Nodes

☑ view keep alives ☑ view pings ☑ view acks ☑ view routing

Info Variables

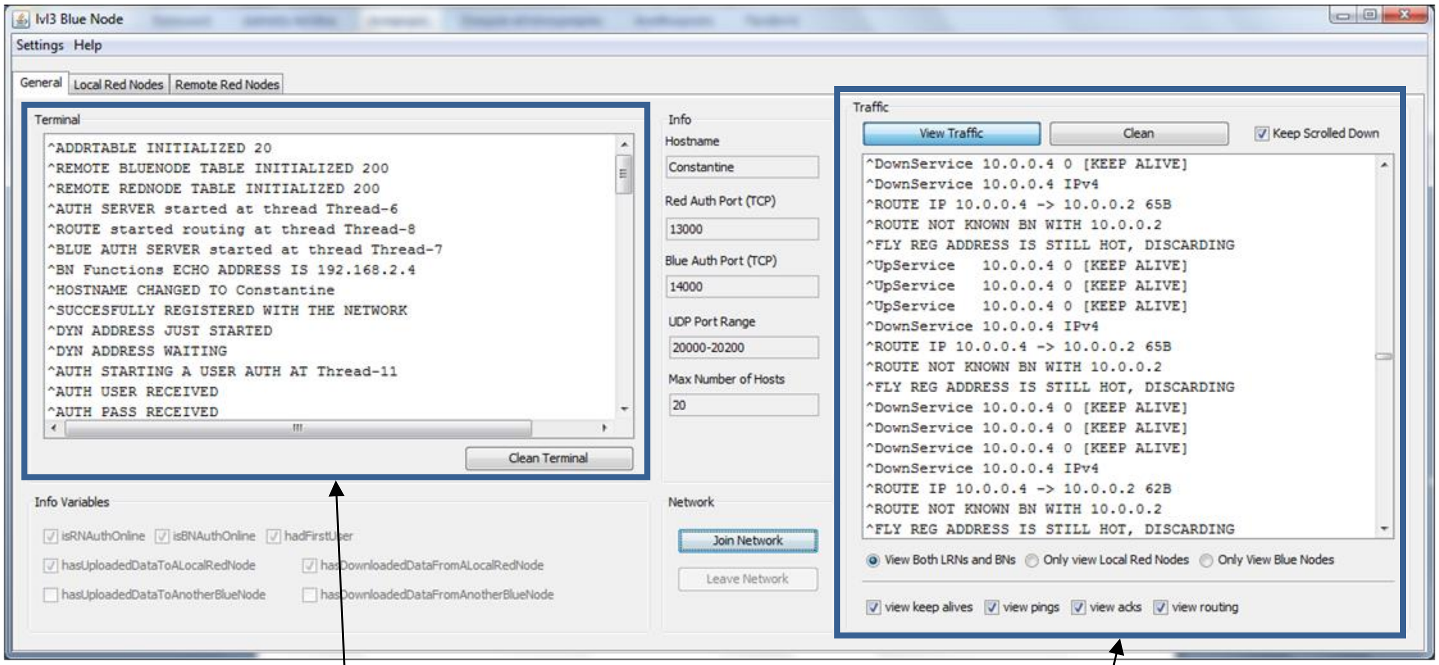
☑ isRNAuthOnline ☑ isBNAuthOnline ☑ hadFirstUser
☑ hasUploadedDataToALocalRedNode ☑ hasDownloadedDataFromALocalRedNode
☐ hasUploadedDataToAnotherBlueNode ☐ hasDownloadedDataFromAnotherBlueNode

General connection information: In this panel, details as the BN's hostname, the ports it uses as a RN top capacity are being presented. An admin can easily view which ports he needs to forward to allow the BN to operate on the Internet.

Information Variables Panel:
The presenting variables are dynamic and may notify an admin for events that have been trigered during the application's runtime. When one is checked it signifies that the event has occurred. The variables are the following:

- isRNAuthOnline ~ Whether the BN has requested from the OS and opened the required port needed to accept RN connections.
- isBNAuthOnline ~ Whether the BN has requested from the OS and opened the required port needed to accept BN connections.
- hadFirstUser ~ Whether the BN has accepted its fist RN.
- hasUploadedDataToALocalRedNode ~ Whether the BN has sent data to a Local RN.
- hasDownloadedDataFromALocalRedNode ~ Whether the BN has received data from a Local RN.
- hasUploadedDataToAnotherBlueNode ~ Whether this BN has sent data to another BN.
- hasDownloadedDataFromAnotherBlueNode ~ Whether this BN has received data from another BN

Picture 24 BlueNode info variables ports and hostname



```
^ADDRTABLE INITIALIZED 20
^REMOTE BLUENODE TABLE INITIALIZED 200
^REMOTE REDNODE TABLE INITIALIZED 200
^AUTH SERVER started at thread Thread-6
^ROUTE started routing at thread Thread-8
^BLUE AUTH SERVER started at thread Thread-7
^BN Functions ECHO ADDRESS IS 192.168.2.4
^HOSTNAME CHANGED TO Constantine
^SUCCESFULLY REGISTERED WITH THE NETWORK
^DYN ADDRESS JUST STARTED
^DYN ADDRESS WAITING
^AUTH STARTING A USER AUTH AT Thread-11
^AUTH USER RECEIVED
^AUTH PASS RECEIVED
```

Hostname: Constantine

Red Auth Port (TCP): 13000

Blue Auth Port (TCP): 14000

UDP Port Range: 20000-20200

Max Number of Hosts: 20

Info Variables

☑ isRNAuthOnline ☑ isBNAuthOnline ☑ hadFirstUser

☑ hasUploadedDataToALocalRedNode ☑ hasDownloadedDataFromALocalRedNode

☐ hasUploadedDataToAnotherBlueNode ☐ hasDownloadedDataFromAnotherBlueNode

☑ Keep Scrolled Down

```
^DownService 10.0.0.4 0 [KEEP ALIVE]
^DownService 10.0.0.4 IPv4
^ROUTE IP 10.0.0.4 -> 10.0.0.2 65B
^ROUTE NOT KNOWN BN WITH 10.0.0.2
^FLY REG ADDRESS IS STILL HOT, DISCARDING
^UpService   10.0.0.4 0 [KEEP ALIVE]
^UpService   10.0.0.4 0 [KEEP ALIVE]
^UpService   10.0.0.4 0 [KEEP ALIVE]
^DownService 10.0.0.4 IPv4
^ROUTE IP 10.0.0.4 -> 10.0.0.2 65B
^ROUTE NOT KNOWN BN WITH 10.0.0.2
^FLY REG ADDRESS IS STILL HOT, DISCARDING
^DownService 10.0.0.4 0 [KEEP ALIVE]
^DownService 10.0.0.4 0 [KEEP ALIVE]
^DownService 10.0.0.4 0 [KEEP ALIVE]
^DownService 10.0.0.4 IPv4
^ROUTE IP 10.0.0.4 -> 10.0.0.2 62B
^ROUTE NOT KNOWN BN WITH 10.0.0.2
^FLY REG ADDRESS IS STILL HOT, DISCARDING
```

● View Both LRNs and BNs ○ Only view Local Red Nodes ○ Only View Blue Nodes

☑ view keep alives ☑ view pings ☑ view acks ☑ view routing

Terminal logging window:
In this window all the BN's basic and executional information is being appended during the application's runtime. If there is any problem, the admin may easily copy the information text for further analysis. With the **Clean Terminal** button, the messages are cleared.

Traffic logging window:
In this window information for all the received and send packets are being appended in an abstract form. An admin may allow or disable the window from appending live data by toggling the **View Traffic** button while he may click the **clean** button to clear the window. Moreover, there are content checkboxes to let him verbose selective information such as ex. only acks and routing.

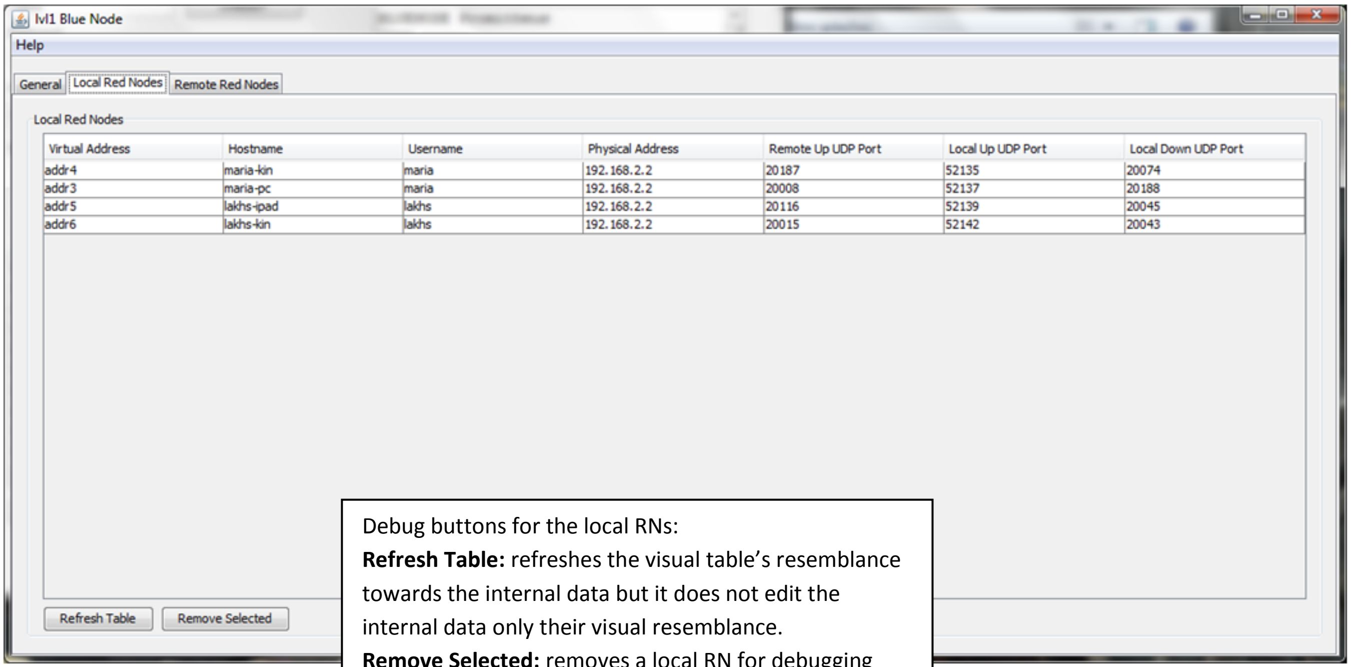
Picture 25 BlueNode console & traffic information

2nd Tab Local Red Nodes

In this tab, information from all the connected red nodes may be displayed. Each line on the visual table represents one local connected RN entry, therefore, an admin may observe new lines to be added when new RNs are connected and lines to be removed when RNs are disconnected. A line depicts multiple information in its cells for the resembling RN. Furthermore, an admin may select one line and for its respective RN entry he may be allowed to execute some debugging options to either escape in a bug and not to drop the whole BN but the corrupt entry instead or to refresh the visual table in its overall in case he suspects that some data may not be displayed correctly.

Each RN entry on the table demonstrates the following elements:

- The host's virtual address ~ An RNs main attribute is its virtual address as each RN has a unique one.
- Hostname ~ Every RN's hostname is unique as its virtual address and, therefore, its associated with the virtual address under a 1-1 correlation.
- Username ~ From which username the given RN is owned from. A user may own multiple devices therefore an admin may observe multiple RNs with the same username
- Physical Address ~ The physical address from where the RN is connected from, this piece of information is confidential only for the local RN and may not be transmitted to the Tracker.
- Uplink port ~ the UDP port which is used for sending traffic to the RN
- Downlink port ~ the UDP port which is used for receiving traffic from the RN



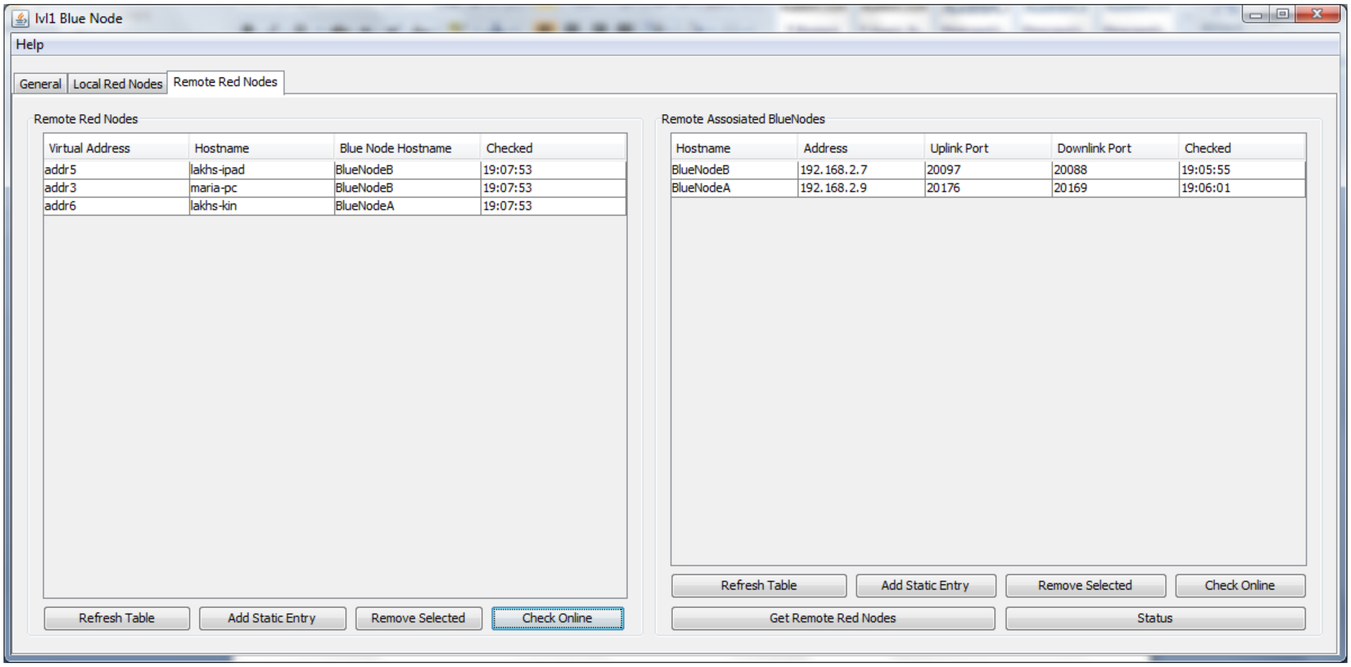
Virtual Address	Hostname	Username	Physical Address	Remote Up UDP Port	Local Up UDP Port	Local Down UDP Port
addr4	maria-kin	maria	192.168.2.2	20187	52135	20074
addr3	maria-pc	maria	192.168.2.2	20008	52137	20188
addr5	lakhs-ipad	lakhs	192.168.2.2	20116	52139	20045
addr6	lakhs-kin	lakhs	192.168.2.2	20015	52142	20043

Debug buttons for the local RNs:
Refresh Table: refreshes the visual table's resemblance towards the internal data but it does not edit the internal data only their visual resemblance.
Remove Selected: removes a local RN for debugging reasons, it should not be used as a kick option but as a tool to eliminate a malfunctioning RNs given the presence of an admin.

Picture 26 BlueNode Local Red Nodes

3d Tab Remote Red Nodes

Upon revising the Unity Network's routing model, it should be reminded that the Remote RNs (or RRNs otherwise) are to be found as clients under different BNs, therefore, a BN may start connections towards other BNs and from them, to collect their RNs and transfer data to them through their hosting BN. Given the above reminder, there should be two tables under this tab one for the remote BNs and another for their respective RNs. Indeed, in the left visual table, data for the RRN (Remote Red Node) entries may be observed whereas in the right, data for the connected BNs. Similarly, with the visual table in the previous tab, in both tables each row represents one entry and its entry's information. Finally, both tables on their bottom have debugging buttons as **Refresh Table** which may be used from an admin to facilitate the platform's operation.



Remote Red Nodes

Virtual Address	Hostname	Blue Node Hostname	Checked
addr5	lakhs-ipad	BlueNodeB	19:07:53
addr3	maria-pc	BlueNodeB	19:07:53
addr6	lakhs-kin	BlueNodeA	19:07:53

Remote Assosiated BlueNodes

Hostname	Address	Uplink Port	Downlink Port	Checked
BlueNodeB	192.168.2.7	20097	20088	19:05:55
BlueNodeA	192.168.2.9	20176	20169	19:06:01

Picture 27 BlueNode Remote Red Nodes (RRDs)

From the above example case, it may be observed from the right table that the example BN is connected with other two BNs 'BlueNodeA' and 'BlueNodeB'. On the left table it may be observed that both 'lakhs-ipad' and 'maria-PC' are RRNs hosted on 'BlueNodeB' wheras 'lakis-kin' on 'BlueNodeA'.

BN Table fields:

- Hostname ~ A BN's name. Each BN has a unique name.
- Address ~ The real IP address of a BN.
- Uplink port ~ The UDP port in which this BN sends traffic to the remote (the selected) BN.
- Downlink port ~ The UDP port in which this BN receives traffic from the remote (the selected) BN.

RRN Table fields:

- Virtual Address ~ the remote RNs virtual address. It's the first attribute as it is frequently used for routing data.
- Hostname ~ the RRNs Hostname, uniquely correlated with its virtual address.
- Blue Node Hostname ~ the hosting BN for the selected RRN entry.

Blue Node Application Installation and Running

Since the application is continuously updated the application's installation and running process are explained in detail under the application's page on GitHub, accessible from the following url:

https://github.com/kostiskag/unitynetwork-bluenode

Hosting a Blue Node

When it comes to the development of service applications, it is a good practice to keep all the available settings under one place, therefore as mentioned, the BN application keeps all its configuration files on its directory which it may read on its start. From these files, the **bluenode.conf** file is responsible to store the application's main configuration settings. An example case of this file may be observed in the next picture.

Editing the configuration file

Under the configuration file, **bluenode.conf** several options may be defined, under this section the most significant are being explained.

- Network ~ true, should an admin desire to make this BN a part of a whole network or false, for a standalone BN.
- UseHostList ~ Under the case of a standalone BN (Network = false), the BN may authenticate hosts based on a list of hosts file named **host.list** if this option is set to true, otherwise, no authentication may be required and RNs may connect without giving credentials at all and may be defined to select their own hostnames or to be appointed random ones.
- udpstart, udpend ~ The BNs defined UDP port range. The same port range should be allowed to be forwarded to the Internet.
- RedNodeLimit ~ RedNode maximum capacity, zero for an unlimited number.
- UseGUI ~ Whether the BN is to be run under a GUI or terminal mode.
- Log ~ Whether the BN should create a log file named **bluenode.log** to capture runtime history.

Editing the host file

Upon the selection of a standalone BN and the selection to use a host file to authenticate the connecting hosts, an admin may define the hosts under the **host.list** file found on the application's directory. Each uncommented line inside the file defines a host's credentials divided by space. The host has to be given a username, a password and a hostname which a user should provide in order for its host to be authenticated. Upon authentication the defined virtual IP may be appointed to the respective host based on its defined incremental number which has to be unique for each entry.

RSA Public and Private key generation

A BN application may automatically generate a keypair in its start given that there is not any. When an admin believes that the BN's private key has been compromised he may terminate the process and simply delete the two files and start the application again. When started, the BN, may generate a different keypair.

The bluenode.conf file

```
######################################
#   BlueNode Configuration File     #
######################################

#
# Insructions for setting up the config file
#
# Do not comment any variable nor remove any from this file as this will result
# in an application error. Change the value to an appropriate input as described
# instead. If this file gets messed up, you may delete it and it will be
# auto-generated from the app.
#

#
# Network Type
#
# Network = false - for Local Network. The BlueNode may not connect to a tracker and will
# serve only local connected RedNodes
# Network = true - for Full Network. The BlueNode will seek a tracker to be a part in
# a full network with other BlueNodess and remote RedNodess
#
Network = false

#
# variables for FullNetwork
#
# if you have selected Local Network these variables will not take any effect
#

# Provide the central tracker's address
# with an IP address or with a domain.
# Provide the tracker's TCP auth port. 8000 is the default.
UnityTrackerAddress = 192.168.1.1
UnityTrackerAuthPort = 8000

# This is the network's reverse lookup time in minutes, it has to be double from
# The tracker's ping time.
TrackerMaxIdleTimeMin = 2

# Set the Name of this BlueNode
# In Full Network the BN's name must be registered in the tracker's database
# Set the TCP auth port. 7000 is the default.
Name = BlueNode
AuthPort = 7000

#
# variables for LocalNetwork
#
# if you have selected Full Network these will not take effect
#

…
```

The host.list file

```
################################
#   Host-Client List File     #
################################

# Use like:
# Username Password Hostname Virtual_Address_Number
#
# for the Virtual_Address_Number field use an integer starting from number 1, # the BlueNode will
# auto convert the number to its respective IP address
# you may repeat the same Username and Password with a different Hostname
# and Virtual_Address in case the same user owns more than one devices
#
# ex.
# bob 12345 bob-laptop 2
# bob 12345 bob-mobile 3
#
pakis 12345 pakis-laptop 3
kostis abcd kostis-pc 2
```

The public.key and private.key files of a BN

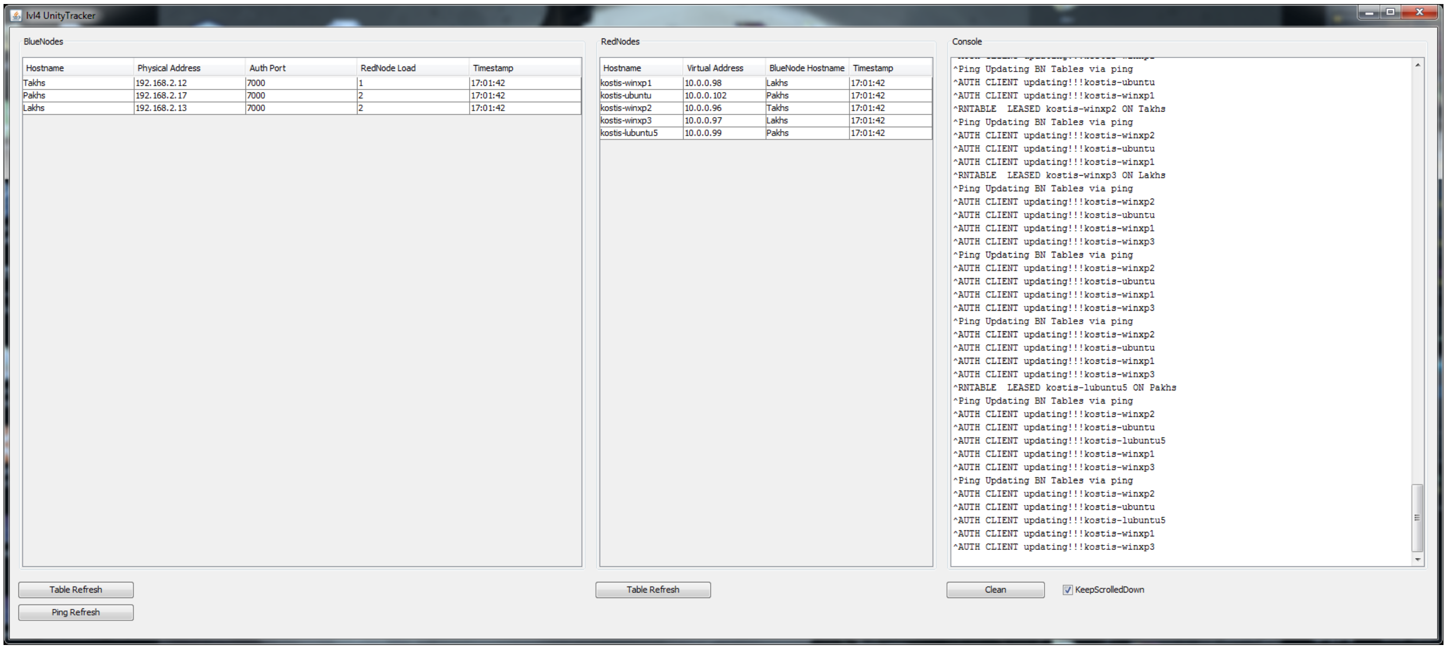
```
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA1/lOrpfVFXgKTk8QWjYj
VFe0hSRLCvZhM2fULj1sRq0Z89gaahJLO1q/xQIWFuNvDv++9d6apqYhhBJ68WPS
KghXWkc64yy9X62A0Yx9ycNW3k7Vm0GEf9q7lHVYmyrJG1zDPaObbW0IsxXTenNc
dgQP5CCN8hcOMXSNZMR8/0Oce10HX+pjjqItuK5qJ3gLcySrHal5fkY3sAUvukZ/
vGR3FDzpYRfse0RF6OuoWoEL/nL60WNm/ll+j5Mwsw3LWy00rQID31cJ9qaclLgO
sl0+hhnJBikOcjx8bi31jH8biTTt6oOcOZDtBqQ2TlbH6gI4+Hts2IR/0ph7jbzM
LQIDAQAB
-----END PUBLIC KEY-----
```

```
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEA1/lOrpfVFXgKTk8
ahJLO1q/xQIWFuNvDv++9d6apqYhhE
m0GEf9q7lHVYmyrJG1zDPaObbW0Is
X+pjjqItuK5qJ3gLcySrHal5fkY3sAUvuk
0WNm/ll+j5Mwsw3LWy00rQID31cJ9
6oOcOZDtBqQ2TlbH6gI4+Hts2IR/0ph
2ZQDdHEVAGuaT1Spw31+P1lJk41Xv5xbPTZ6zwCQ093tHWJRMDDp+EpwAhBT+ke
wGtzoAedyR42iJUAfEUAYdJH+y1/mA/kxITS1pL5olhLi3BCdMGdVr1//+F/d0GG
Ieq+vMKtba3kVG7b/UYmtR04jqbPbSIn7k9RZtpO/d2vGH41lXg1UyUgtEQgIpOW
WDaG074rL6KuxkM97e5D/e4oPptZBDvi1uQcFT1yIFDkvs6gjJE86DWZH7uIMejf
1pABUsPwzOPeJ5gOaw+qhSBUtPuRzB4VILfX9vvdwNxFSB14RI72DB23kaFyVioK
zTm3iGkCgYEA/Ebj50+zYpTvJWgcjlVMEeu9AOqEHqH7QaM61lyyXlK5zSEfea6L
3vsWisUTWM0oBiUeKJ3n9im3WJ+kgOrOFPrZX503opul4D7xuKVzBnDHbg2Zx+AB
uWyrnUGkW6e4bhKEtdvst8v7PuUWkVjp1xyjvO4Tk0Ua/CqDX2mHA08CgYEA2ylD
V/BcwqGugcthNSGXVQElqtTeaiVvTbby8Zk0jHcrepJG76yQqSW+TMNFtG8Pjj9J
PcvmdbMAdZTqPgiNV2MO88QYHcT4h9Upa/giAsb1BAsnZqlCj6YRHwwcXIWEAuFp
7xsWYeF0jKJB1NfsQD5f1vFyr1kYPPTgZBrgicMCgYEA0IpNVOYGdKSG99YTXPlX
s6y6hWpXIvdlusGTHqZr8BrUaqRJ342RJBdNcBMvRgX5YvMF9i9qE4wyerklBEiV
aLRgQnC1D984hKGjsa5a4mUSBoCJsbcT1dLmHk2n7vg7Ngpq1+ZfzSN6omg/epEU
ZHTRSZlIZ0IF55PBG3shV3MCgYBSwmY30wB0TvHC+bYfhivLYb+DnxbOJoy9YBSl
vyDk2iuFAa/f2d5WwXX3LtYnqLjLEoLp3xGL6KiHvlAmVLxq/3EqBCbHNxZS1N/r
cawGOHNVr5CVZ91GuYNFoiEjnxeWruB99lChba3BXZRWd6MzL1qppEuWg6Jvglkp
9CxOWQKBgQCZ5ZqBmOkLqlL9x9COZ9IL6djMIRmS4HXmCq1tSjzXjOAOg6Y/pw7w
Nz/UWHi/f4RECnVsC2+I5D3875onOAMVjsFwii4oGs/F3F/4EJ58ijs1EDw/ZO3J
wuvKuhNu0kqfc7GT9EgOgQbdK5P/CCYf8W+W+rciz15t/zPJapYorw==
-----END RSA PRIVATE KEY-----
```

2.3 Unity Network Tracker

A Tracker is the application needed to coordinate a network that its being apparated by more than one BNs. The tracker's first task in one line is to either allow or reject BNs to participate on the network and to do the same thing for the connecting RNs through the BNs. Its next task is to trace back the BNs in order to examine whether they are **still-alive** after a defined amount of time as the tracker does not keep persistent connections towards the BNs. The tracker application similarly with the BN application may be run under a terminal or in a GUI mode. Another thing in common with the BN application is that it may be defined by the configuration files and the keypair-files found under the application's directory which make the task to set a tracker extremely convenient. Regarding a network's intended size, a tracker may keep the entries under a local database file in a SQLite scheme or decide to host an external database system possibly connected with a web interface as well, to let the users manage their content via the web. Ultimately, a tracker can be scaled to either fit a small or an extensive network. Ultimately, since the network is using the 10.0.0.0/8 range, assigned by IANA under free use, the network may reach a top capacity of $2^{24} – 2 – 2$ special reserved addresses which makes a total of 16.777.212 IP addresses in total, therefore, 16.777.212 available host entries.

On the following window, an instance of a Unity Network Tracker may be observed. Under a tracker's GUI, an admin may observe two tabs. The first tab is about viewing the live network data whereas the second is about defining and registering users, RNs and BNs in the database. The first tab, which is selected in the following figure has three columns. The first is about viewing all the authenticated BNs and their information, the second is about viewing all the connected RNs to the BNs from the first list whereas the third is a console output which depicts login, logout and authentication information.



Hostname	Physical Address	Auth Port	RedNode Load	Timestamp
Takhs	192.168.2.12	7000	1	17:01:42
Pakhs	192.168.2.17	7000	2	17:01:42
Lakhs	192.168.2.13	7000	2	17:01:42

Hostname	Virtual Address	BlueNode Hostname	Timestamp
kostis-winxp1	10.0.0.98	Lakhs	17:01:42
kostis-ubuntu	10.0.0.102	Pakhs	17:01:42
kostis-winxp2	10.0.0.96	Takhs	17:01:42
kostis-winxp3	10.0.0.97	Lakhs	17:01:42
kostis-lubuntu5	10.0.0.99	Pakhs	17:01:42

```
^Ping Updating BN Tables via ping
^AUTH CLIENT updating!!!kostis-ubuntu
^AUTH CLIENT updating!!!kostis-winxp1
^RNTABLE  LEASED kostis-winxp2 ON Takhs
^Ping Updating BN Tables via ping
^AUTH CLIENT updating!!!kostis-winxp2
^AUTH CLIENT updating!!!kostis-ubuntu
^AUTH CLIENT updating!!!kostis-winxp1
^RNTABLE  LEASED kostis-winxp3 ON Lakhs
^Ping Updating BN Tables via ping
^AUTH CLIENT updating!!!kostis-winxp2
^AUTH CLIENT updating!!!kostis-ubuntu
^AUTH CLIENT updating!!!kostis-winxp1
^AUTH CLIENT updating!!!kostis-winxp3
^Ping Updating BN Tables via ping
^AUTH CLIENT updating!!!kostis-winxp2
^AUTH CLIENT updating!!!kostis-ubuntu
^AUTH CLIENT updating!!!kostis-winxp1
^AUTH CLIENT updating!!!kostis-winxp3
^Ping Updating BN Tables via ping
^AUTH CLIENT updating!!!kostis-winxp2
^AUTH CLIENT updating!!!kostis-ubuntu
^AUTH CLIENT updating!!!kostis-winxp1
^AUTH CLIENT updating!!!kostis-winxp3
^RNTABLE  LEASED kostis-lubuntu5 ON Pakhs
^Ping Updating BN Tables via ping
^AUTH CLIENT updating!!!kostis-winxp2
^AUTH CLIENT updating!!!kostis-ubuntu
^AUTH CLIENT updating!!!kostis-lubuntu5
^AUTH CLIENT updating!!!kostis-winxp1
^AUTH CLIENT updating!!!kostis-winxp3
^Ping Updating BN Tables via ping
^AUTH CLIENT updating!!!kostis-winxp2
^AUTH CLIENT updating!!!kostis-ubuntu
^AUTH CLIENT updating!!!kostis-lubuntu5
^AUTH CLIENT updating!!!kostis-winxp1
^AUTH CLIENT updating!!!kostis-winxp3
```

Picture 28 Tracker main window

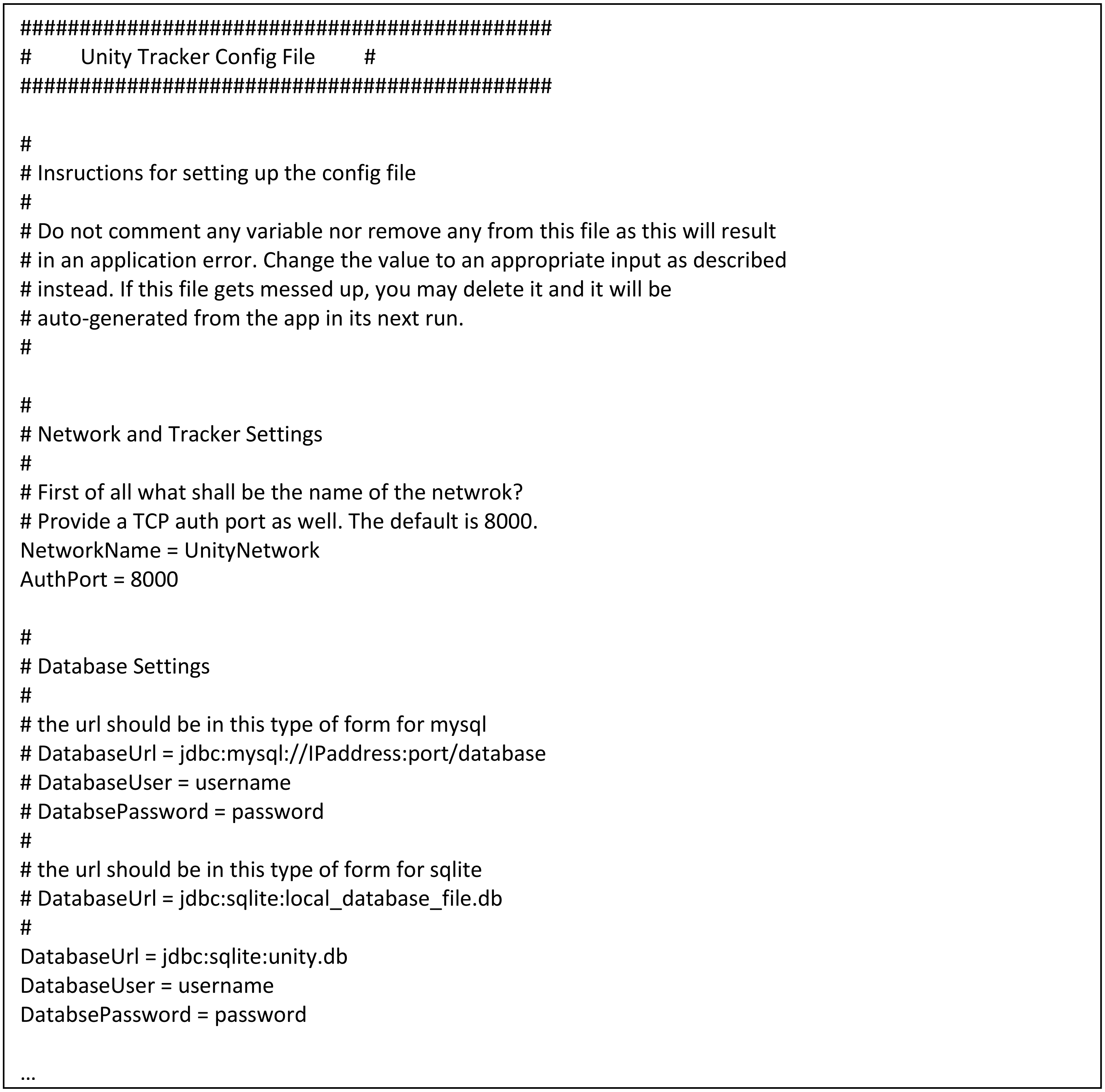
Unity Network Tracker Installation and Running

As the other two applications, the tracker's app may continuously be updated. For this reason, the application's installation and running process are explained in detail under the application's page on GitHub, accessible from the following url:

https://github.com/kostiskag/unitynetwork-tracker

Hosting a Unity Network Tracker

The first step after downloading and setting the tracker's app for use is to edit its configuration file to set the service under a desired manner. Inside the **tracker.conf** file, visible in the following figure, the most significant choices an admin may choose is to define the TCP port used from the BNs to locate the server, decide whether to use a local sql file or connect the application to a larger database service and whether or not to run the app under GUI or terminal. It is important to note that the tracker's defined authentication port in the configuration file should be forwarded by the admin appropriately to the Internet whereas it is a good practice for the tracker's host to own a domain address to be easily located by the connecting BNs and RNs. As a rule of thumb, for a small network, there may be selected a local database file with the GUI option whereas for a bigger, no GUI with an external service linked to a web-interface.



```
######################################################
#        Unity Tracker Config File        #
######################################################

#
# Insructions for setting up the config file
#
# Do not comment any variable nor remove any from this file as this will result
# in an application error. Change the value to an appropriate input as described
# instead. If this file gets messed up, you may delete it and it will be
# auto-generated from the app in its next run.
#

#
# Network and Tracker Settings
#
# First of all what shall be the name of the netwrok?
# Provide a TCP auth port as well. The default is 8000.
NetworkName = UnityNetwork
AuthPort = 8000

#
# Database Settings
#
# the url should be in this type of form for mysql
# DatabaseUrl = jdbc:mysql://IPaddress:port/database
# DatabaseUser = username
# DatabsePassword = password
#
# the url should be in this type of form for sqlite
# DatabaseUrl = jdbc:sqlite:local_database_file.db
#
DatabaseUrl = jdbc:sqlite:unity.db
DatabaseUser = username
DatabsePassword = password

…
```

RSA Public and Private key generation

Similarly with the other two applications, a tracker may automatically generate an RSA public/private keypair on its start given that there are no keypair files on its directory. The keys are mainly used from the connecting BNs and RNs to authenticate the tracker's identity prior to the tracker identifying theirs. For this reason, if a tracker's admin believes that its tracker's private key has been compromised and decides to delete the keypair files, two other may be generated on the application's start. Given that the BN's are not aware with this update, the tracker's admin should let the BN admins know that the tracker's key has been changed in order, in their turn, to delete their old tracker's public key copy and accept the new one so that they may be able to connect.

Linking the network's database to a web Interface (optionally)

In order to set the platform with a web interface, the first task at hand is to select an external SQL service such as mySQL, PostgreSQL or any another SQL based database. Next, after defining the database, its password and updating the tracker's config file, a web server should be installed and set to read from the database. On its turn, the server should be able to retrieve and store data from the database. A good example of an implementation for this architecture is to use a mySQL service and a PHP 5 or 6 enabled Apache service. In a Unix based system the process may be very close with the following commands.

```
su

apt-get install mysql-server mysql-client

apt-get install apache2

apt-get install php6 libapache2-mod-php6

/etc/init.d/apache2 restart
```

After that, there is the option to install and set PhpMyAdmin in order to view the database data through the web. The PhpMyAdmin has to be installed to the apache's web content directory appropriately and linked with the database. On a successful set the admin may define a web-interface by PHP. Last but not least, the admin should allow port 80 to be forwarded in order to let web clients access the web-page, create accounts and manage their data.

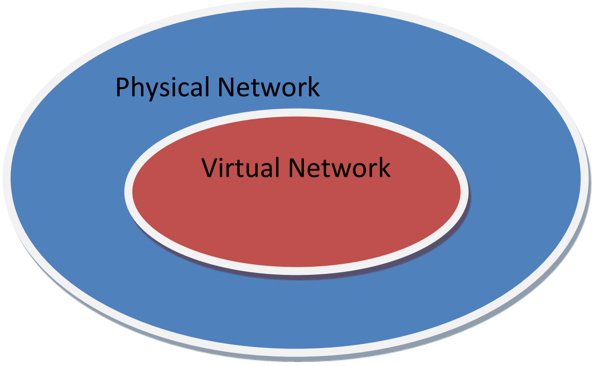
3. VPN Principles & Classic VPN analysis

3.1 Introductory VPN notions

When were the VPNs introduced, which were their initial goals?

VPN networks started around the age of 2000 along with the introduction of the PPTP protocol [5]. The age's main goal regarding VPNs was to facilitate the public services, the companies or the organizations to be able to privately access network resources as: network printers, databases, internal web pages and file servers given that these resources were scattered among many different buildings, remote to one another. In order for a company to connect its infrastructure it would either need to literally dig in the ground and install a network wire from one building to another or use the Internet, to connect its network segments by making use of VPN tunnels [6]. A VPN allowed a company to unite its segments without letting any other external members from the Internet be allowed to access its internal resources. From the other hand, the employees themselves, if happened to be away from the organization, could use a VPN client to 'hook-up' their remote device (host) to the organization's LAN. From there after, their computer could access the desired service with the same behavior as a local host.

A VPN's scope



One of the most common notions when referring to a VPN or a virtual network in general may be its scope. A VPN's scope, simply stated, is the angle in which it is being examined. Typically, a virtual network has two scopes, the physical or external and the virtual or internal [4]. It may be better described if we had to view a VPN be resembled as a bubble, it would lay inside another bigger bubble, the actual network, where members of the inner bubble have to be members to the external but not all the members of the external are necessarily members of the inner bubble, the VPN. Given that, if all the routing and non-observable devices were to be removed, a network is simply apparated by its connected hosts. If we had to examine its behavior in terms of exchanging data, a simple example could be an external network apparated by hosts named: A, B, C, D and E were out of them, hosts: A, B and E decided to create a VPN to privately exchange data. In the beginning of their connection to the VPN, they would first collect a name for the virtual network and then they would be able to address one another given that name in order to exchange data. Let's say for the example 'A' was named as '1' in the virtual network, 'B' as '2' and 'E' as '3'. Now, host 'A' has two names, it may be referred both as 'A' or as '1' however host 'C' may refer 'A' only in its first name and not as '1' as it is not a member of the given VPN.

Virtualization in the present

In the very century we are crossing, notions as virtualization, clouding, object-orientation and modularity have become extremely popular due to the need to efficiently and abstractedly manage the behavior of systems, like ex. hardware devices as virtual objects which may exit and 'live', or in other words, have state (be statefull) inside other systems or other virtual objects. This approach is pursued as from the one hand, hardware may be costly to obtain, timely to set-up and occupies physical space. From the other, for a better object resource management, a better distribution of processing power and co-ordination in a more logical rather than physical level ex. 1000 web server objects may perform from the same device. Distribution and object-orientation, as notions and as a scientific domain, are also concerned and build around the relation a virtual object may present towards its host in terms of:

- Privacy, Should the hosting object be allowed to view the hosted object’s data or not.
- Control, Is the object’s host allowed to terminate the virtual object? Is he allowed to change its state?
- Allocation, Should the host be allowed to send or copy the virtual object into another host?

Similarly, VPNs as other network services, are adjusting to this phase in order to be less related to hardware and more related towards object-orientation.

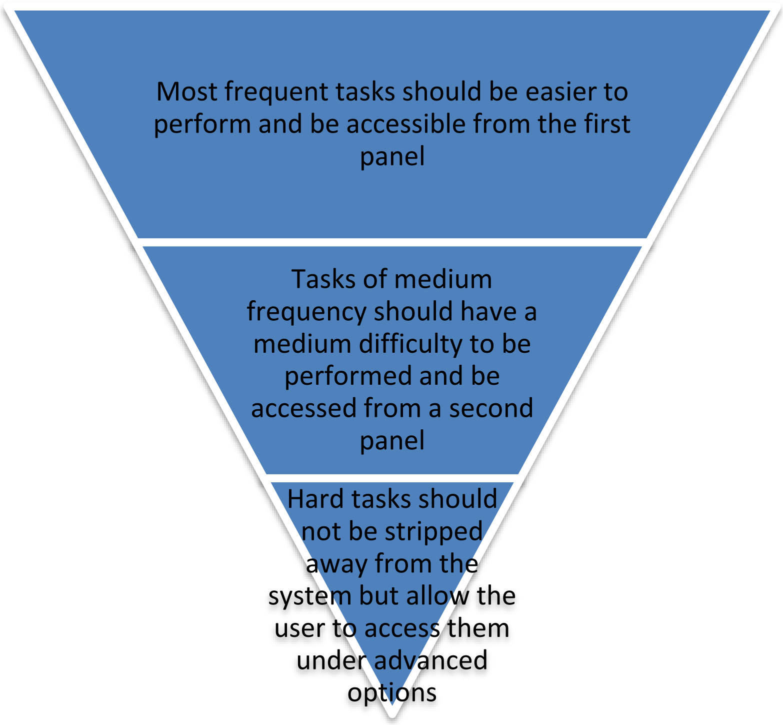
VPNs vs Clouds

As it may be observed, VPNs share many similarities to another technology known as Cloud computing as regards their common background in virtualization and their similar intention to provide services. In contrast towards VPNs, clouds are significantly much more popular in a trending point whereas VPNs are considered an established technology. However, in an attempt to decouple the ‘marketing-in-effect’ around clouds in order to view them as pure technology, many similarities towards VPNs are to be observed. Under this section, common approaches as differences for the two technologies are annotated in order to examine their relation further.

Under their common background regarding providing services, in a cloud, a user may obtain a client application which may allow him to get connected directly with the service and manage the available content. The application is significantly easier to use as it is centered around withholding information from the user rather than letting him have access to its inner structure, approach which from the one hand, emphasizes to the higher sake of simplicity for the user as he comes in contact only with the final, the front, system and to contain any probable competition from the possible rivals of the service’s owner. However, in many conditions may present a direct conflict with modern notions as informing the user about a system’s ambiguity and, in general, letting him know and interact with what is there. Ultimately, **closed architecture** approaches have a direct impact to lead technology to the point of hooking-up users rather than allowing them to have the appropriate control over it. A VPN has a similar usage regarding providing services, only that a VPN uses an **open architecture**, as a user has to first connect to the virtual network and then to access a service from inside the network.

Best Practices for user-centered service development

When building a service around a user, two abstractive notions rumble. The first is the **level of control** a user may be given and the second is the **level of simplicity** the service is built with. When a user has more control over a service he may perform more actions, but the service’s complexity is increased. On the other hand, when he is given less control, the service is easier and less frustrating to use, but it may contain the danger to limit the user’s abilities and narrow the possible spectrum of tasks he could have achieved with more control. Ultimately, to build a good system, it needs to be balanced somewhere in between if the notions were regarded to be compromising to each other. On the other hand, what if we could use these notions in a non-compromising manner and let them self-complement one another? To explain this, if there was the need to observe a user’s probable actions from all the available spectrum of a given service, it would be observed that these form a power distribution related with time and a user’s level of experience. In more detail, most of the time a user uses a service he is attempting a basic set of tasks while the more tailed the tasks are, they are allocated in a smaller amount of time or performed from a smaller sub-group of more advanced users. Under this approach, an architecture should follow a reverse pyramid structure where the most frequent tasks should be the easiest to perform while the more complex an information or a task may be, it should allow the user to dig dipper into the system to perform it without compromising his available options. The following diagram demonstrates this approach.


Most frequent tasks should be easier to perform and be accessible from the first panel
Tasks of medium frequency should have a medium difficulty to be performed and be accessed from a second panel
Hard tasks should not be stripped away from the system but allow the user to access them under advanced options

Unity Network, was built regarding the above model where a user is initially faced with easy to use and manage options that represent the majority of the available tasks, but should he dig deeper, he may find the platform to make use of an open architecture and be allowed additional actions.

3.2 A taxonomy of well-established VPN protocols and solutions

In this section, well-established VPN network solutions, standardized and not, are examined to observe and contrast their behavior towards one another.

PPTP

The PPTP protocol [5] was developed by a union of companies formed by Microsoft, Ascend Communications, 3Com and others and its description was published as an RFC file in June of 1999 [7]. The protocol describes a method in which it can create a tunnel that may forward PPP traffic over IP networks and thus, create a VPN network between two end-point networks. The protocol is connection-oriented and describes the necessary processes for establishing the tunnel as the client's authentication, the exchange of traffic and the client's logging out process.

PPTP connection

Regarding its connection, it uses a control channel over a TCP socket with which it may manage the tunnel's state and health. For the network traffic exchange, it makes use of a modified version of the GRE transmission protocol, and through its channel, PPP datagrams are carried. The GRE protocol, apart from the traffic and its other features, contains data control flow messages used by the control-flow algorithm to dynamically determine and allocate the tunnel's bandwidth. In order for the GRE traffic to be carried, the end points should allow the GRE transmission header to be forwarded.

PPTP security and encryption

For the initial authentication it uses two sub-protocols: PAP, CHAP, MS-CHAPv2. One of its setbacks is that GRE packets are non-encrypted and any MITM may collect or modify the exchanged traffic.

L2TP/IPsec

This particular protocol is a combination of the L2TP and IPsec sub-protocols which may operate independently but they may also complement each other. Therefore, initially they may be examined separately and at the end as a combined model.

L2TP

The L2TP protocol was developed as an advanced version for the PPTP and L2F (Cisco's layer two forwarding protocol) protocols while its model was published as an RFC document in 1999 [8]. In general, its goal and operation to forward PPP traffic over IP networks is similar with its predecessor, although, more security and control-flow features were included.

L2TP connection

The protocol encapsulates the forwarding PPP packets under UDP datagrams or under other protocols as ATM. It may carry two kinds of messages: control messages and the encapsulated PPP packets.

L2TP authentication and encryption

Regarding the authentication process, the protocol makes use of the CHAP and CHAPv2 sub-protocols similarly with PPTP. However, L2TP although implements authentication procedures, on its own it does not make use of any encryption model for the carried traffic.

IPsec

The IPsec is a protocol, which introduces encryption and authentication methods as confidentiality and integrity, with the aim to provide privacy and security for the carried traffic between two nodes in an IP network. Since the carried traffic is Layer 2 on the TCP/IP model, the two sides do not need to necessarily make use of a secure, application layer service, as SSL, HTTPS or SSH as the very tunnel is encrypted on its basis. The connection's two end points may have agreed under a common password based on a symmetric encryption model or each side to keep a public/private keypair and thus, to achieve an asymmetric or hybrid encryption between them. Finally, the protocol by itself does not introduce any VPN features as it solemnly describes the tunnel's encryption and forwarding over the carried data.

L2TP/IPsec

In this model, the two protocols are to complement one another and collaborate under a common scheme. Each one is responsible for its own set of introduced responsibilities. The L2TP is related with the tunnel's establishment, the virtualization, the user's authentication and logout whereas IPsec for the encryption and transfer of the produced traffic. The initial authentication and the final logout are being performed by a sub-protocol, the ISAKMP, which is encapsulated in UDP datagrams. The virtual networking traffic is encrypted and carried under ESP (transmission layer) packets. Finally, L2TP/IPsec may present the best protocol based model to support a private and secure VPN, however, its most significant setback, as it will be introduced in more detail in the following section, is that even though the traffic is encrypted, there is observable information that may indicate or alert a MITM that VPN traffic is being carried, a setback which may allow an attacker to develop patterns as ex. on which time is the VPN mostly active.

OpenVPN

OpenVPN is an open source software which implements VPN techniques in order to establish secure and private network tunnels. It has been developed by James Yonan in the C programming language, licensed under the GNU license. Some of its features are the use of SSL/TLS encrypted tunnels, NAT traversing techniques, independence towards the application and the operating system as many other customizable features which make it like a swiss-knife for the VPNs. Some of those features are:

Picture 29 OpenVPN

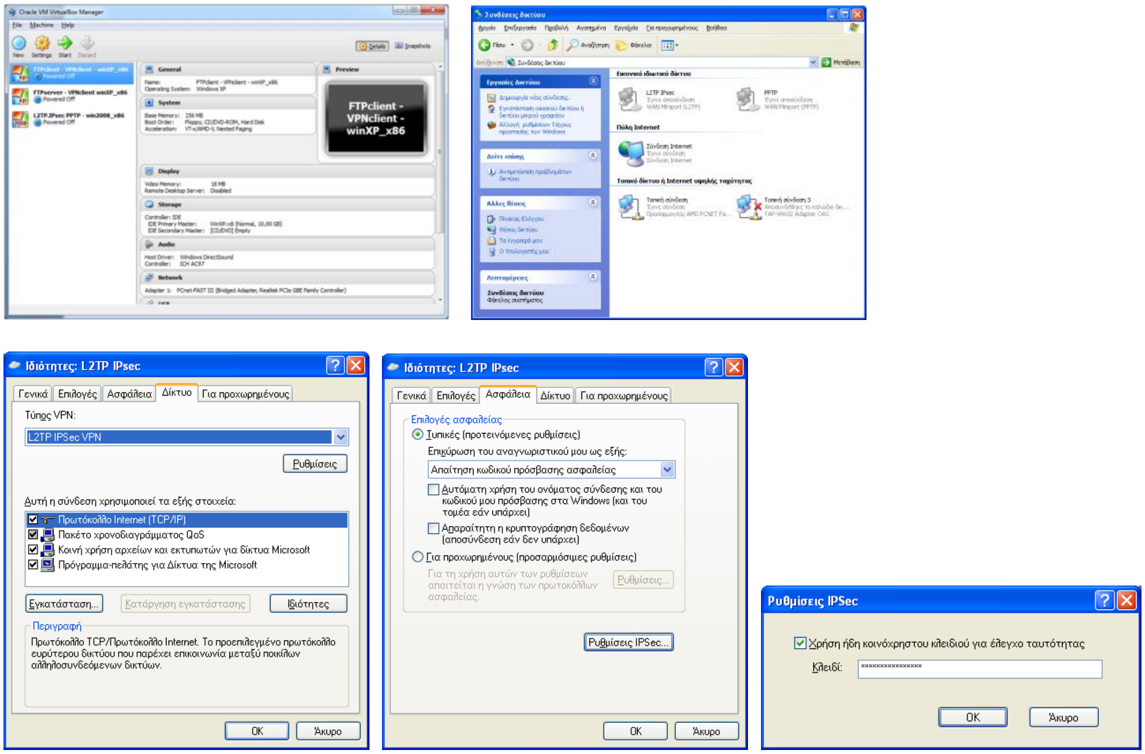
- To connect multiple LANs as a common LAN.

- To connect multiple remote hosts to the same LAN.
- To establish multiple VPNs in a row which may defend the host's identity by using a chain routing model.
- The feature to appoint to a remote host a real IP address in order to for its user to be allowed into services with a geographical limitation which he was not allowed to access from his location.
- To divide the network's load by making use of multiple VPN servers connected to the same LAN.

Apart from all its features, the most significant of them may be that it offers a fully encrypted traffic forwarding model encapsulated over either UDP or TCP which make it significantly harder, for the forwarded traffic over an IP network, to be detected and matched as a VPN traffic by MITMS and third parties, compared to the previously mentioned implementations.

3.3 VPN Protocol Reverse Engineering

In order for the afore introduced protocols to be closely observed, the following environment was established. Three host OS systems were emulated by using the VirtualBox software [6] installed on a physical (existing) host system. From a networking perspective, the three nodes were bridged in the hosting computer's networking card in order to be discoverable from the external Local Area Network members. Regarding their communication roles, the two out of the tree nodes were to establish their own VPN tunnel and exchange data while the third node was used to facilitate their discovery and co-ordination when needed while using to the various VPN protocols. The external hosting system was to not take any active role in the data exchange process but was instead responsible to collect the produced network traffic by monitoring the VPN tunnel from its external scope. The overall task for the two nodes, was to open one VPN tunnel for each protocol at a time, exchange a file via the FTP protocol and then close the tunnel. In the performed process the expected outcome was for the monitoring system to not be able to discover the file which was to be exchanged by observing the network traffic but rather be faced with the VPN traffic. Finally, the observing system was to make use of Wireshark [5] as a network traffic capturing software in order to collect the traffic, store it to files and examine the behavior of the captured VPN protocols.



FTPclient - VPNclient - winXP_x86 — Powered Off
FTPserver - VPNclient winXP_x86 — Powered Off
L2TP.IPsec PPTP - win2008_x86 — Powered Off

Name: FTPclient - VPNclient - winXP_x86
Operating System: Windows XP
Base Memory: 256 MB
Boot Order: Floppy, CD/DVD-ROM, Hard Disk
Acceleration: VT-x/AMD-V, Nested Paging
Video Memory: 18 MB
Remote Desktop Server: Disabled
Controller: IDE
IDE Primary Master: WinXP.vdi (Normal, 10,00 GB)
IDE Secondary Master: [CD/DVD] Empty
Host Driver: Windows DirectSound
Controller: ICH AC97
Adapter 1: PCnet-FAST III (Bridged Adapter, Realtek PCIe GBE Family Controller)

L2TP IPsec — Έγινε αποσύνδεση — WAN Miniport (L2TP)
PPTP — Έγινε αποσύνδεση — WAN Miniport (PPTP)
Σύνδεση Internet — Έγινε σύνδεση — Σύνδεση Internet
Τοπική σύνδεση — Έγινε σύνδεση — Προσαρμογέας AMD PCNET Fa...
Τοπική σύνδεση 3 — Αποσυνδέθηκε το καλώδιο δικ... — TAP-Win32 Adapter OAS

Τύπος VPN: L2TP IPSec VPN

Αυτή η σύνδεση χρησιμοποιεί τα εξής στοιχεία:
☑ Πρωτόκολλο Internet (TCP/IP)
☑ Πακέτο χρονοδιαγράμματος QoS
☑ Κοινή χρήση αρχείων και εκτυπωτών για δίκτυα Microsoft
☑ Πρόγραμμα-πελάτης για Δίκτυα της Microsoft

Περιγραφή
Πρωτόκολλο TCP/Πρωτόκολλο Internet. Το προεπιλεγμένο πρωτόκολλο ευρύτερου δικτύου που παρέχει επικοινωνία μεταξύ ποικίλων αλληλοσυνδεόμενων δικτύων.

Επιλογές ασφαλείας
● Τυπικές (προτεινόμενες ρυθμίσεις)
Επικύρωση του αναγνωριστικού μου ως εξής: Απαίτηση κωδικού πρόσβασης ασφαλείας
□ Αυτόματη χρήση του ονόματος σύνδεσης και του κωδικού μου πρόσβασης στα Windows (και του τομέα εάν υπάρχει)
□ Απαραίτητη η κρυπτογράφηση δεδομένων (αποσύνδεση εάν δεν υπάρχει)
○ Για προχωρημένους (προσαρμόσιμες ρυθμίσεις)
Για τη χρήση αυτών των ρυθμίσεων απαιτείται η γνώση των πρωτοκόλλων ασφαλείας.

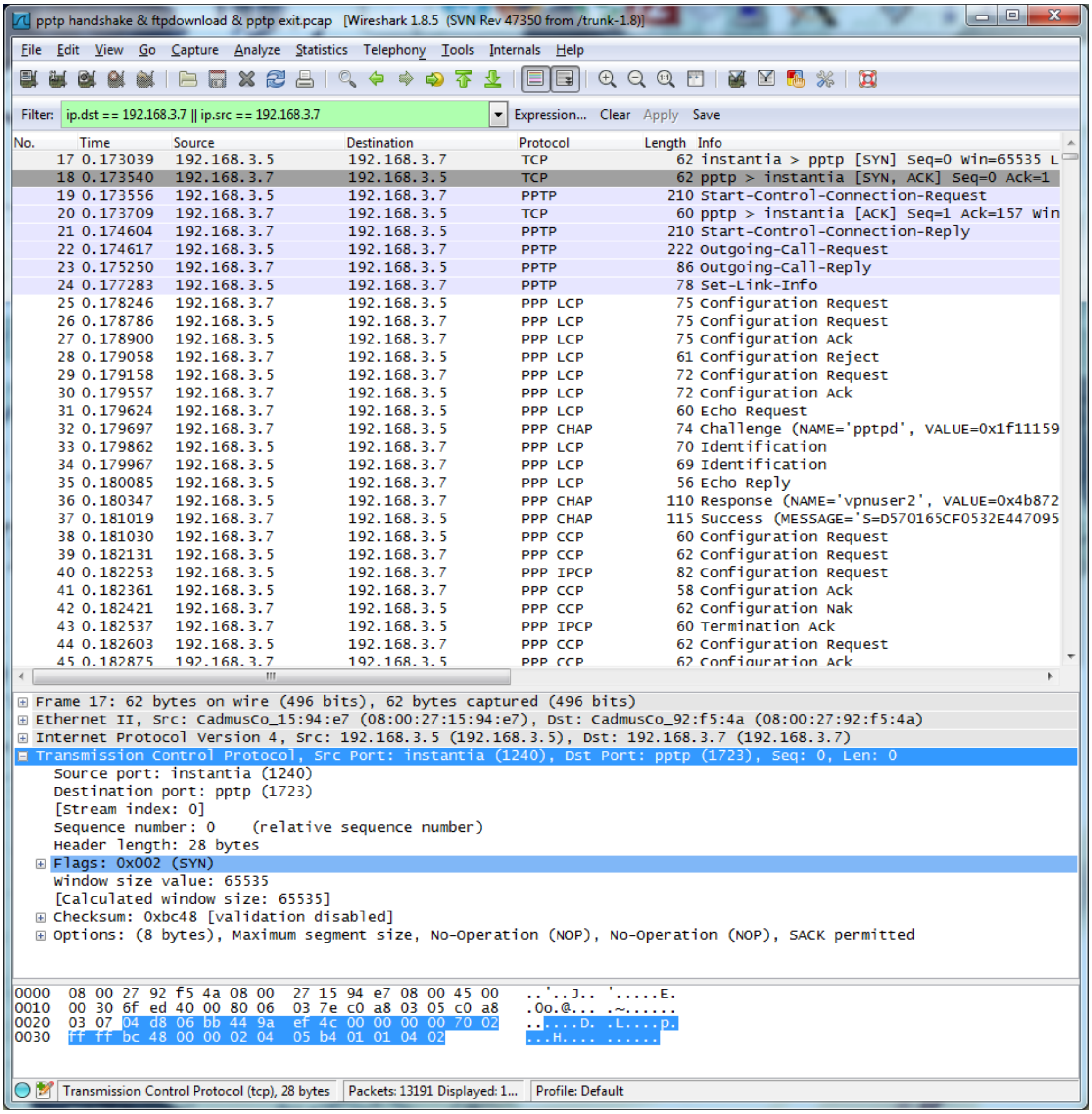
☑ Χρήση ήδη κοινόχρηστου κλειδιού για έλεγχο ταυτότητας
Κλειδί: ****************

Picture 30 Various settings from the multiple VPN protocol testing

PPTP

In the following picture, the PPTP protocol's traffic may be observed in brief. In the saved traffic capture file, produced by Wireshark, the following filter is applied to only display the relevant traffic out of the total captured. This filter will ensure the one of the two nodes to be either the sender or the receiver and since the one node is exchanging data only with the other, only the traffic between the two nodes may be observed.

ip.dst == 192.168.3.7 || ip.src == 192.168.3.7


pptp handshake & ftpdownload & pptp exit.pcap [Wireshark 1.8.5 (SVN Rev 47350 from /trunk-1.8)]
File
Edit
View
Go
Capture
Analyze
Statistics
Telephony
Tools
Internals
Help
Expression...
Clear
Apply
Save

Filter: ip.dst == 192.168.3.7 || ip.src == 192.168.3.7

```
No.     Time        Source          Destination      Protocol    Length Info
     17 0.173039    192.168.3.5     192.168.3.7      TCP             62 instantia > pptp [SYN] Seq=0 Win=65535 L
     18 0.173540    192.168.3.7     192.168.3.5      TCP             62 pptp > instantia [SYN, ACK] Seq=0 Ack=1
     19 0.173556    192.168.3.5     192.168.3.7      PPTP           210 Start-Control-Connection-Request
     20 0.173709    192.168.3.7     192.168.3.5      TCP             60 pptp > instantia [ACK] Seq=1 Ack=157 Win
     21 0.174604    192.168.3.7     192.168.3.5      PPTP           210 Start-Control-Connection-Reply
     22 0.174617    192.168.3.5     192.168.3.7      PPTP           222 Outgoing-Call-Request
     23 0.175250    192.168.3.7     192.168.3.5      PPTP            86 Outgoing-Call-Reply
     24 0.177283    192.168.3.5     192.168.3.7      PPTP            78 Set-Link-Info
     25 0.178246    192.168.3.7     192.168.3.5      PPP LCP         75 Configuration Request
     26 0.178786    192.168.3.5     192.168.3.7      PPP LCP         75 Configuration Request
     27 0.178900    192.168.3.5     192.168.3.7      PPP LCP         75 Configuration Ack
     28 0.179058    192.168.3.7     192.168.3.5      PPP LCP         61 Configuration Reject
     29 0.179158    192.168.3.5     192.168.3.7      PPP LCP         72 Configuration Request
     30 0.179557    192.168.3.7     192.168.3.5      PPP LCP         72 Configuration Ack
     31 0.179624    192.168.3.7     192.168.3.5      PPP LCP         60 Echo Request
     32 0.179697    192.168.3.7     192.168.3.5      PPP CHAP        74 Challenge (NAME='pptpd', VALUE=0x1f11159
     33 0.179862    192.168.3.5     192.168.3.7      PPP LCP         70 Identification
     34 0.179967    192.168.3.5     192.168.3.7      PPP LCP         69 Identification
     35 0.180085    192.168.3.5     192.168.3.7      PPP LCP         56 Echo Reply
     36 0.180347    192.168.3.5     192.168.3.7      PPP CHAP       110 Response (NAME='vpnuser2', VALUE=0x4b872
     37 0.181019    192.168.3.7     192.168.3.5      PPP CHAP       115 Success (MESSAGE='S=D570165CF0532E447095
     38 0.181030    192.168.3.7     192.168.3.5      PPP CCP         60 Configuration Request
     39 0.182131    192.168.3.5     192.168.3.7      PPP CCP         62 Configuration Request
     40 0.182253    192.168.3.5     192.168.3.7      PPP IPCP        82 Configuration Request
     41 0.182361    192.168.3.5     192.168.3.7      PPP CCP         58 Configuration Ack
     42 0.182421    192.168.3.7     192.168.3.5      PPP CCP         62 Configuration Nak
     43 0.182537    192.168.3.7     192.168.3.5      PPP IPCP        60 Termination Ack
     44 0.182603    192.168.3.5     192.168.3.7      PPP CCP         62 Configuration Request
     45 0.182875    192.168.3.7     192.168.3.5      PPP CCP         62 Configuration Ack
```

```
⊞ Frame 17: 62 bytes on wire (496 bits), 62 bytes captured (496 bits)
⊞ Ethernet II, Src: CadmusCo_15:94:e7 (08:00:27:15:94:e7), Dst: CadmusCo_92:f5:4a (08:00:27:92:f5:4a)
⊞ Internet Protocol Version 4, Src: 192.168.3.5 (192.168.3.5), Dst: 192.168.3.7 (192.168.3.7)
⊟ Transmission Control Protocol, Src Port: instantia (1240), Dst Port: pptp (1723), Seq: 0, Len: 0
    Source port: instantia (1240)
    Destination port: pptp (1723)
    [Stream index: 0]
    Sequence number: 0    (relative sequence number)
    Header length: 28 bytes
  ⊞ Flags: 0x002 (SYN)
    Window size value: 65535
    [Calculated window size: 65535]
  ⊞ Checksum: 0xbc48 [validation disabled]
  ⊞ Options: (8 bytes), Maximum segment size, No-Operation (NOP), No-Operation (NOP), SACK permitted
```

```
0000  08 00 27 92 f5 4a 08 00  27 15 94 e7 08 00 45 00   ..'..J.. '.....E.
0010  00 30 6f ed 40 00 80 06  03 7e c0 a8 03 05 c0 a8   .0o.@... .~......
0020  03 07 04 d8 06 bb 44 9a  ef 4c 00 00 00 00 70 02   .......D. .L....p.
0030  ff ff bc 48 00 00 02 04  05 b4 01 01 04 02         ...H.... ......
```

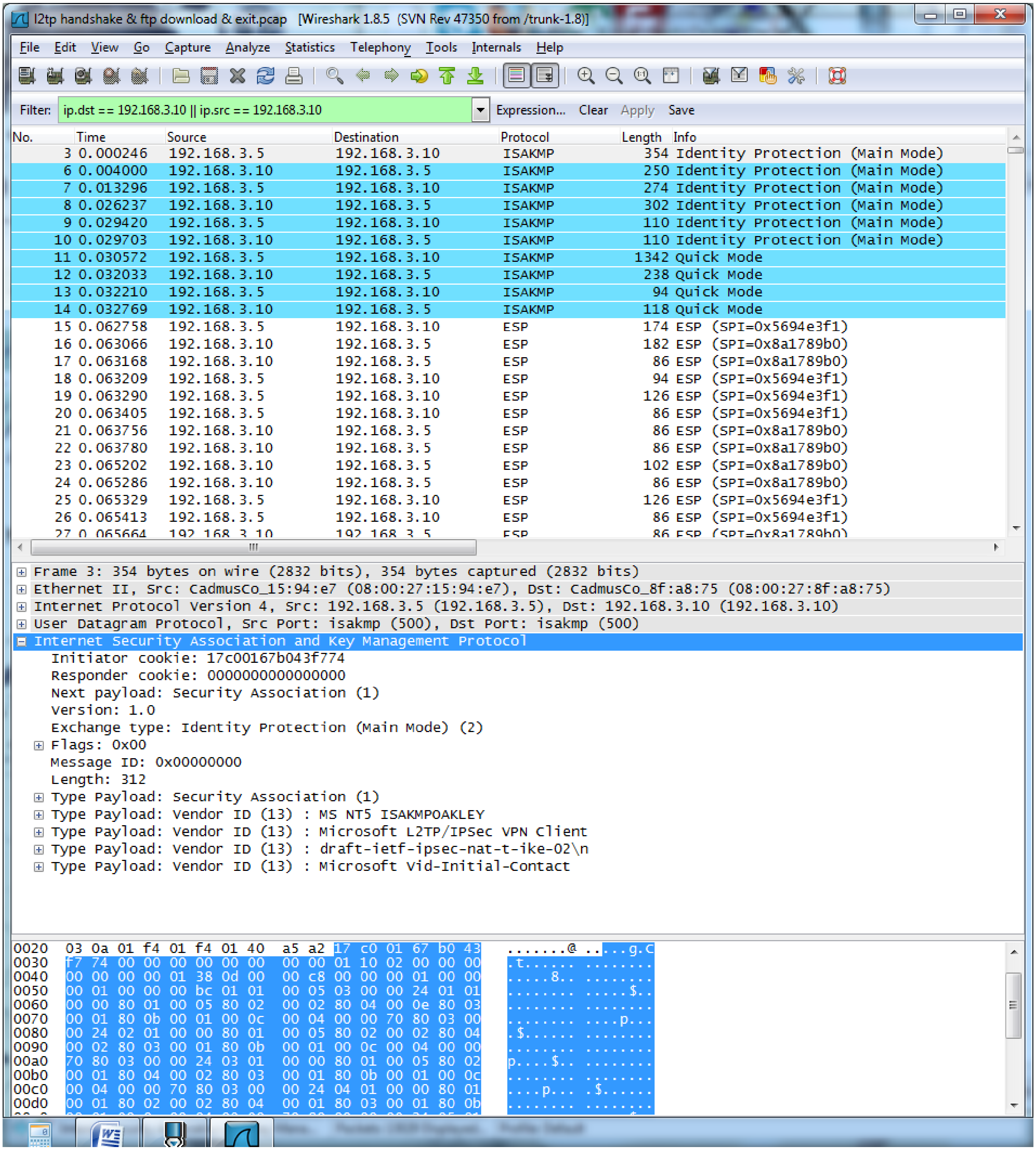
Transmission Control Protocol (tcp), 28 bytes | Packets: 13191 Displayed: 1... | Profile: Default

Picture 31 Wireshark PPTP

As anticipated, the first observed process is the tunnel's initialization and the node authentication which is performed by the PPTP tunnel's control socket over the TCP protocol. After that, the PPP protocol will send and receive compressed datagrams from/to the two nodes over the GRE protocol. The produced traffic is generated by the file to be send over the FTP inside the VPN tunnel. At the end, the control socket is called again to terminate the session between the two nodes.

L2TP/IPsec

Similarly for the L2TP/IPsec captured traffic, the same filter with the PPTP is applied with the one node's IP address this time to be 192.168.3.10.



Filter: ip.dst == 192.168.3.10 || ip.src == 192.168.3.10

```
No.  Time       Source         Destination    Protocol  Length Info
   3 0.000246   192.168.3.5    192.168.3.10   ISAKMP      354 Identity Protection (Main Mode)
   6 0.004000   192.168.3.10   192.168.3.5    ISAKMP      250 Identity Protection (Main Mode)
   7 0.013296   192.168.3.5    192.168.3.10   ISAKMP      274 Identity Protection (Main Mode)
   8 0.026237   192.168.3.10   192.168.3.5    ISAKMP      302 Identity Protection (Main Mode)
   9 0.029420   192.168.3.5    192.168.3.10   ISAKMP      110 Identity Protection (Main Mode)
  10 0.029703   192.168.3.10   192.168.3.5    ISAKMP      110 Identity Protection (Main Mode)
  11 0.030572   192.168.3.5    192.168.3.10   ISAKMP     1342 Quick Mode
  12 0.032033   192.168.3.10   192.168.3.5    ISAKMP      238 Quick Mode
  13 0.032210   192.168.3.5    192.168.3.10   ISAKMP       94 Quick Mode
  14 0.032769   192.168.3.10   192.168.3.5    ISAKMP      118 Quick Mode
  15 0.062758   192.168.3.5    192.168.3.10   ESP         174 ESP (SPI=0x5694e3f1)
  16 0.063066   192.168.3.10   192.168.3.5    ESP         182 ESP (SPI=0x8a1789b0)
  17 0.063168   192.168.3.10   192.168.3.5    ESP          86 ESP (SPI=0x8a1789b0)
  18 0.063209   192.168.3.5    192.168.3.10   ESP          94 ESP (SPI=0x5694e3f1)
  19 0.063290   192.168.3.5    192.168.3.10   ESP         126 ESP (SPI=0x5694e3f1)
  20 0.063405   192.168.3.5    192.168.3.10   ESP          86 ESP (SPI=0x5694e3f1)
  21 0.063756   192.168.3.10   192.168.3.5    ESP          86 ESP (SPI=0x8a1789b0)
  22 0.063780   192.168.3.10   192.168.3.5    ESP          86 ESP (SPI=0x8a1789b0)
  23 0.065202   192.168.3.10   192.168.3.5    ESP         102 ESP (SPI=0x8a1789b0)
  24 0.065286   192.168.3.10   192.168.3.5    ESP          86 ESP (SPI=0x8a1789b0)
  25 0.065329   192.168.3.5    192.168.3.10   ESP         126 ESP (SPI=0x5694e3f1)
  26 0.065413   192.168.3.5    192.168.3.10   ESP          86 ESP (SPI=0x5694e3f1)
  27 0.065664   192.168.3.10   192.168.3.5    ESP          86 ESP (SPI=0x8a1789b0)
```

```
⊞ Frame 3: 354 bytes on wire (2832 bits), 354 bytes captured (2832 bits)
⊞ Ethernet II, Src: CadmusCo_15:94:e7 (08:00:27:15:94:e7), Dst: CadmusCo_8f:a8:75 (08:00:27:8f:a8:75)
⊞ Internet Protocol Version 4, Src: 192.168.3.5 (192.168.3.5), Dst: 192.168.3.10 (192.168.3.10)
⊞ User Datagram Protocol, Src Port: isakmp (500), Dst Port: isakmp (500)
⊟ Internet Security Association and Key Management Protocol
    Initiator cookie: 17c00167b043f774
    Responder cookie: 0000000000000000
    Next payload: Security Association (1)
    Version: 1.0
    Exchange type: Identity Protection (Main Mode) (2)
  ⊞ Flags: 0x00
    Message ID: 0x00000000
    Length: 312
  ⊞ Type Payload: Security Association (1)
  ⊞ Type Payload: Vendor ID (13) : MS NT5 ISAKMPOAKLEY
  ⊞ Type Payload: Vendor ID (13) : Microsoft L2TP/IPSec VPN Client
  ⊞ Type Payload: Vendor ID (13) : draft-ietf-ipsec-nat-t-ike-02\n
  ⊞ Type Payload: Vendor ID (13) : Microsoft Vid-Initial-Contact
```

```
0020  03 0a 01 f4 01 f4 01 40  a5 a2 17 c0 01 67 b0 43   .......@ .....g.C
0030  f7 74 00 00 00 00 00 00  00 00 01 10 02 00 00 00   .t...... ........
0040  00 00 00 00 01 38 0d 00  00 c8 00 00 00 01 00 00   .....8.. ........
0050  00 01 00 00 00 bc 01 01  00 05 03 00 00 24 01 01   ........ .....$..
0060  00 00 80 01 00 05 80 02  00 02 80 04 00 0e 80 03   ........ ........
0070  00 01 80 0b 00 01 00 0c  00 04 00 00 70 80 03 00   ........ ....p...
0080  00 24 02 01 00 00 80 01  00 05 80 02 00 02 80 04   .$...... ........
0090  00 02 80 03 00 01 80 0b  00 01 00 0c 00 04 00 00   ........ ........
00a0  70 80 03 00 00 24 03 01  00 00 80 01 00 05 80 02   p....$.. ........
00b0  00 01 80 04 00 02 80 03  00 01 80 0b 00 01 00 0c   ........ ........
00c0  00 04 00 00 70 80 03 00  00 24 04 01 00 00 80 01   ....p... .$......
00d0  00 01 80 02 00 02 80 04  00 01 80 03 00 01 80 0b   ........ ........
```

Picture 32 Wireshark L2TP

Initially, the ISAKMP application protocol over the UDP transmission protocol is to establish and set up the VPN tunnel and to authenticate the one of the two nodes by the other. Next, encrypted and not compressed packets as before, of the ESP transmission type are to carry the traffic and the control flow messages from the one to the other node. Finally, as the file over FTP is transferred, the receiving node is disconnected and the ISAKMP terminates the VPN tunnel.

Conclusions

PPTP and L2TP/IPsec are both strict VPN tunnel protocols. In other words, the two protocols are not related in any form with tasks outside the tunnel, its creation and destruction and its control flow. For instance, they are not related with the user's registration process on the service or with how one host should discover another. As a result, they are usually combined with other supportive services or infrastructure in order to demonstrate a complete and universal behavior.

Authentication

In authentication, both protocols make use of similar processes while the L2TP makes use of a more secure approach.

Encryption

PPTP does not make use of encryption of any form but uses compression as opposed to L2TP/IPsec which carries fully encrypted traffic.

Traffic routing and encapsulation

Under the transmission network layer, the PPTP protocol makes use of TCP for the tunnel's control and GRE for carrying the VPN traffic. The L2TP/IPsec protocol uses UDP and IPsec over UDP (ESP). As a result, in order for the traffic to be carried over a network, for instance the Internet, all the routing nodes in between the two VPN tunnel ends should allow the above transmission protocols to be transferred. Another noteworthy fact is that PPTP makes use of two connection sockets, one for the control and another for the data carriage as opposed to the L2TP/IPsec which uses a single socket to carry both control co-ordination messages and data.

Address attribution

Another good feature for the two protocols is that they do not nessesairily need to use a DHCP service in order to attribute IP addresses to the virtual hosts as the protocols have this functionality integrated. However, if a DHCP service is not used, the virtual IP addresses are appointed as the users are logged in and by operating in such a manner there is no correlation between an IP address and a user's device. In other words, each time a user logs in the service, he may get a different IP address depending on the number of users which were connected before him.

Architecture

Both examined protocols are tightly related towards the Operating System while their set up is heavily related with the latter's proper configuration. It is a process which may be extremely demanding and complicated and, in general terms, may be performed only from admins. Their set up process includes the protocol's installation as a service, to configure all the closely related services to co-operate like the DHCP and DNS, to allow the OS to forward traffic and to configure the host's firewall and all the intermediate firewalls to allow the VPN traffic to be carried. The client's setup is easier where a client's device may even have an integrated VPN client, however, the overall centralized setup which has to be configured prevents the users from fully experiencing the service without the presence of a ready and set VPN server.

Standardization

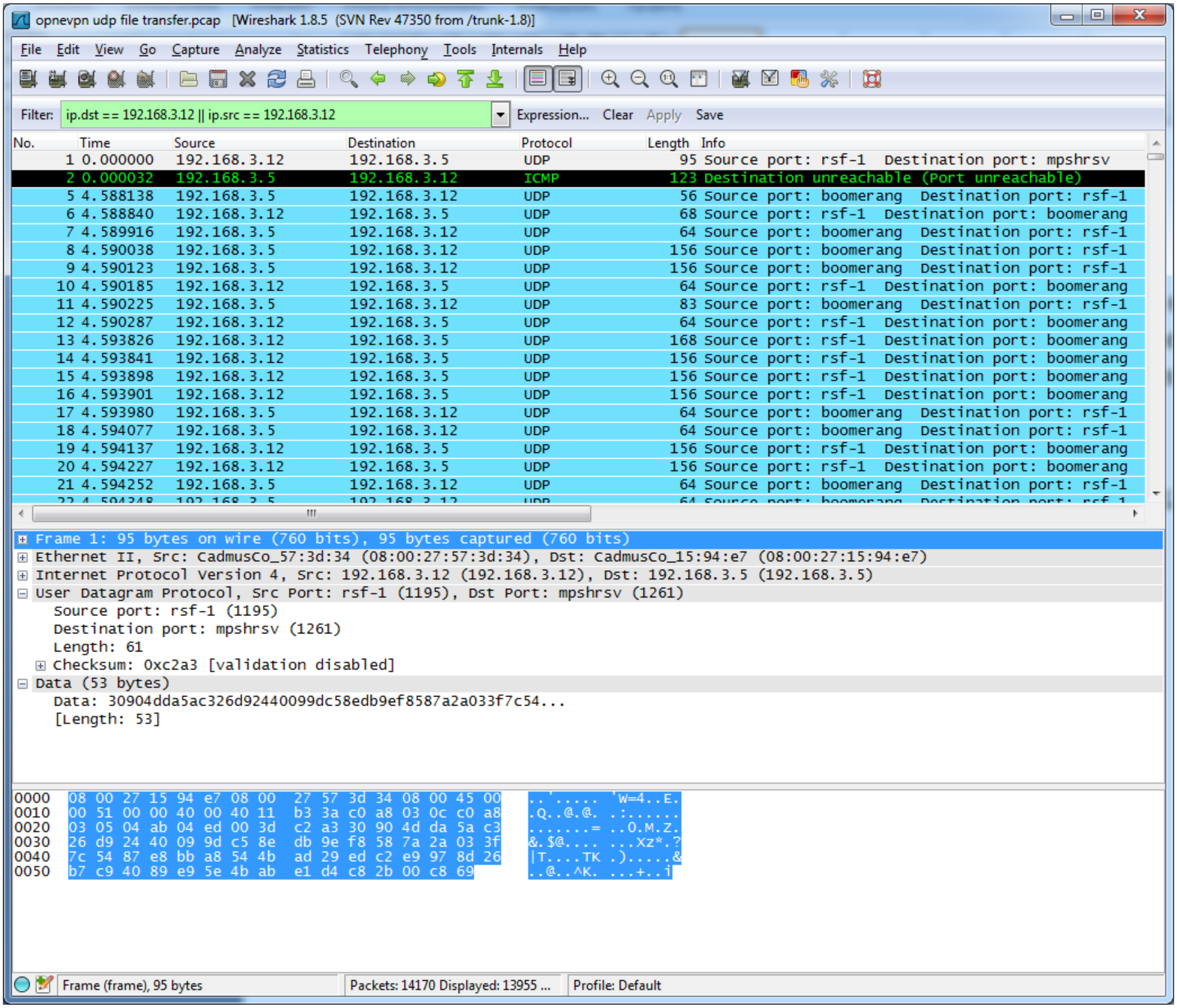
Due to protocols' standardized nature and the usage of specialized and scarcely used transmission protocols as GRE and ESP, the produced VPN streams are highly discoverable both from a local LAN admin as from all the network intermediates. Consequently, they are **easy to be policed over the Internet** while a more specialized router is required with a NAT capable to forward these transmission protocols.

OpenVPN

OpenVPN's creation was highly related to specific factors which had to do with the classic protocols and their inefficiency to satisfy certain demands. As introduced, the classic protocols demonstrate extensively standardized features, formed from specialized headers and specific transmission protocols which, on their turn, make the produced VPN traffic discoverable and easily policed or blocked by the network traffic analyzers and firewalls. In contrast, OpenVPN makes use of an SSL/TLS communication scheme for encryption with the option to use either UDP or TCP as a transmission protocol to carry the VPN traffic while it holds all the co-ordination and data messages for each connection inside the same transmission stream, all of which, have the ultimate effect to properly protect the stream's privacy over the network and establish integrity and confidentiality. Moreover, regarding the OS, OpenVPN does not require a tight coupling of the VPN service with the OS or a demanding configuration process as opposed to the formerly introduced classic VPN protocols. Meaning that from the one hand, it is easier to be installed and from the other, it limits the need to extensively configure the OS. Finally, as OpenVPN is capable to transfer data both under the UDP and TCP streams, network traffic was captured for both of the cases.

Open VPN/UDP traffic

Using the same filter as the precious cases with the IP address to be this time 192.168.3.12.


opnevpn udp file transfer.pcap [Wireshark 1.8.5 (SVN Rev 47350 from /trunk-1.8)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Expression... Clear Apply Save

Filter: ip.dst == 192.168.3.12 || ip.src == 192.168.3.12

```
No.   Time      Source         Destination    Protocol  Length  Info
    1 0.000000  192.168.3.12   192.168.3.5    UDP           95 Source port: rsf-1  Destination port: mpshrsv
    2 0.000032  192.168.3.5    192.168.3.12   ICMP         123 Destination unreachable (Port unreachable)
    5 4.588138  192.168.3.5    192.168.3.12   UDP           56 Source port: boomerang  Destination port: rsf-1
    6 4.588840  192.168.3.12   192.168.3.5    UDP           68 Source port: rsf-1  Destination port: boomerang
    7 4.589916  192.168.3.5    192.168.3.12   UDP           64 Source port: boomerang  Destination port: rsf-1
    8 4.590038  192.168.3.5    192.168.3.12   UDP          156 Source port: boomerang  Destination port: rsf-1
    9 4.590123  192.168.3.5    192.168.3.12   UDP          156 Source port: boomerang  Destination port: rsf-1
   10 4.590185  192.168.3.12   192.168.3.5    UDP           64 Source port: rsf-1  Destination port: boomerang
   11 4.590225  192.168.3.5    192.168.3.12   UDP           83 Source port: boomerang  Destination port: rsf-1
   12 4.590287  192.168.3.12   192.168.3.5    UDP           64 Source port: rsf-1  Destination port: boomerang
   13 4.593826  192.168.3.12   192.168.3.5    UDP          168 Source port: rsf-1  Destination port: boomerang
   14 4.593841  192.168.3.12   192.168.3.5    UDP          156 Source port: rsf-1  Destination port: boomerang
   15 4.593898  192.168.3.12   192.168.3.5    UDP          156 Source port: rsf-1  Destination port: boomerang
   16 4.593901  192.168.3.12   192.168.3.5    UDP          156 Source port: rsf-1  Destination port: boomerang
   17 4.593980  192.168.3.5    192.168.3.12   UDP           64 Source port: boomerang  Destination port: rsf-1
   18 4.594077  192.168.3.5    192.168.3.12   UDP           64 Source port: boomerang  Destination port: rsf-1
   19 4.594137  192.168.3.12   192.168.3.5    UDP          156 Source port: rsf-1  Destination port: boomerang
   20 4.594227  192.168.3.12   192.168.3.5    UDP          156 Source port: rsf-1  Destination port: boomerang
   21 4.594252  192.168.3.5    192.168.3.12   UDP           64 Source port: boomerang  Destination port: rsf-1
   22 4.594348  192.168.3.5    192.168.3.12   UDP           64 Source port: boomerang  Destination port: rsf-1
```

```
⊞ Frame 1: 95 bytes on wire (760 bits), 95 bytes captured (760 bits)
⊞ Ethernet II, Src: CadmusCo_57:3d:34 (08:00:27:57:3d:34), Dst: CadmusCo_15:94:e7 (08:00:27:15:94:e7)
⊞ Internet Protocol Version 4, Src: 192.168.3.12 (192.168.3.12), Dst: 192.168.3.5 (192.168.3.5)
⊟ User Datagram Protocol, Src Port: rsf-1 (1195), Dst Port: mpshrsv (1261)
    Source port: rsf-1 (1195)
    Destination port: mpshrsv (1261)
    Length: 61
  ⊞ Checksum: 0xc2a3 [validation disabled]
⊟ Data (53 bytes)
    Data: 30904dda5ac326d92440099dc58edb9ef8587a2a033f7c54...
    [Length: 53]
```

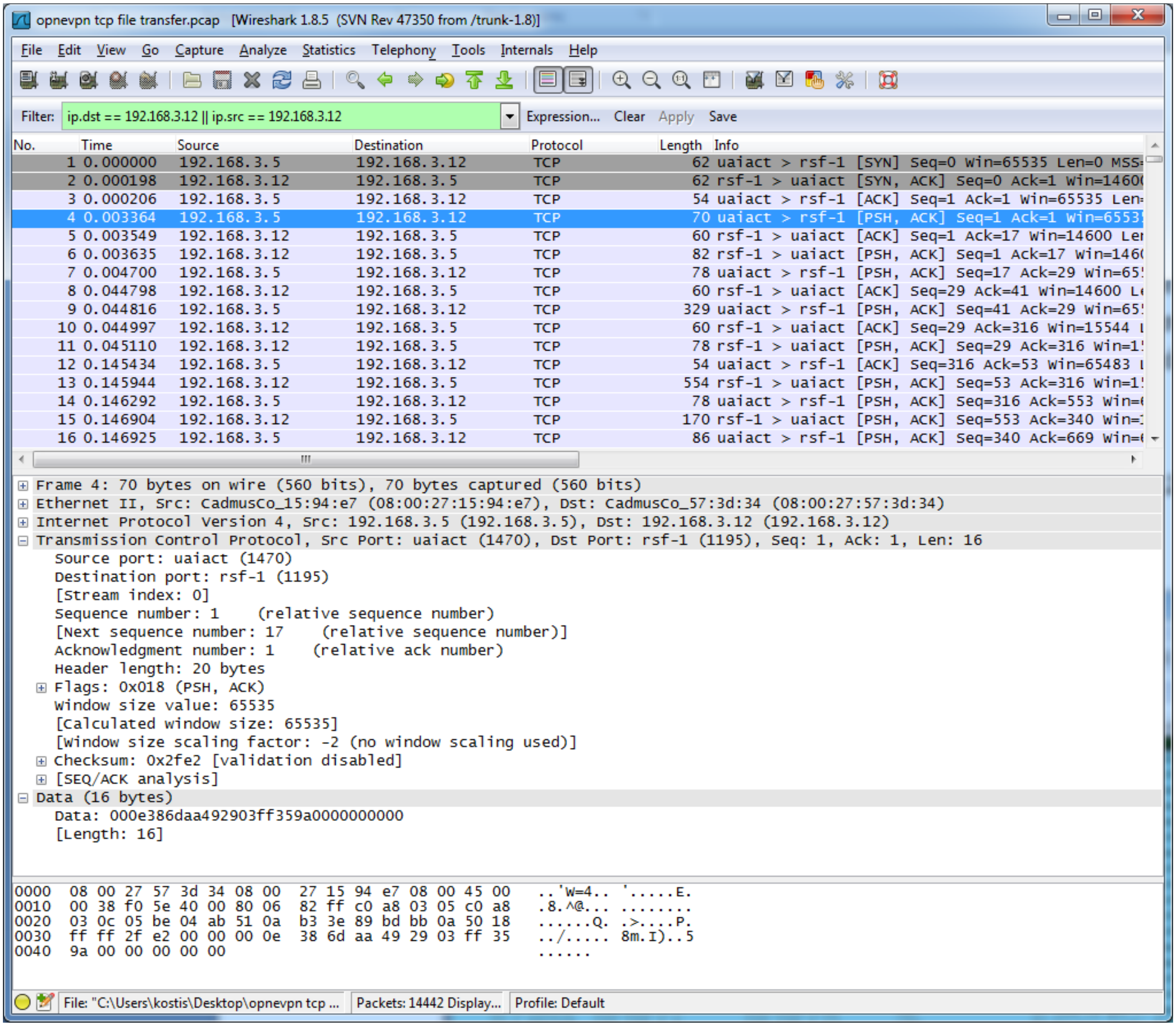
```
0000  08 00 27 15 94 e7 08 00  27 57 3d 34 08 00 45 00   ..'.....  'W=4..E.
0010  00 51 00 00 40 00 40 11  b3 3a c0 a8 03 0c c0 a8   .Q..@.@.  .:......
0020  03 05 04 ab 04 ed 00 3d  c2 a3 30 90 4d da 5a c3   .......=  ..0.M.Z.
0030  26 d9 24 40 09 9d c5 8e  db 9e f8 58 7a 2a 03 3f   &.$@....  ...Xz*.?
0040  7c 54 87 e8 bb a8 54 4b  ad 29 ed c2 e9 97 8d 26   |T....TK  .).....&
0050  b7 c9 40 89 e9 5e 4b ab  e1 d4 c8 2b 00 c8 69      ..@..^K.  ...+..i
```

Frame (frame), 95 bytes | Packets: 14170 Displayed: 13955 ... | Profile: Default

Picture 33 Wireshark OpenVPN UDP

In the above picture, we may observe the produced traffic from/to the OpenVPN server. What may be observed at a glance is that **the whole VPN traffic is fully encrypted and encapsulated under UDP packets while there are no traces of authentication processes or of an encapsulated packet transfer**. A network analyzer upon observing this stream may not be aware that this is a VPN stream if not provided with further information apart from the stream itself!

Open VPN/TCP traffic


opnevpn tcp file transfer.pcap [Wireshark 1.8.5 (SVN Rev 47350 from /trunk-1.8)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Expression... Clear Apply Save
File: "C:\Users\kostis\Desktop\opnevpn tcp ...
Packets: 14442 Display...
Profile: Default

Filter: ip.dst == 192.168.3.12 || ip.src == 192.168.3.12

```
No.   Time      Source         Destination    Protocol  Length Info
    1 0.000000  192.168.3.5    192.168.3.12   TCP           62 uaiact > rsf-1 [SYN] Seq=0 Win=65535 Len=0 MSS=
    2 0.000198  192.168.3.12   192.168.3.5    TCP           62 rsf-1 > uaiact [SYN, ACK] Seq=0 Ack=1 Win=1460
    3 0.000206  192.168.3.5    192.168.3.12   TCP           54 uaiact > rsf-1 [ACK] Seq=1 Ack=1 Win=65535 Len=
    4 0.003364  192.168.3.5    192.168.3.12   TCP           70 uaiact > rsf-1 [PSH, ACK] Seq=1 Ack=1 Win=6553
    5 0.003549  192.168.3.12   192.168.3.5    TCP           60 rsf-1 > uaiact [ACK] Seq=1 Ack=17 Win=14600 Len
    6 0.003635  192.168.3.12   192.168.3.5    TCP           82 rsf-1 > uaiact [PSH, ACK] Seq=1 Ack=17 Win=146
    7 0.004700  192.168.3.5    192.168.3.12   TCP           78 uaiact > rsf-1 [PSH, ACK] Seq=17 Ack=29 Win=65
    8 0.044798  192.168.3.12   192.168.3.5    TCP           60 rsf-1 > uaiact [ACK] Seq=29 Ack=41 Win=14600 L
    9 0.044816  192.168.3.5    192.168.3.12   TCP          329 uaiact > rsf-1 [PSH, ACK] Seq=41 Ack=29 Win=65
   10 0.044997  192.168.3.12   192.168.3.5    TCP           60 rsf-1 > uaiact [ACK] Seq=29 Ack=316 Win=15544
   11 0.045110  192.168.3.12   192.168.3.5    TCP           78 rsf-1 > uaiact [PSH, ACK] Seq=29 Ack=316 Win=1
   12 0.145434  192.168.3.5    192.168.3.12   TCP           54 uaiact > rsf-1 [ACK] Seq=316 Ack=53 Win=65483
   13 0.145944  192.168.3.12   192.168.3.5    TCP          554 rsf-1 > uaiact [PSH, ACK] Seq=53 Ack=316 Win=1
   14 0.146292  192.168.3.5    192.168.3.12   TCP           78 uaiact > rsf-1 [PSH, ACK] Seq=316 Ack=553 Win=
   15 0.146904  192.168.3.12   192.168.3.5    TCP          170 rsf-1 > uaiact [PSH, ACK] Seq=553 Ack=340 Win=
   16 0.146925  192.168.3.5    192.168.3.12   TCP           86 uaiact > rsf-1 [PSH, ACK] Seq=340 Ack=669 Win=
```

```
⊞ Frame 4: 70 bytes on wire (560 bits), 70 bytes captured (560 bits)
⊞ Ethernet II, Src: CadmusCo_15:94:e7 (08:00:27:15:94:e7), Dst: CadmusCo_57:3d:34 (08:00:27:57:3d:34)
⊞ Internet Protocol Version 4, Src: 192.168.3.5 (192.168.3.5), Dst: 192.168.3.12 (192.168.3.12)
⊟ Transmission Control Protocol, Src Port: uaiact (1470), Dst Port: rsf-1 (1195), Seq: 1, Ack: 1, Len: 16
    Source port: uaiact (1470)
    Destination port: rsf-1 (1195)
    [Stream index: 0]
    Sequence number: 1    (relative sequence number)
    [Next sequence number: 17    (relative sequence number)]
    Acknowledgment number: 1    (relative ack number)
    Header length: 20 bytes
  ⊞ Flags: 0x018 (PSH, ACK)
    Window size value: 65535
    [Calculated window size: 65535]
    [Window size scaling factor: -2 (no window scaling used)]
  ⊞ Checksum: 0x2fe2 [validation disabled]
  ⊞ [SEQ/ACK analysis]
⊟ Data (16 bytes)
    Data: 000e386daa492903ff359a0000000000
    [Length: 16]
```

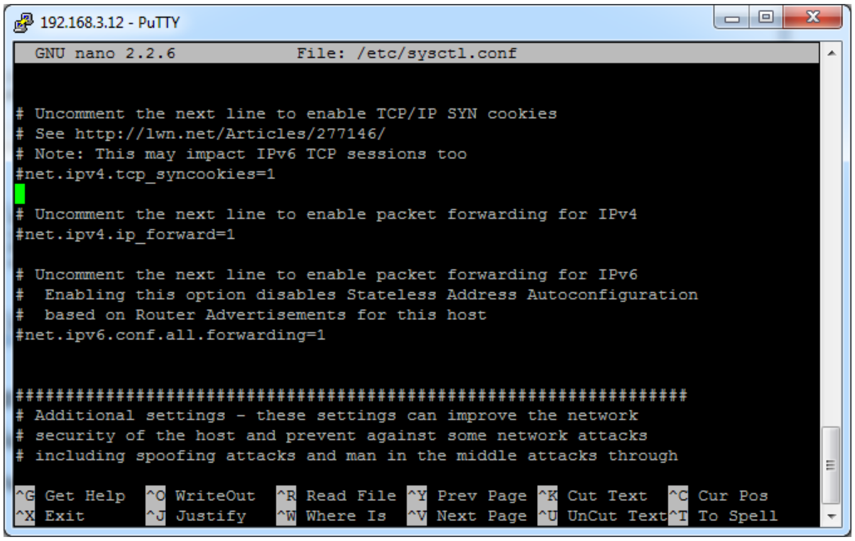
```
0000  08 00 27 57 3d 34 08 00  27 15 94 e7 08 00 45 00   ..'W=4.. '.....E.
0010  00 38 f0 5e 40 00 80 06  82 ff c0 a8 03 05 c0 a8   .8.^@... ........
0020  03 0c 05 be 04 ab 51 0a  b3 3e 89 bd bb 0a 50 18   ......Q. .>....P.
0030  ff ff 2f e2 00 00 00 0e  38 6d aa 49 29 03 ff 35   ../..... 8m.I)..5
0040  9a 00 00 00 00 00                                  ......
```

Picture 34 Wireshark OpenVPN TCP

Accordingly for the TCP traffic, it is noted that the VPN traffic may not be easily identified as such as most of the tunnel’s sensitive information are being protected by the produced TCP tunnel.

VPN autonomy towards the OS

In our VPN’s hosting OS, the internal packet routing can be enabled or disabled from the ‘**/etc/sysctl.conf’** file if it is a UNIX based system. As opposed to the classic VPN protocols, OpenVPN does not require the internal system routing functionality to be enabled while the classic ones do. Therefore, in the PPTP and L2TP case the ‘**net.ipv4.ip_forward=1’** line should be uncommented to notify the OS to allow traffic to be forwarded while when hosting OpenVPN, it may only be required in cases where there is the need to connect multiple LANs.



```
  GNU nano 2.2.6              File: /etc/sysctl.conf


# Uncomment the next line to enable TCP/IP SYN cookies
# See http://lwn.net/Articles/277146/
# Note: This may impact IPv6 TCP sessions too
#net.ipv4.tcp_syncookies=1

# Uncomment the next line to enable packet forwarding for IPv4
#net.ipv4.ip_forward=1

# Uncomment the next line to enable packet forwarding for IPv6
#  Enabling this option disables Stateless Address Autoconfiguration
#  based on Router Advertisements for this host
#net.ipv6.conf.all.forwarding=1


###################################################################
# Additional settings - these settings can improve the network
# security of the host and prevent against some network attacks
# including spoofing attacks and man in the middle attacks through

^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page ^U UnCut Text^T To Spell
```

Picture 35 The /etc/sysctl.conf file

3.4 Comparison with Unity Network

Points of interest

- In cases of extreme standardization over VPN protocols, security and privacy are dramatically reduced.
- The traffic in the afore mentioned cases is easily policed and/or blocked.
- Some protocols as PPTP prefer to use separate transmission sockets for the connection's control and data while others as OpenVPN prefer to use a single one to carry both of them.
- Some protocols prefer tight coupling towards the OS while others loose.

It is noteworthy to be reminded that the captured traffic was collected from the VPN's **external scope**. If the traffic had to be captured from the **internal VPN's scope** by using Wireshark in either one of the two exchanging hosts, the collected traffic would have been observed to be as a usual LAN traffic of a file being carried by the FTP application protocol.

Differences between Unity Network and the afore protocols and services

To begin with, the most significant difference between Unity and the afore protocols and solutions is that Unity is not a protocol. In contrast, it is a platform! This means that while the afore protocols are strictly related with the VPN tunnel, Unity offers features that extend above the VPN tunnel like user registration, node discovery and other in order to provide a complete network platform solution from top-to-bottom that may occupy many host-client systems capable to exchange data between them. In addition, Unity is based on a loose coupling with the OS meaning that it does not require an extensive OS configuration, downloading, setting and running the node applications is just enough to get it running. Unity holds track of the connected hosts in the network and appoints a unique address to each one regardless from where it is connected without making use of any external DHCP and DNS services. Similarly with OpenVPN, it uses encrypted UDP streams which are able to provide not only privacy but also confidentiality and integrity for the produced VPN streams. Finally, it demonstrates a more resilient and flexible character as the client nodes or routing nodes (BNs) may be connected and disconnected from the network at any given moment.

OpenVPN vs Unity

Unity shares some similar features with OpenVPN as the tunnel's structure, encryption and privacy. The difference lies in their distribution. More specifically, OpenVPN follows a centralized server logic to serve its host-clients which results in:

- The same server to route and co-ordinate the produced virtual network.
- A closer dependence in hardware, when the hosting system crashes, the whole platform dies.
- A lack of distribution results for the network's capacity to be the central server's capacity.
- A difficulty into extending the network. There is the option to extend the network but only by making use of extra HW or a second VPN server which has to exist on the same LAN. In such a case, each VPN server authenticates the users independently.

In contrast, Unity introduces a distributed logic:

- A different node co-ordinates the network (Tracker) while multiple nodes (BNs) carry the traffic.
- The platform is resilient and decoupled from HW. If a BN dies, only its users are disconnected, the platform logs the dropped users, while the latter ones may reconnect from a different BN.
- Thanks to the distribution logic, bigger virtual networks with more host-clients may be created.
- BNs can be added or removed dynamically from the platform without the constraint to be on the same LAN.

Chapter 4 – currently under translation

Chapter 5 – currently under translation

Chapter 6 – currently under translation

Bibliography

[1] Wikipedia en, Network Address Translation/Drawbacks, Available: http://en.wikipedia.org/wiki/Network_address_translation#Drawbacks

[2] IPv6.com, NAT-IN-DEPTH, Available: http://ipv6.com/articles/nat/NAT-In-Depth.htm

[3] UseIPv6.com, Why not just use Network Address Translation (NAT)?, Available: http://www.useipv6.com/

[4] RFC 2685, Available: http://www.rfc-editor.org/info/rfc2685

[5] Wikipedia en, PPTP Protocol, https://en.wikipedia.org/wiki/Point-to-Point_Tunneling_Protocol

[6] Andrew S. Tanenbaum, David J. Wetherall, 2012, Computer Networks, Pearson Education

[7] RFC 2637, K. Hamzeh, G. Pall, W. Verthein, J. Taarud, W. Little, G. Zorn, 1999, Available: http://www.rfc-editor.org/info/rfc2637

[8] RFC 2661, W. Townsley, A. Valencia, A. Rubens, G. Pall, G. Zorn, B. Palter, 1999, Available: http://www.rfc-editor.org/info/rfc2661

Software and tools

[1] OpenVPN (TUN/TAP adaptor is part of OpenVPN), Available: https://openvpn.net/

[2] Phpseclib, Available: http://phpseclib.sourceforge.net/

[3] Bouncy castle, Available: https://www.bouncycastle.org/

[4] PhpMyAdmin, Available: http://www.phpmyadmin.net/home_page/index.php

[5] Wireshark, Available: http://www.wireshark.org/

[6] VirualBox, Available: https://www.virtualbox.org/

[7] Nmap utilities, Available: http://nmap.org/

Appendix

Further project details

Due to the project's demands, the project's source code, usage examples and the project's news have been moved outside of the report in order to establish freshness as the project is evolving.

Source Code

The source code is available from GitHub by accessing the following urls.

https://github.com/kostiskag/unitynetwork-rednode

https://github.com/kostiskag/unitynetwork-bluenode

https://github.com/kostiskag/unitynetwork-tracker

The project's latest news

The project's latest news will be accessible from the following blog page.

https://kostiskag.wordpress.com/2017/05/25/unity-network/