



PPPoE Implementation Example

[.Fluidity: *Bring your telecoms infrastructure to the future*]

The Computer Hardware Lab

By **Charalampos Kapolonaris (Technical Lead)**
& **Vasilios Koutlas (Software Tester)**

.Fluidity PPPoE Implementation Example

The Addressing Scheme

.FLUIDITY PPPoE EXAMPLE SYNOPSIS			
.Fluidity Devices	External IP Address/External Interface	.Fluidity Device Password	List of Internal Interfaces on each .Fluidity Device
.Fluidity Server	static_ip_from_isp/ppp0	_Flu1dl1ty-S3rv3r-2020	Network 1: eth0
.Fluidity Client	static_ip_from_isp/ppp0	_Flu1dl1ty_Cli3nt	Network 2: eth0

PPPoE VPN TUNNELING ADDRESSING SCHEME					
Tunnel Connections	Server Address	Client Address	Subnet Mask	Network Address	Broadcast Address
Server - Client (VPN NETWORK)	192.168.54.1	192.168.54.2	255.255.255.252	192.168.54.0	192.168.54.3

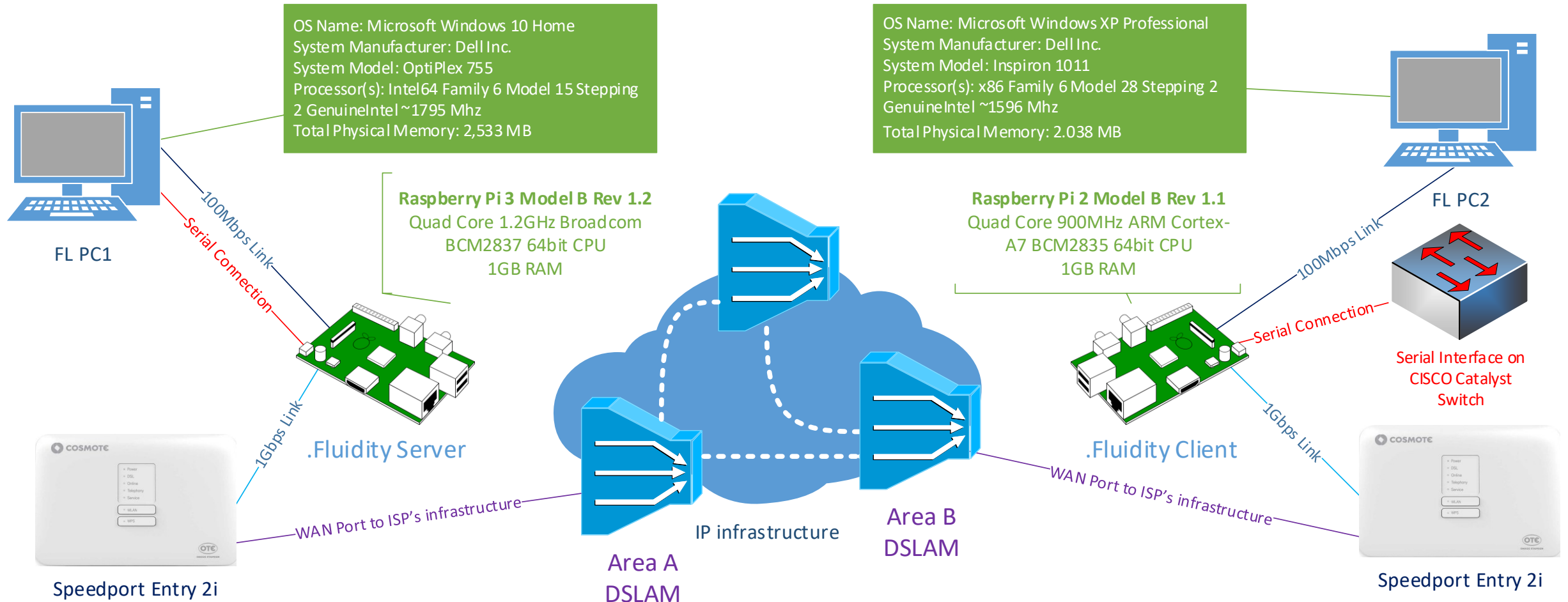
SERIAL TUNNELING SCHEME	
Serial Tunnel to .FL Server	Serial Tunnel from .FL Client 1
FL Server PC_1 -> .Fluidity Server	.Fluidity Client 1 -> Remote Console System (FL 1.1)

.Fluidity PPPoE Implementation Example

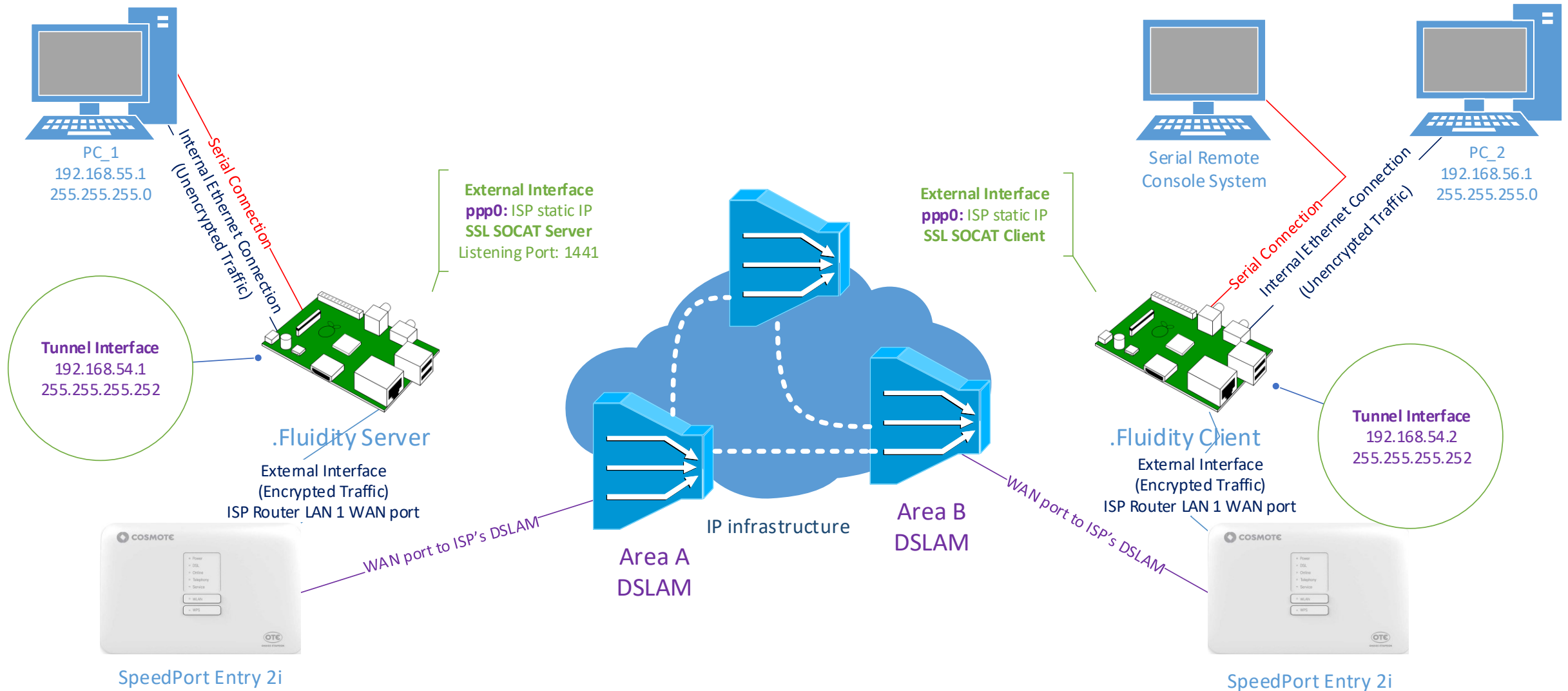
The Networking Scheme in Detail

Devices Attached and the Networking Addressing Scheme in Detail	
Network 1	
CIDR Network Address	192.168.55.0/24
PC_1:	192.168.55.1
.Fluidity Server:	192.168.55.254
Broadcast Address	192.168.55.255
Network 2	
CIDR Network Address	192.168.56.0/24
PC_2:	192.168.56.1
.Fluidity Client 1:	192.168.56.254
Broadcast Address	192.168.56.255

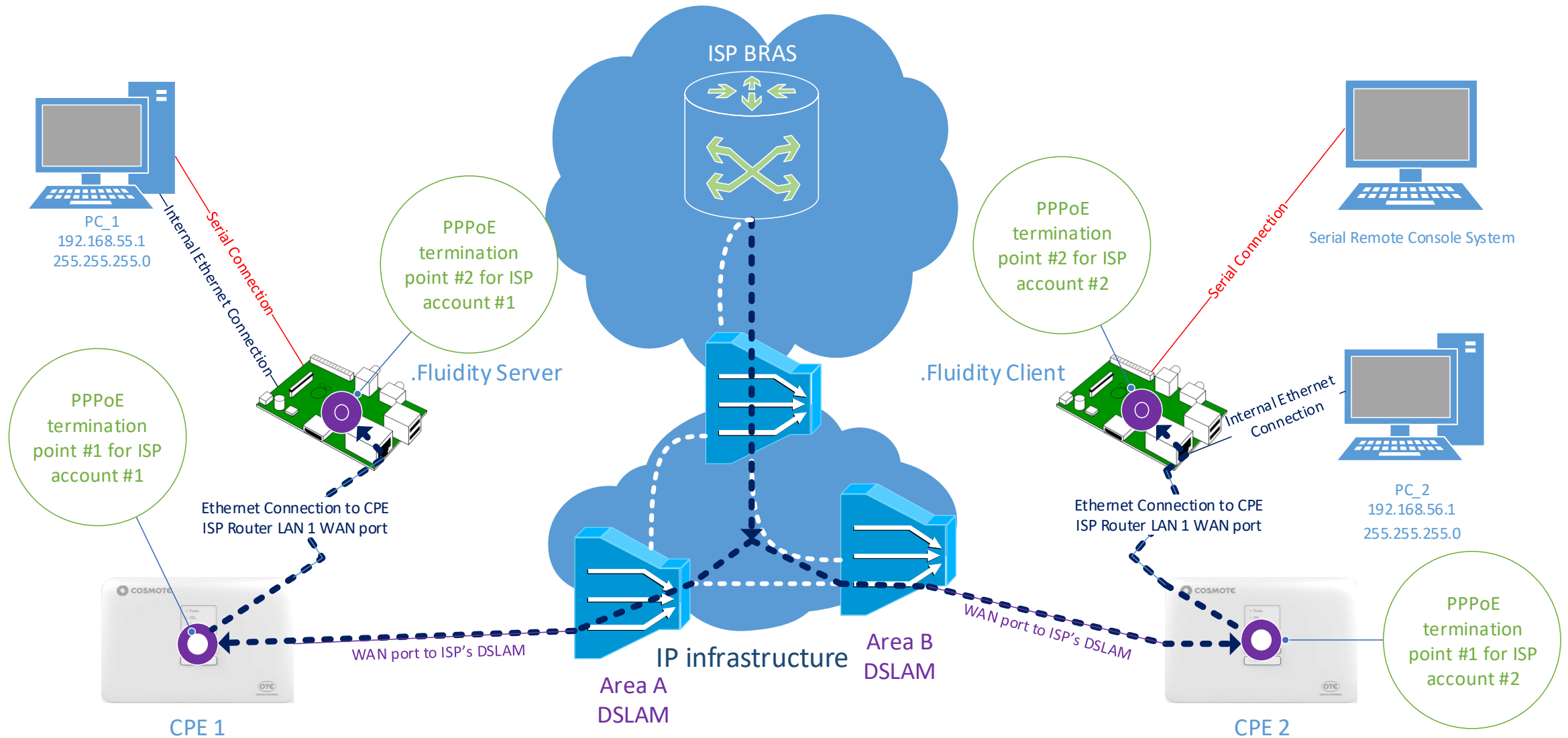
.Fluidity PPPoE Example – The Lab's Technical Specs



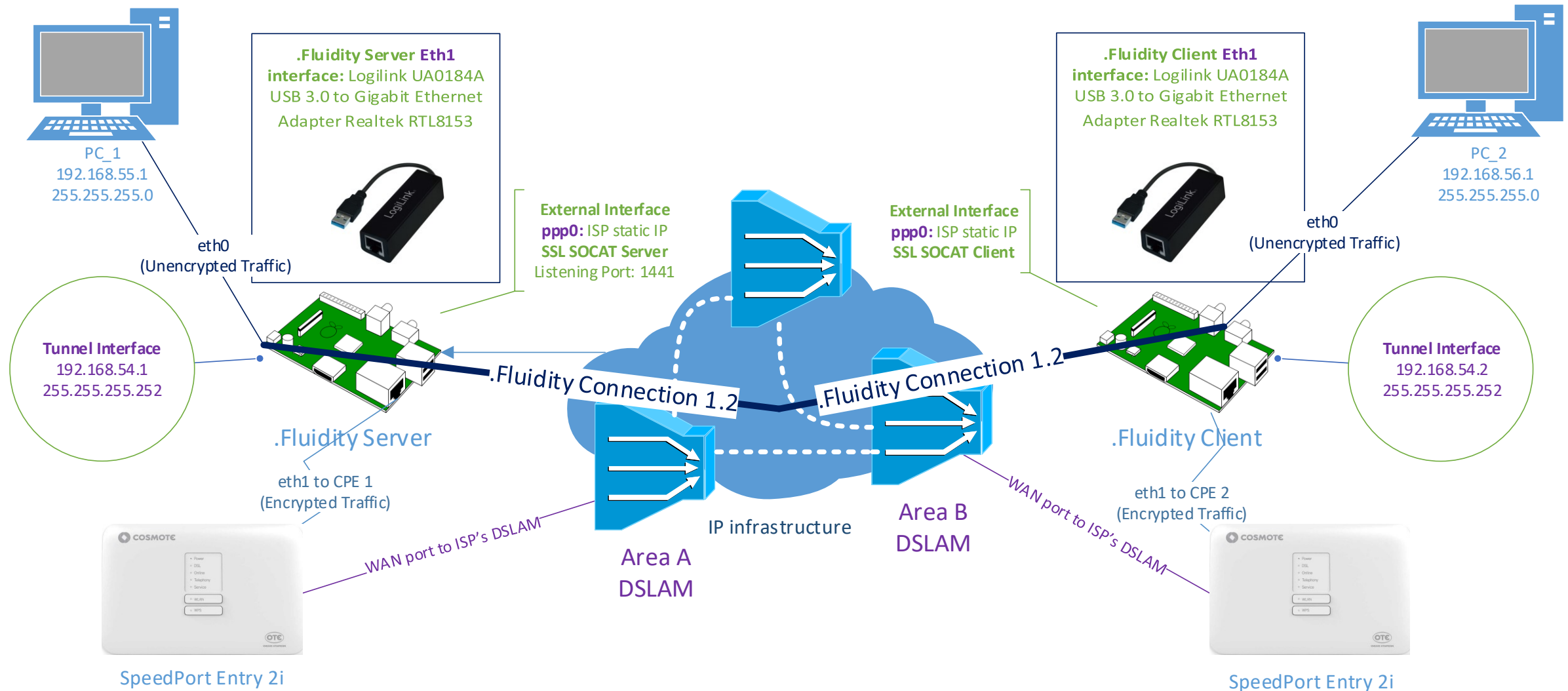
.Fluidity PPPoE Example - Visual Overview



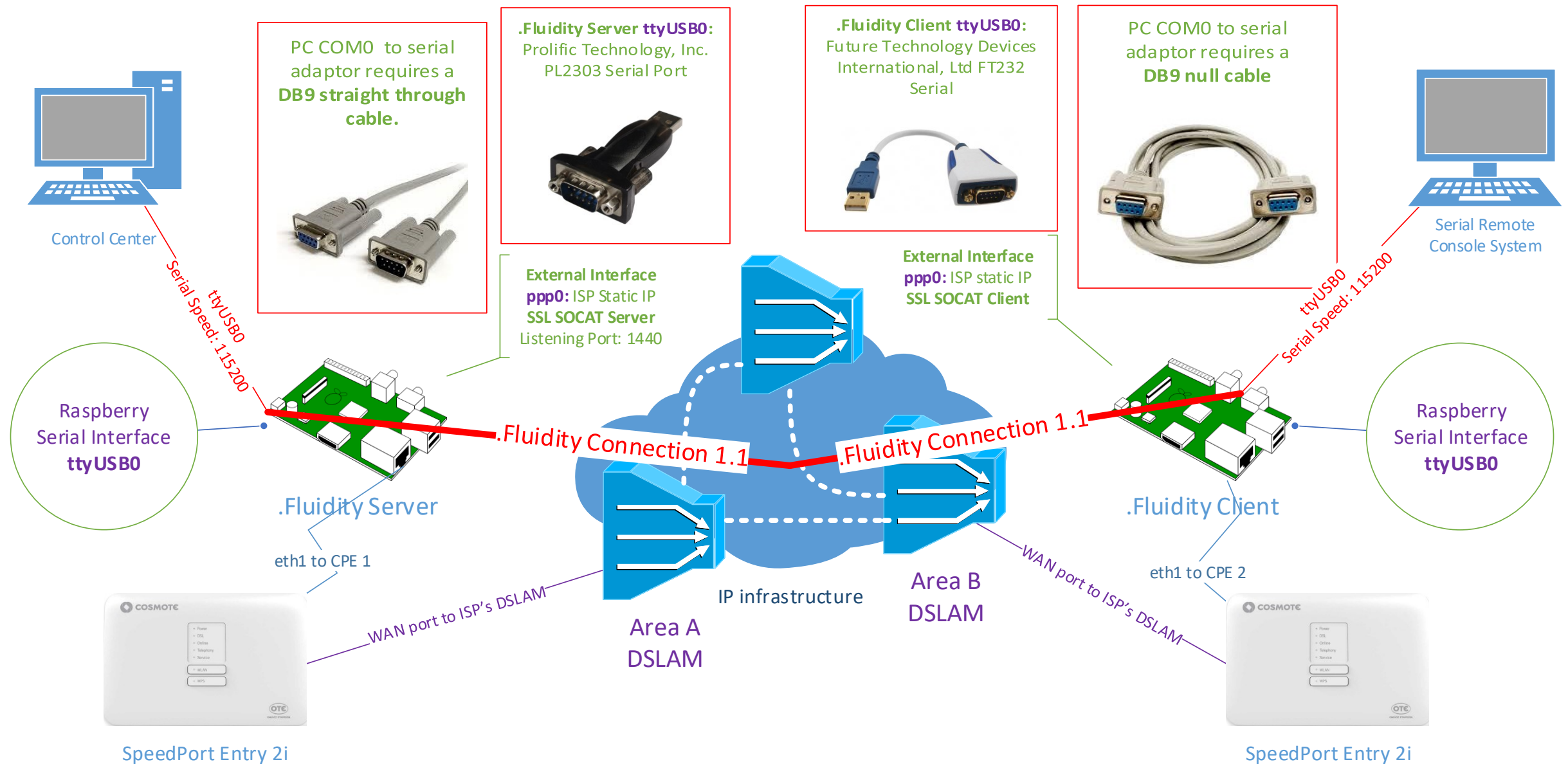
.Fluidity PPPoE Example PPPoE Termination Points



.Fluidity PPPoE Example - VPN Networking Visual Overview



.Fluidity PPPoE Example - Serial Connectivity Visual Overview



.Fluidity PPPoE Example. Install and Setup pppoeconf on each .Fluidity device (leave default settings).

Actions Performed	BASH Commands
Install PPPoE	<code>sudo apt-get install pppoeconf</code>
Setup PPPoE	<code>pppoeconf</code>

OKAY TO MODIFY

If you continue with this program, these configuration files will be modified : /etc/ppp/peers/dsl-provider /etc/network/interfaces and /etc/ppp/*-secrets. Please make sure that you have a backup copy before saying Yes.

Continue with configuration?

<Yes> **<No>**

ENTER USERNAME

Please enter the username which you usually need for the PPP login to your provider in the input box below. If you wish to see the help screen, delete the username and press OK.

username

<Ok>

ENTER PASSWORD

Please enter the password which you usually need for the PPP login to your provider in the input box below.

NOTE: you can see the password in plain text while typing.

<Ok>

LIMITED MSS PROBLEM

Many providers have routers that do not support TCP packets with a MSS higher than 1460. Usually, outgoing packets have this MSS when they go through one real Ethernet link with the default MTU size (1500). Unfortunately, if you are forwarding packets from other hosts (i.e. doing masquerading) the MSS may be increased depending on the packet size and the route to the client hosts, so your client machines won't be able to connect to some sites. There is a solution: the maximum MSS can be limited by pppoe. You can find more details about this issue in the pppoe documentation.

Should pppoe clamp MSS at 1452 bytes?

If unsure, say yes.

(If you still get problems described above, try setting to 1412 in the dsl-provider file.)

<Yes> **<No>**

DONE

Your PPPD is configured now. Would you like to start the connection at boot time?

<Yes> **<No>**

.Fluidity VLAN Implementation Example

Step by step .Fluidity installation and configuration...

Actions Performed	BASH Commands
Load .Fluidity's Command Line Interface (CLI)	<code>source Fluidity.sh</code>

Actions Performed	.Fluidity Commands
Install the .Fluidity Server to device	<code>installFluidity</code>
Add the .Fluidity clients	
Add Client	<code>addFluidityClient 1 static_server_ip static_client_ip _Fluidity_Client_1 pi</code>
Attach networks to .Fluidity VPN network	
Attach network 1 to Server which is on eth0	<code>setInternalInterface eth0</code>
Attach network 2 to Client which is on eth0	Execute locally on Client 1: <code>setInternalInterface eth0</code>
Add .Fluidity client connections	
Add a connection to Client	<code>addFluidityConnection 1 1</code>
Add a second connection to Client	<code>addFluidityConnection 1 2</code>
Do routing from .FL Server (Network 1)	
to Network 2	<code>addServerRoute ip route add 192.168.56.0/24 via 192.168.54.1</code>
Do routing from .Fluidity Client (Network 2)	
to Network 1	<code>addClientRoute 1 2 ip route add 192.168.55.0/24 via 192.168.54.2</code>

.Fluidity VLAN Implementation Example

Lighting the Spark...

List of VPN Connections					
.Fluidity Client 1 Connection Parameters					
Connection Number	Connection Type	Server Listening Port	Server Tunnel Interface IP	Client Tunnel Interface IP	Tunnel Network Subnet Mask
2	Tunnel	1441	192.168.54.1	192.168.54.2	30
Initiate .Fluidity VPN connection to Client 1		<code>runFluidity -t 1 2 1441 192.168.54.1 192.168.54.2 30</code>			
Stop .Fluidity VPN connection to Client 1		<code>stopFluidity 1 2</code>			

List of Serial Connections					
.Fluidity Client 1 Connection Parameters					
Connection Number	Connection Type	Server Listening Port	Server Serial Interface	Client Serial Interface	Serial Speed
1	Serial	1440	ttyS1	ttyS1	115200
Initiate .Fluidity serial connection to Client 1		<code>runFluidity -s 1 1 1440 ttyUSB0 ttyUSB0 115200</code>			
Stop .Fluidity serial connection to Client 1		<code>stopFluidity 1 1</code>			

.Fluidity PPPoE Implementation Example

Step by step .Fluidity removal...

Actions Performed	.Fluidity Commands
Remove routes from .FL Server (Network 1)	
to Network 2	<code>removeServerRoute ip route add 192.168.56.0/24 via 192.168.54.1</code>
Remove routes from .Fluidity Client 1 (Network 2)	
to Network 1	<code>removeClientRoute 1 2 ip route add 192.168.55.0/24 via 192.168.54.2</code>
Remove .Fluidity client connections	
Remove the #1 Connection from Client 1	<code>removeFluidityConnection 1 1</code>
Remove the #2 Connection from Client 1	<code>removeFluidityConnection 1 2</code>
Remove the .Fluidity clients	
Remove Client 1	<code>removeFluidityClient 1</code>
Detach the networks from the .Fluidity VPN network	
Detach network 1 from Server which is on eth0	<code>removeInternalInterface eth0</code>
Detach network 2 from Client which is on eth0	<code>Execute locally on Client 1: removeInternalInterface eth0</code>
Remove the .Fluidity server	
ecryptFs unmount the .Fluidity Server folder	<code>BASH: sudo umount Fluidity_Server</code>
Remove .Fluidity main Server folder	<code>BASH: rm -r Fluidity_Server</code>

.Fluidity PPPoE Example – Visual Guided Tour

Step-by-step installation & configuration

```
pi@fluidity-server: ~
File Edit Tabs Help
pi@fluidity-server:~ $ source Fluidity.sh
pi@fluidity-server:~ $
```

1. source Fluidity.sh
Import .Fluidity's command set

```
pi@fluidity-server: ~
File Edit Tabs Help
pi@fluidity-server:~ $ installFluidity
```

2. installFluidity
Begin the installation process

```
pi@fluidity-server: ~
File Edit Tabs Help
--- 8.8.8.8 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 5ms
rtt min/avg/max/mdev = 41.779/42.291/42.773/0.406 ms
Get:1 http://raspbrian.raspberrypi.org/raspbian buster InRelease [15.0 kB]
Hit:2 http://archive.raspberrypi.org/debian buster InRelease
Get:3 http://raspbrian.raspberrypi.org/raspbian buster/main armhf Packages [13.0 MB]
Fetched 13.0 MB in 17s (762 kB/s)
Reading package lists...
Reading package lists...
Building dependency tree...
Reading state information...
Calculating upgrade...
The following package was automatically installed and is no longer required:
  libhavege1
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.

Fluidity requires a high quality entropy source
which utility you prefer to choose?
1. for Haveged
2. for rng-tools
2
```

2.1 installFluidity
Install rng-tools (choice 2)

```
pi@fluidity-server: ~
File Edit Tabs Help
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.

Fluidity requires a high quality entropy source
which utility you prefer to choose?
1. for Haveged
2. for rng-tools
2
Installing rng-tools
Reading package lists...
Building dependency tree...
Reading state information...
rng-tools is already the newest version (2-unofficial-mt.14-1).
The following package was automatically installed and is no longer required:
  libhavege1
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Entropy is within limits (more than 1000).
If you are on a newly configured Fluidity server, you can now
proceed adding a new client by using addFluidityClient.
net.ipv4.ip_forward = 1

Please choose your Fluidity master password:
.0000
```

2.2 installFluidity
Give the master password (i.e. 0000)

```
ppp0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST> mtu 1492
inet 2.85.51.124 netmask 255.255.255.255 destination 80.106.125.100
ppp txqueuelen 3 (Point-to-Point Protocol)
RX packets 750566 bytes 316087878 (301.4 MiB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 577131 bytes 204438554 (194.9 MiB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

3. ifconfig
For the ppp0 interface, observe the server's IP address

```
pi@fluidity-server: ~/Fluidity_Server
File Edit Tabs Help
pi@fluidity-server:~/Fluidity_Server $ addFluidityClient 1 2.85.51.124 2.85.61.132 _Fluid
ity_cli3nt pi
```

4. addFluidityClient 1
Add .Fluidity client 1 by using local and remote ppp0 IP address

Next Slide

.Fluidity PPPoE Example – Visual Guided Tour

Step-by-step configuration

Previous Slide

```
pi@fluidity-server: ~/Fluidity_Server
File Edit Tabs Help
Enter passphrase for /home/pi/.ssh/client.1:
Identity added: /home/pi/.ssh/client.1 (SSH remote connection to Fluidity client 1)
spawn ssh pi@2.85.61.132
The authenticity of host '2.85.61.132 (2.85.61.132)' can't be established.
ECDSA key fingerprint is SHA256:zofITX/VfAbgNQIw9q9ZwdW7bmu0WjZ20-qWwbjMa8.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '2.85.61.132' (ECDSA) to the list of known hosts.
pi@2.85.61.132's password: /usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed:
"/home/pi/.ssh/client.1.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any
that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it
is to install the new keys

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'pi@2.85.61.132'"
and check to make sure that only the key(s) you wanted were added.

Fluidity requires a high quality entropy source
Which utility do you prefer to choose?
1. for HwEggs
2. for rng-tools
3
```

4.1 addFluidityClient 1

Client Side: Install rng-tools (choice 2)

```
pi@fluidity-server: ~/Fluidity_Server
File Edit Tabs Help
Default routed policy changed to 'allow'
(be sure to update your rules accordingly)
Skipping adding existing rule
Skipping adding existing rule (v6)
Rule added
Status: active

To Action From
----
22/tcp ALLOW Anywhere
5900 ALLOW Anywhere
61551/tcp ALLOW 2.85.61.124 # HFBCvIa7h 2.85.61.132
22/tcp (v6) ALLOW Anywhere (v6)
5900 (v6) ALLOW Anywhere (v6)

Rule deleted
Rule deleted (v6)
cat: /home/pi/.fluidity/installation_outcome.txt: No such file or directory
Seek And Encrypt token filename is: 4D7P6bZ2VD
Seal 1 password is: qg804TpvRbBenG3KapsNA==
Seal 2 password is: HF706FSTkqTnZ2fs4M3bkW==
FLdaemon_SeekAndEncrypt.sh 100% 4572 73.5KB/s 00:00
FLdaemon_SeekAndEncrypt.service 100% 198 9.8KB/s 00:00
pi@fluidity-server:~/Fluidity_Server $
```

4.2 addFluidityClient 1

Finalizing client 1 installation

```
pi@fluidity-server:~/Fluidity_Server $ addFluidityConnection 1 1
PING 2.85.61.132 (2.85.61.132) 56(64) bytes of data:
64 bytes from 2.85.61.132: icmp_seq=1 ttl=63 time=16.4 ms
64 bytes from 2.85.61.132: icmp_seq=2 ttl=63 time=15.3 ms
64 bytes from 2.85.61.132: icmp_seq=3 ttl=63 time=15.3 ms

--- 2.85.61.132 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 5ms
rtt min/avg/max/mdev = 15.252/15.632/16.367/0.519 ms
Connectivity test to host 2.85.61.132 succeeded. Proceeding with client configuration.
2048 SHA256:MF4LQudntr41z9Jumz3qfCjKvHucVy13TsFA61KAoPE SSH remote connection to Fluidity
client 1 (RSA)
Fluidity system file integrity test passed
Entropy test passed. Proceeding with installSSLCertificates.
cqll1AwSQuImAVz
spawn sudo mount -t ecryptfs -o key=passphrase:passphrase_passwd=cqll1AwSQuImAVz,ecryptf
s_cipher=aes,ecryptfs_key_bytes=32,ecryptfs_enable_filename=y,ecryptfs_passthrough=n,ecry
ptfs_enable_filename_crypto=y ./Fluidity_Client/connection.1.1 ./Fluidity_Client/connecti
on.1.1
Filename Encryption Key (FNEK) Signature [ab845847d0aa9bfe]:
Attempting to mount with the following options:
ecryptfs_unlink_sigs
ecryptfs_fnek_sig=ab845847d0aa9bfe
ecryptfs_key_bytes=32
```

5. addFluidityConnection 1 1

Add connection 1 to client 1

```
pi@fluidity-server: ~/Fluidity_Server
File Edit Tabs Help
Would you like to proceed with the mount (yes/no)? : yes
Would you like to append sig [ab845847d0aa9bfe] to
[/root/.ecryptfs/sig-cache.txt]
In order to avoid this warning in the future (yes/no)? : yes
Successfully appended new sig to user sig cache file
Mounted ecryptfs
VZUW9v56AEdS6A2A
fkG95a/FzgKtZNTF
6P0gVg96TWnARkg
.....++++
spawn openssl req -new -key servercon.1.1.key -x509 -days 3653 -subj /C=GR/ST=Serres/L=Se
rrres/O=OTE Group/OU=Infrastructure Team/CN=2.85.61.124 -out servercon.1.1.crt
Enter pass phrase for servercon.1.1.key:
.....++++
spawn openssl req -new -key clientcon.1.1.key -x509 -days 3653 -subj /C=GR/ST=Serres/L=Se
rrres/O=OTE Group/OU=Infrastructure Team/CN=2.85.61.132 -out clientcon.1.1.crt
Enter pass phrase for clientcon.1.1.key:
clientcon.1.1.crt 100% 1383 45.1KB/s 00:00
clientcon.1.1.pem 100% 3257 66.4KB/s 00:00
servercon.1.1.crt 100% 1383 45.6KB/s 00:00
pi@fluidity-server:~/Fluidity_Server $
```

5.1 addFluidityConnection 1 1

Finalizing connection 1 1 installation

```
pi@fluidity-server:~/Fluidity_Server $ addFluidityConnection 1 2
PING 2.85.61.132 (2.85.61.132) 56(64) bytes of data:
64 bytes from 2.85.61.132: icmp_seq=1 ttl=63 time=15.4 ms
64 bytes from 2.85.61.132: icmp_seq=2 ttl=63 time=16.1 ms
64 bytes from 2.85.61.132: icmp_seq=3 ttl=63 time=14.7 ms

--- 2.85.61.132 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 5ms
rtt min/avg/max/mdev = 14.665/15.407/16.130/0.615 ms
Connectivity test to host 2.85.61.132 succeeded. Proceeding with client configuration.
2048 SHA256:MF4LQudntr41z9Jumz3qfCjKvHucVy13TsFA61KAoPE SSH remote connection to Fluidity
client 1 (RSA)
Fluidity system file integrity test passed
Entropy test passed. Proceeding with installSSLCertificates.
UkXQGD0qyQXeuZFn
spawn sudo mount -t ecryptfs -o key=passphrase:passphrase_passwd=UkXQGD0qyQXeuZFn,ecryptf
s_cipher=aes,ecryptfs_key_bytes=32,ecryptfs_enable_filename=y,ecryptfs_passthrough=n,ecry
ptfs_enable_filename_crypto=y ./Fluidity_Client/connection.1.2 ./Fluidity_Client/connecti
on.1.2
Filename Encryption Key (FNEK) Signature [a274f1a2068cd092]:
Attempting to mount with the following options:
ecryptfs_unlink_sigs
ecryptfs_fnek_sig=a274f1a2068cd092
```

6. addFluidityConnection 1 2

Add connection 2 to client 1

```
pi@fluidity-server:~/Fluidity_Server
File Edit Tabs Help
Would you like to append sig [a274f1a2068cd092] to
[/root/.ecryptfs/sig-cache.txt]
In order to avoid this warning in the future (yes/no)? : yes
Successfully appended new sig to user sig cache file
Mounted ecryptfs
XT/eFmJ2vmsk+7n
6P0gVg96TWnARkg
.....++++
spawn openssl req -new -key servercon.1.2.key -x509 -days 3653 -subj /C=GR/ST=Serres/L=Se
rrres/O=OTE Group/OU=Infrastructure Team/CN=2.85.61.132 -out servercon.1.2.crt
Enter pass phrase for servercon.1.2.key:
.....++++
spawn openssl req -new -key clientcon.1.2.key -x509 -days 3653 -subj /C=GR/ST=Serres/L=Se
rrres/O=OTE Group/OU=Infrastructure Team/CN=2.85.61.132 -out clientcon.1.2.crt
Enter pass phrase for clientcon.1.2.key:
clientcon.1.2.crt 100% 1383 45.8KB/s 00:00
clientcon.1.2.pem 100% 3257 67.0KB/s 00:00
servercon.1.2.crt 100% 1383 44.5KB/s 00:00
pi@fluidity-server:~/Fluidity_Server $
```

6.1 addFluidityConnection 1 2

Finalizing connection 1 2 installation

Next Slide

.Fluidity PPPoE Example - Visual Guided Tour

Step-by-step configuration

Previous Slide

```
pi@fluidity-server: ~/Fluidity_Server
File Edit Tabs Help
pi@fluidity-server:~/Fluidity_Server $ addServerRoute ip route add 192.168.56.0/24 via 192.168.54.1
Creating serverRoutes.sh
pi@fluidity-server:~/Fluidity_Server $
```

7. addServerRoute

Instruct the server's OS to reach network 192.168.56.0/24 via the IP address 192.168.54.1 (this will be the IP address of tun0 interface when runFluidity -t is executed).

```
pi@fluidity-server: ~/Fluidity_Server
File Edit Tabs Help
pi@fluidity-server:~/Fluidity_Server $ addClientRoute 1 2 ip route add 192.168.55.0/24 via 192.168.54.2
Creating clientRoutes.1.2.sh
pi@fluidity-server:~/Fluidity_Server $
```

8. addClientRoute

Instruct client 1 connection 2 to reach network 192.168.55.0/24 via the IP address 192.168.54.2 (this will be the IP address of the tun0 interface when runFluidity -t is executed).

```
pi@fluidity-server: ~
File Edit Tabs Help
pi@fluidity-server:~ $ setInternalInterface eth0
eth0
Rule added
Rule added (v6)
Rule added
Rule added (v6)
```

9. setInternalInterface eth0

Server Side: Designate interface eth0 as an internal interface, so that traffic can go through it freely.

```
pi@fluidity-server: ~
File Edit Tabs Help
pi@fluidity-server:~ $ sudo ufw status
Status: active

To Action From
--
5900/tcp ALLOW Anywhere
Anywhere on eth0 ALLOW Anywhere
Anywhere (v6) on eth0 ALLOW Anywhere (v6)
Anywhere ALLOW OUT Anywhere on eth0
Anywhere (v6) ALLOW OUT Anywhere (v6) on eth0
```

9.1 setInternalInterface eth0

Server Side: Observe the orange frames indicating the change on firewall rules.

```
pi@fluidity-client: ~
File Edit Tabs Help
pi@fluidity-client:~ $ source Fluidity.sh
pi@fluidity-client:~ $ setInternalInterface eth0
eth0
Rule added
Rule added (v6)
Rule added
Rule added (v6)
```

10. setInternalInterface eth0

Client Side: Designate interface eth0 as an internal interface, so that traffic can go through it freely.

```
pi@fluidity-client: ~
File Edit Tabs Help
pi@fluidity-client:~ $ sudo ufw status
Status: active

To Action From
--
61351/tcp ALLOW 2.85.51.124 # HFBcvIa7h 2.
85.61.132
Anywhere on eth0 ALLOW Anywhere
Anywhere (v6) on eth0 ALLOW Anywhere (v6)
Anywhere ALLOW OUT Anywhere on eth0
Anywhere (v6) ALLOW OUT Anywhere (v6) on eth0
```

10.1 setInternalInterface eth0

Client Side: Observe the orange frames indicating the change on firewall rules.

Next Slide

.Fluidity PPPoE Example – Visual Guided Tour

.Fluidity execution

Previous Slide

```
pi@fluidity-server: ~/Fluidity_Server
File Edit Tabs Help
pi@fluidity-server:~/Fluidity_Server $ runFluidity -s 1 1 1440 ttyUSB0 ttyUSB0 9600
```

11. runFluidity -s

Execute .Fluidity and link serial devices through an ethernet tunnel. This is .Fluidity for client 1 and connection 1, through port 1440, for local serial device ttyUSB0 and client serial device ttyUSB0 for 9600 symbols/sec.

```
pi@fluidity-server: ~/Fluidity_Server
File Edit Tabs Help
Inside the Loop and proceeding with runSOCATclient.
Ping delay is: 2
servercon.1.1.pem MD5 is: 7d2ce944fcb67dc6fc39b2f627e1b50a
servercon.1.1.crt MD5 is: 7d2ce944fcb67dc6fc39b2f627e1b50a
clientcon.1.1.pem MD5 is: 12c148317464e561520e98801496d40e
clientcon.1.1.crt MD5 is: 12c148317464e561520e98801496d40e
doAClientServerMD5EquivalencyCheck PASSED
servercon.1.1.pem SHA256 is: d518482441b40ab20f548330383ee424
servercon.1.1.crt SHA256 is: d518482441b40ab20f548330383ee424
clientcon.1.1.pem SHA256 is: e72117e9004d96e016b4e354f1c5b41a
clientcon.1.1.crt SHA256 is: e72117e9004d96e016b4e354f1c5b41a
doAClientServerSHA256EquivalencyCheck PASSED
spawn socat openssl:2.85.51.124:1441,verify=1,cert=clientcon.1.1.pem,cafile=servercon.1.1.crt, /dev/ttyUSB0,b9600,echo=0,raw
Enter PEM pass phrase:tcp 0 0 2.85.51.124:1440 2.85.61.132:40082
ESTABLISHED 30274/socat
1440
1440 (v6) ALLOW IN Anywhere
ALLOW IN Anywhere (v6)

[3] Done reportWhenLinkIsEstablished $1 $3
[4]- Done openTheTunnelInterfaces $1 $2 $3 $4 $9
[5]+ Done reportWhenFirewallRulesAreAdded $1 $3
pi@fluidity-server:~/Fluidity_Server $
```

11.1 runFluidity -s

.Fluidity reports that both MD5 and SHA256 SSL certificate hashes match, that for port 1440 a TCP link is established and that a firewall rule was added allowing traffic through port 1440.

```
pi@fluidity-server: ~/Fluidity_Server
File Edit Tabs Help
pi@fluidity-server:~/Fluidity_Server $ runFluidity -t 1 2 1441 192.168.54.1 192.168.54.2 30
```

12. runFluidity -t

Execute .Fluidity and establish a tunnel link. The tunnel will consist of IP addresses 192.168.54.1 and 192.168.54.2 on server and client, respectively, to securely connect the networks attached on each .Fluidity device.

```
pi@fluidity-server: ~/Fluidity_Server
File Edit Tabs Help
Inside the Loop and proceeding with runSOCATclient.
Ping delay is: 2
servercon.1.2.pem MD5 is: 7fdad7c8be8f41959a6a5c459e125818
servercon.1.2.crt MD5 is: 7fdad7c8be8f41959a6a5c459e125818
clientcon.1.2.pem MD5 is: c2951b66375076f7b2bf8c2c9062fe50
clientcon.1.2.crt MD5 is: c2951b66375076f7b2bf8c2c9062fe50
doAClientServerMD5EquivalencyCheck PASSED
servercon.1.2.pem SHA256 is: dcf7c506184bab414eb647032605c5bd
servercon.1.2.crt SHA256 is: dcf7c506184bab414eb647032605c5bd
clientcon.1.2.pem SHA256 is: 9be2b8a3344ac12b6e5b2393352c1fb8
clientcon.1.2.crt SHA256 is: 9be2b8a3344ac12b6e5b2393352c1fb8
doAClientServerSHA256EquivalencyCheck PASSED
spawn sudo socat openssl:2.85.51.124:1441,verify=1,cert=clientcon.1.2.pem,cafile=servercon.1.2.crt, TUN:192.168.54.2/30,up
Enter PEM pass phrase:ready to execute
tcp 0 0 2.85.51.124:1441 2.85.61.132:60738 ESTABLISHED -
1441
1441 (v6) ALLOW IN Anywhere
ALLOW IN Anywhere (v6)

[5] Done reportWhenLinkIsEstablished $1 $3
[6]- Done openTheTunnelInterfaces $1 $2 $3 $4 $9
[7]+ Done reportWhenFirewallRulesAreAdded $1 $3
pi@fluidity-server:~/Fluidity_Server $
```

12.1 runFluidity -t

.Fluidity reports that both MD5 and SHA256 SSL certificate hashes match, that for port 1441 a TCP link is established and that a firewall rule was added allowing traffic through port 1441.

Next Slide

.Fluidity PPPoE Example – Visual Guided Tour

A closer look to the SOCAT generated tunnel, active link firewall statuses and .Fluidity's reporting commands.

Previous Slide

```
tun0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST> mtu 1500
    inet 192.168.54.1 netmask 255.255.255.252 destination 192.168.54.1
    inet6 fe80::d597:f3bc:55b2:1065 prefixlen 64 scopeid 0x20<link>
    unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00 txqueuelen 500
```

13.1 SERVER SIDE: ifconfig

Once the link is up, tun0 interface is created. Observe tun0 IP address and network mask for the demonstrated ifconfig output.

```
tun0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST> mtu 1500
    inet 192.168.54.2 netmask 255.255.255.252 destination 192.168.54.2
    inet6 fe80::85d4:a2f:6b24:70fe prefixlen 64 scopeid 0x20<link>
    unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00 txqueuelen 500
```

13.2 CLIENT SIDE: ifconfig

Respectively, observe the tun0 IP address and network mask for the demonstrated ifconfig output on client's side

```
pi@fluidity-server: ~
File Edit Tabs Help

pi@fluidity-server:~ $ sudo ufw status
Status: active

To Action From
--
Anywhere on eth0 ALLOW Anywhere
1440 ALLOW Anywhere
1441 ALLOW Anywhere
Anywhere on tun0 ALLOW Anywhere

Anywhere (v6) on eth0 ALLOW Anywhere (v6)
1440 (v6) ALLOW Anywhere (v6)
1441 (v6) ALLOW Anywhere (v6)
Anywhere (v6) on tun0 ALLOW Anywhere (v6)

Anywhere ALLOW OUT Anywhere on eth0
Anywhere ALLOW OUT Anywhere on tun0
Anywhere (v6) ALLOW OUT Anywhere (v6) on eth0
Anywhere (v6) ALLOW OUT Anywhere (v6) on tun0
```

14. SERVER SIDE: ufw status

Observe the rules created allowing traffic through ports 1440 and 1441 for the active .Fluidity connections [1,1] and [1,2].

```
pi@fluidity-server: ~
File Edit Tabs Help

pi@fluidity-server:~ $ showLinkStatus 1 1
Fluidity Connection ID: 1.1
Fluidity Flavour: Serial Link
Serial Link Speed: 9600
Server Listening Port: 1440
Server Serial Interface: ttyUSB0
Server IP Address: 2.85.51.124
Client Serial Interface: ttyUSB0
Client IP Address: 2.85.61.132
Client Username: pi
Netstat Reports: ESTABLISHED
Fluidity Connection Status: ACTIVE
```

15.1 showLinkStatus 1 1
showLinkStatus enables us to see the current .Fluidity link status.

```
pi@fluidity-server: ~
File Edit Tabs Help

pi@fluidity-server:~ $ showLinkStatus 1 2
Fluidity Connection ID: 1.2
Fluidity Flavour: Tunnel Link
Server Listening Port: 1441
Server IP Address: 2.85.51.124
Server Tunnel IP Address: 192.168.54.1
Client IP Address: 2.85.61.132
Client Tunnel IP Address: 192.168.54.2
Network Subnet Mask: 30
Client Username: pi
Netstat Reports: ESTABLISHED
Fluidity Connection Status: ACTIVE
```

15.2 showLinkStatus 1 2
showLinkStatus enables us to see the current .Fluidity link status.

Enjoy your .Fluidity!

