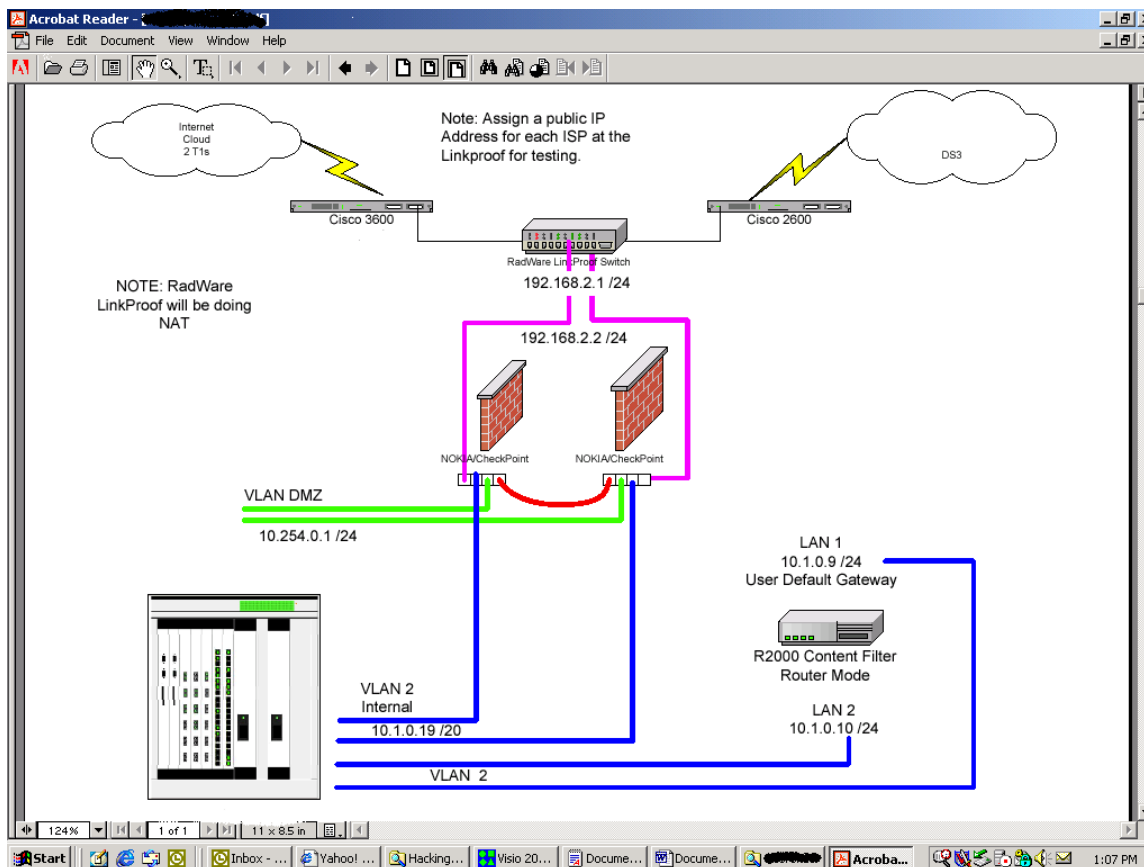


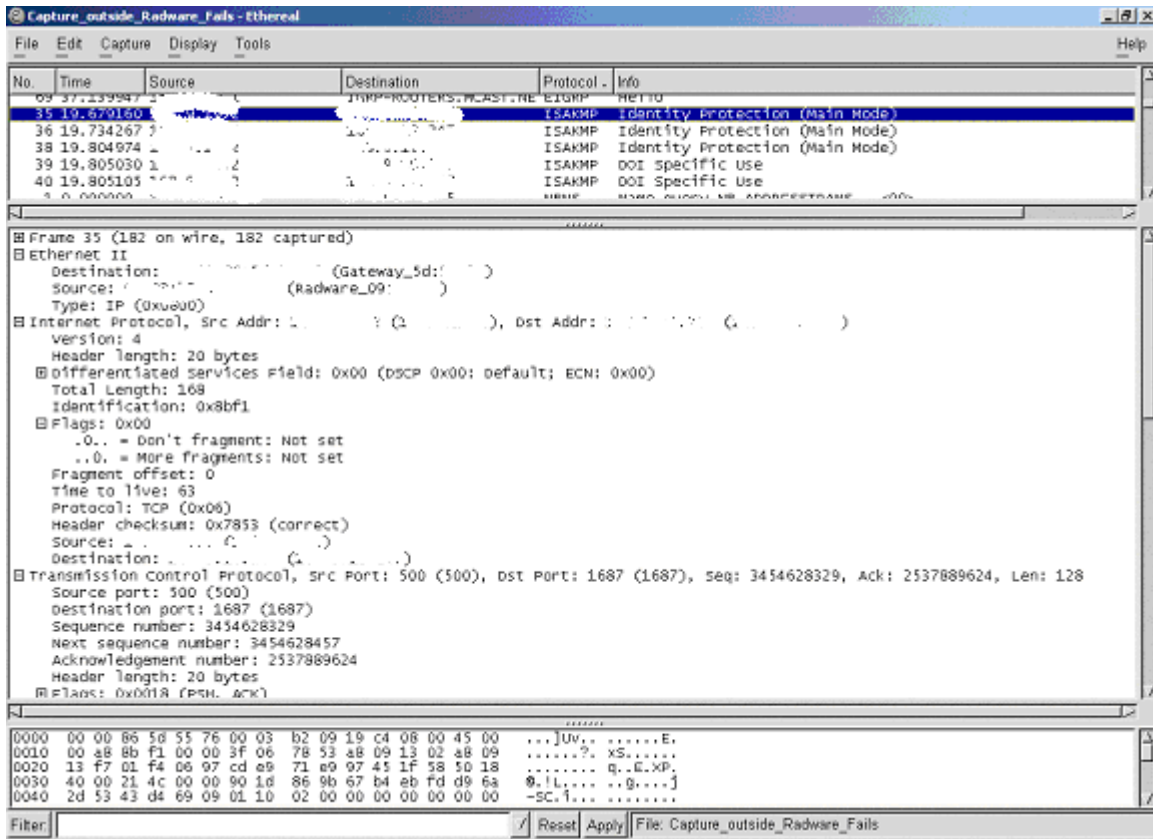
SecureClient to a Firewall with NATed IPs

Question – Will SecureClient work with a Firewall that has a private external IP address, where the Firewall’s private external address is being NATed to a public IP by another network device?

The following diagram presents a configuration where the firewalls are running in High Availability and have private external IP addresses. The addresses were being NATed to public IPs by a Radware Linkproof external to the firewall pair.

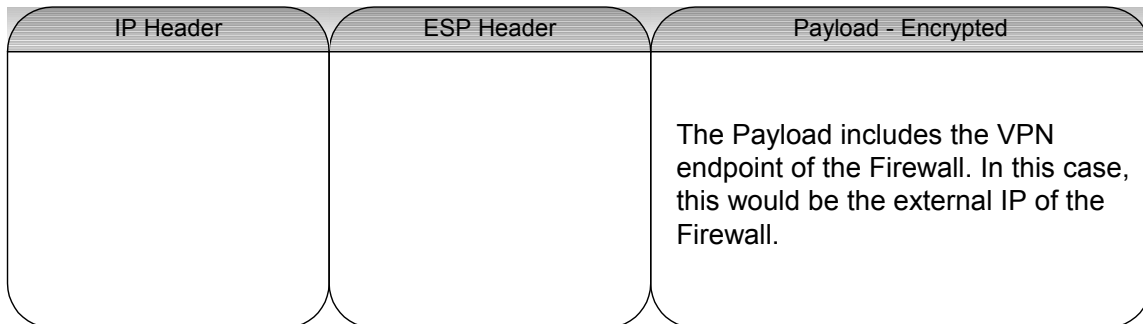


When testing SecureClient, we pointed the client to the public IP being advertised by the Radware. The problem is that the IKE handshake would not complete. It appears that the client is unable to determine the VPN endpoint. The initial assumption was that the Radware was not able to correctly NAT the encrypted packet. But further investigation concluded this was not so. See the Ethereal capture below.



Reviewing the UserC.C file indicated that he was receiving the topology from the firewall, but the firewall was sending it's private IP, because that's the only IP he knows about. The IKE handshake was unable to complete because the client knows of a different endpoint 192.168.2.2 instead of <Public IP>. The Payload contains the VPN endpoint sent from the Firewall to the client. In this case, it's the wrong IP.

In addition, we need to support ESP only, no AH. AH was been widely known to not work through NATed devices. No problem, we discovered that CheckPoint's implementation of IKE/ESP doesn't include AH. Authentication and Integrity are provided within CheckPoint's ESP implementation without the use of AH.



:sds_servers ()

Also, these must be configured in the client:

- Force UDP encapsulation
- Support IKE over TCP

But three problems still exist:

1. How do we get the firewall to send the SecureClient it's public IP, when the public IP is not part of the firewall's configuration?
2. How do we prevent the client from downloading and new topology, thus overwriting the good UserC.C file?
3. How do we give the user the correct UserC.C file, short of having them manually copy a file onto their machine?

We were unable to get the firewall to send it's public IP. CheckPoint provided an unsupported solution that apparently works fine for a single firewall configuration, but not an H/A configuration such as ours:

CheckPoint's Recommendation:

There is a work around for this, although it is not officially supported. A example description is below, assume the external firewall is the Radware device and static NAT will be required....

SR----Internet ----- (1.1.1.1) External_FW (10.1.11) -----(10.1.1.2) Internal_FW (11.1.1.1)----

The goal is to have SecuRemote encrypt directly to the Internal Fw. The issue is that SecuRemote tries to perform a key exchange with the 10.1.1.2 address. There is a Static NAT rule for 10.1.1.2 to translate it to 1.1.1.2 on the External_FW. To address this situation you need to assign a bogus address to the firewall for 1.1.1.2. Meaning, after you get the interfaces, you would need to manually add the 1.1.1.2 interface and make that the IP of your gateway object. This is so the gateway gives the client proper information about it's NATed address and not it's real address. In addition to this you should force UDP encapsulation on the client to help with phase I.

It sounds easy enough – Simply change the IP on the General Tab to the public IP and create a bogus interface in the Interfaces tab with the public IP as well. Unfortunately, the Cluster Firewall Object only allows you to change the IP on the General tab. We completely broke the VPN when we performed this. Next we edited the individual firewalls and added the bogus interfaces, but that prevented us from installing a policy.

So, we were unable to resolve Question #1 - **How do we get the firewall to send the SecureClient it's public IP, when the public IP is not part of the firewall's configuration?**

Even so, we needed to provide the customer with a workable solution to his dilemma.

So we proceeded to Question #2 - **How do we prevent the client from downloading and new topology, thus overwriting the good UserC.C file?**

We simply disabled the Automatic Download feature in the Global Properties and set the update feature to 5000000 hours. Not foolproof, but should prevent it for the next year or so. ;-)

Onto Question#3 - **How do we give the user the correct UserC.C file, short of having them manually copy a file onto their machine?**

This is where the SecureClient packaging tool comes in handy. We simply downloaded the latest version of SecureClient and unzipped it. Next we manually copied the "good" UserC.C file and overwrote the one in the install directory. Then we kicked off the SecureClient packaging tool and created our install executable which has the "good" UserC.C file bundled with it. Running the install gave the client the correct UserC.C file and a VPN that works!

Many thanks to the following people for their relentless dedication and time in assisting with resolving this:

Gary Scott - VeriSign
Scott Rachford - VeriSign
Tim Hildebrand – CheckPoint
Lisa Phifer - CoreCom