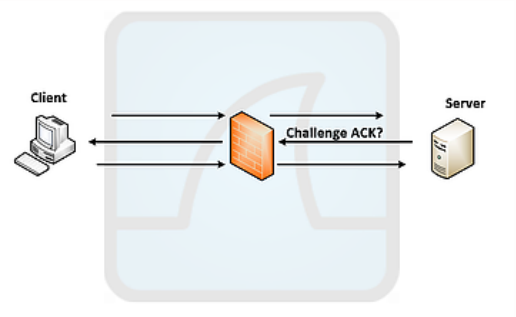
Brandon Hitzel · Nov 26 · 10 min read

Troubleshooting with Wireshark: The Case of the TCP Challenge ACK

Updated: Nov 28

Troubleshooting an issue where the server replies with an ACK only instead of SYN/ACK. Also some simple Wireshark tips.



I think there is one sentence you never want to hear as a network engineer or firewall/security administrator and that is "The website works everywhere except on our network, it must be a firewall problem!". Well in some cases it might be and in other cases it's the other network's problem.

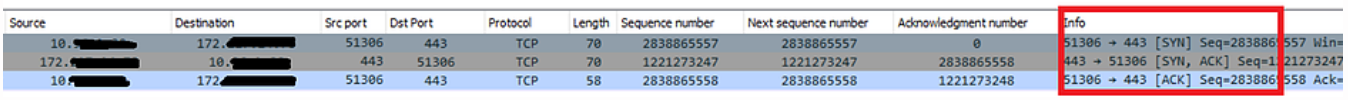
Recently I was confronted with this issue for one of my customers stating this exact problem. Certain websites would load via other connections off network, but on their network sometimes the web pages wouldn't load, it was fairly intermittent but eventually started lasting for days it seemed. Below I will outline what I found and how it was resolved and discuss a few things related to it including some wireshark tips.

Background

When troubleshooting via a firewall one of the best things you can do is setup packet captures. Sometimes you will have to even setup multiple captures: one at the client side, one at the firewall, then one on the server side or outside of the firewall. This is so you can see if packets are being dropped or changed along the path for example. However in my case it became pretty clear on why certain web pages were not loading within the network once I performed a packet capture on the firewall.

First, during normal TCP connection conditions a 3-way handshake is established. The client will send a TCP packet with the SYN (Synchronization) flag set, secondly the receiving server will send its own SYN with the ACK (Acknowledgement) flag also set. This is so it can acknowledge the previous SYN from the client. Finally during the 3rd step the client will respond with an ACK to the SYN the server sent. After that the two computers can exchange whatever data they intended. Also during the life of the TCP connection there are

exchange whatever data they intended. Also during the life of the TCP connection there are numbers called 'sequence numbers' which are used to track the exchange of information, this fact will be important later.



Source	Destination	Src port	Dst Port	Protocol	Length	Sequence number	Next sequence number	Acknowledgment number	Info
10.[redacted]	172.[redacted]	51306	443	TCP	70	2838865557	2838865557	0	51306 → 443 [SYN] Seq=283886[redacted]557 Win=[redacted]
172.[redacted]	10.[redacted]	443	51306	TCP	70	1221273247	1221273247	2838865558	443 → 51306 [SYN, ACK] Seq=1[redacted]21273247 [redacted]
10.[redacted]	172.[redacted]	51306	443	TCP	58	2838865558	2838865558	1221273248	51306 → 443 [ACK] Seq=283886[redacted]558 Ack=[redacted]

Good TCP 3-Way Handshake

Simple right, you've probably heard it a 100 times. BUT what do you do if you see something happening during the 3-way handshake that is puzzling? In this case the packet captures were showing the server was sending an ACK on part 2 of the 3-way handshake without the SYN flag set which was causing the client to send a TCP RST (reset) and therefore not establishing the connection as it should.

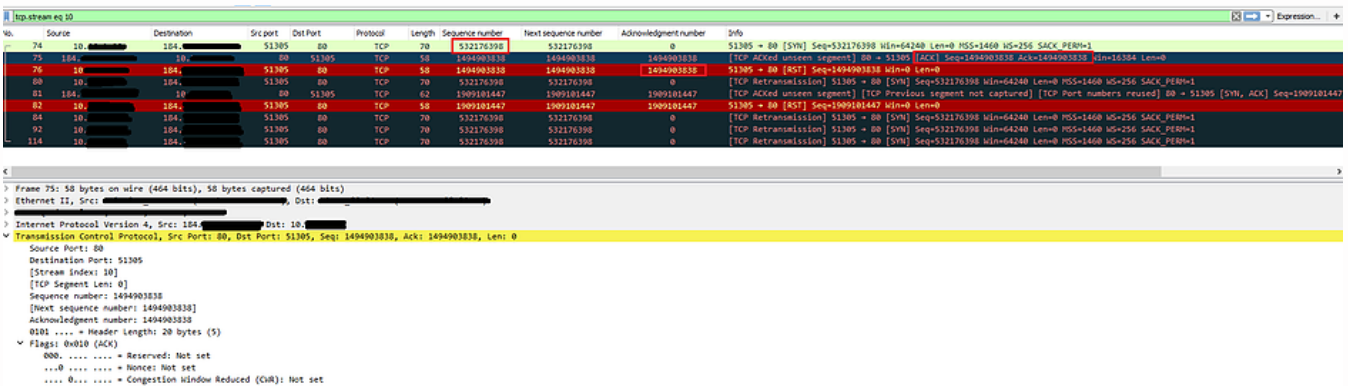
The Case of the Challenge ACK

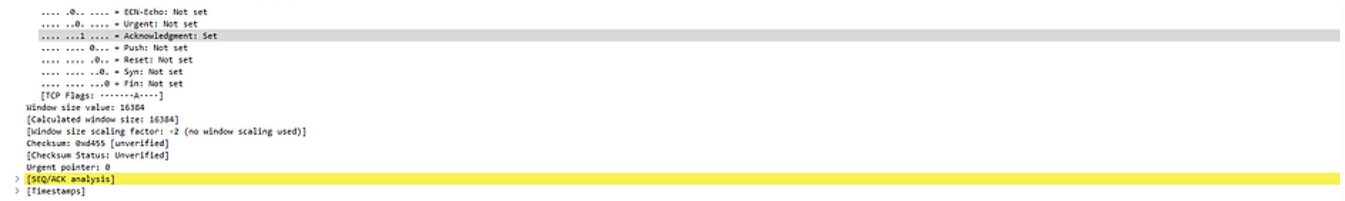
The scope of the problem was certain websites all hosted by the same provider (who will not be named) couldn't load or redirect to a different page. It appeared a lot of different websites all had the same IP A record, i.e. were resolving to the same IP address. I assume this is due to some type of proxy or load- balancer receiving the connections before the web servers. Another thing was some of the websites reported actually needed redirects, so without the established TCP connection the HTTP 301 redirect never came to the client PC.

First thing I checked was DNS internal vs. external - because you know, its *always* DNS. That checked out fine. After trying a few times over the course of 2 days the websites finally stopped being reachable so then I began checking firewalls logs for the IP address of my test machine and then did packet captures. That is when I noticed the 3-way handshake was not completing.

Essentially what I ran into was the server side sending a "Challenge ACK" aka Arbitrary ACK reply aka blind TCP reset attack mitigation. This is outlined in RFC 5961 Sec 3 and 4.

As you can see from the below screenshot of the packet capture, the client was sending the typical SYN, however the reply from the server only had the ACK flag set (second packet is highlighted providing the info in the lower plane), and the acknowledgement number wasn't even matching the original SYN's sequence number. If you look at the first screenshot of the post you can see how the acknowledged sequence number matches with what the client sent.





The first packet listed is the client SYN, you can see the sequence number is 532176398, however in the second packet which is the challenge ACK from the server you can see the acknowledged sequence number is 1494903838 which doesn't appear to match the flow. It should have been 532176399 with the SYN flag also set. The client then sends a reset matching that sequence number.

I later would come to find out that this is correct behavior per the RFC.

> A legitimate peer, after restart, would not have a TCB in the
> synchronized state. Thus, when the ACK arrives, the peer should send
> a RST segment back with the sequence number derived from the ACK
> field that caused the RST.

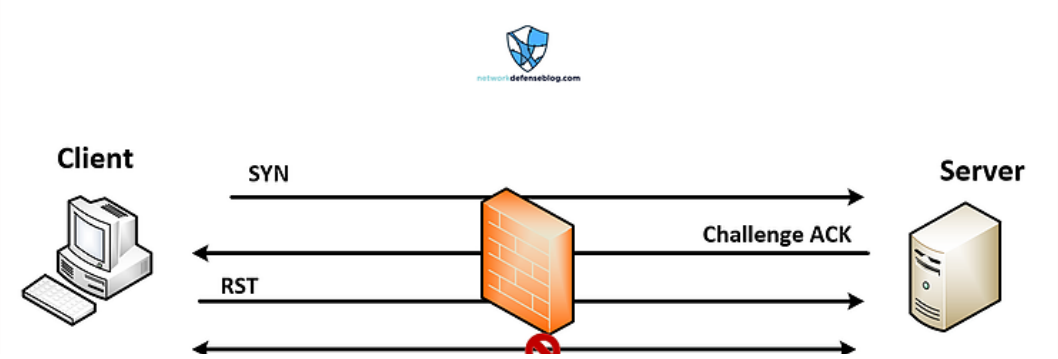
When the client sends the reset it basically closes that connection on the server which could still be open from a previous flow with the same parameters. So when the client retransmits and tries to open the connection again the server replies with the expected SYN/ACK.

Also notice that wireshark is warning of [TCP ACKed unseen segment]. The reason it is showing this message is because when the challenge ACK came in the acknowledgment number was for data that was not present in the capture. Sometimes you will see this if there is packet loss or if the capture lost some packets and did not capture them.

```
51305 → 80 [SYN] Seq=532176398 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
[TCP ACKed unseen segment] 80 → 51305 [ACK] Seq=1494903838 Ack=1494903838 Win=16384 Len=0
51305 → 80 [RST] Seq=1494903838 Win=0 Len=0
```

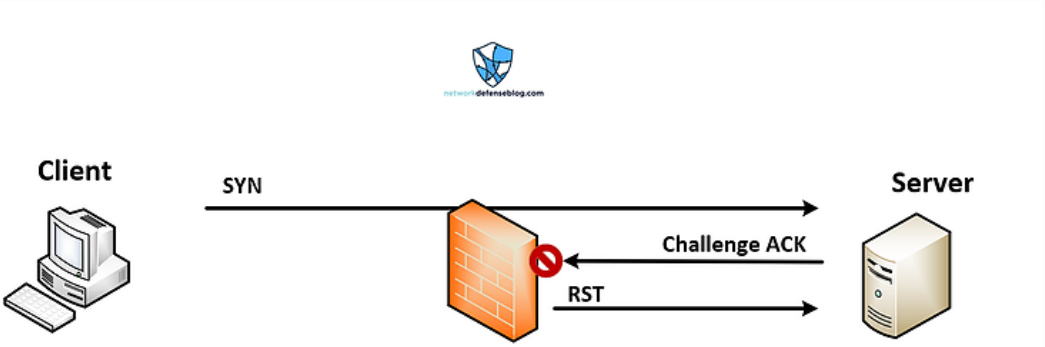
Additionally, wireshark likes to color certain packets. Generally issues like ACKed unseen segment, retransmissions, out-of-order packets and other 'bad TCP' messages are highlighted with red text and black lines. Just something to look out for when scanning a capture for the first time.

Back to the firewall. Once the client reset was seen by the firewall it was marking the connection as completed and dropping subsequent attempts from the client to establish the connection. This next diagram displays this simply.



Firewall Blocks Subsequent attempts on
same IP/Port etc. combination

From my research it seems a lot of session tracking mechanisms would block this. For example the Cisco ASA likely would drop it due to "TCP Reset-I" or reset seen from internal host, and in my case the Palo Alto firewall was dropping it due to "out-of-window-packet-drop". Furthermore, it appears some firewalls performing TCP intercept could potentially drop the challenge ACK before forwarding it to the client. The example below shows the possibility where a reset was sent directly due to dropping the connection before its even established between client and server. This would be dependent on vendor platform and configuration etc.



Firewall Blocks Challenge ACK and sends RST due
to ACK having out of window sequence number

I speculate this challenge ACK was being done by the hosting company because of a DDoS mitigation type mechanism to prevent an excessive amount of SYNs from the same IP or to verify clients are legitimate, because in the case of an attack a lot of times the source addresses would be spoofed. I talk about that a bit in a previous post about "Deception Operations".

Alternatively, perhaps the server is running out of available TCP sockets so it is using the client RSTs to free up connections on reused ports. This would apply if the server still thought the connection was active, it would reply with an acknowledgment of what sequence number it was expecting next. i.e. in the case of NAT you could potentially have a large number of requests coming from a few IP addresses if a large enterprise is accessing the same website/IP or something similar.

I'm not too familiar with the load balancing or web hosting side, and my research didn't really turn up much as far as configuration guides to set up this mechanism. I did see someone talked about this TCP vulnerability was patched in linux about 8 years ago and I found a Red hat patch around the same time, so I assume some deployments of linux hardware will show this type of behavior.

The Fix

After a few calls by the team attempting to escalate up the support chain and getting nowhere - because the firewall drops were obviously being caused by the server TCP responses - I was able to activate "allow challenge ACK" on the Palo firewall to allow the connection behavior to be understood by the software to allow the communication to

complete. This Palo Alto KB article is what lead me to the resolution. Bless the useful vendor docs! Starting with Pan OS 8.0.7 and above you need to enable this feature, it seems before it was allowed automatically. This type of change is something for you to keep in mind when upgrading to newer versions of software. That is of course if it's actually in the release notes.

My googling skills were definitely put to the test on this one until I found out about the RFC and the words "challenge ACK". I was able to confirm other people have experienced this in the past, although the community replies might have pointed to something else being the root cause. In searching for other fixes as if I had a different firewall vendor I found Checkpoint had a feature called Smart Connection Reuse which looked promising or as a lead to research if you have the problem with a Checkpoint in the mix. Another fix which pertains to ASA was found in this TCP normalization config guide and appeared it could be "allow invalid ACK" but I haven't confirmed it.

The Attack

The caveat with enabling some of these features is it could enable an attacker doing a sequence number guess attack to more easily inject a RST packet to tear down a connection. For quick bursty flows that have a short duration this isn't really an issue, but for long standing connections with well-known ports like BGP it could be a problem. MD5 authentication is always recommended with routing protocols and is the mitigation for that problem in the attack scenario.

Although, the sequence number guessing attack is generally possible regardless of allowing the challenge ACK or not. In another Palo Alto KB article it stated these items were needed in order to perform the attack.

"To do this, the attacker must have or guess several pieces of information:

- 4-tuple (IP address and TCP port for both sides of the connection).
- Sequence number that will be used in the RST.
- Window size that the two endpoints are using. This value does ***not*** have to be the exact window size since a smaller value used in lieu of the correct one will just cause the attacker to generate more segments before succeeding in this mischief. Frequently the attacker, with a fair degree of certainty (knowing the application that is under attack), can come up with a very close approximation as to the actual window size in use on the connection.
- The receive window is the number of bytes a sender can transmit without receiving an acknowledgment. "

Nevertheless, some fixes suggested (along with search results) were to allow TCP non-SYN bypass aka TCP-state bypass, or allow a connection without fully seeing the 3-way handshake to establish. Typically this is recommend for scenarios where asymmetric routing is happening with multiple firewalls and/or paths, but if you don't have this type of problem then I wouldn't recommend enabling that feature unless needed due to the security implications of allowing unauthorized connections. (In Palo this configuration is found in the zone-protection profile).

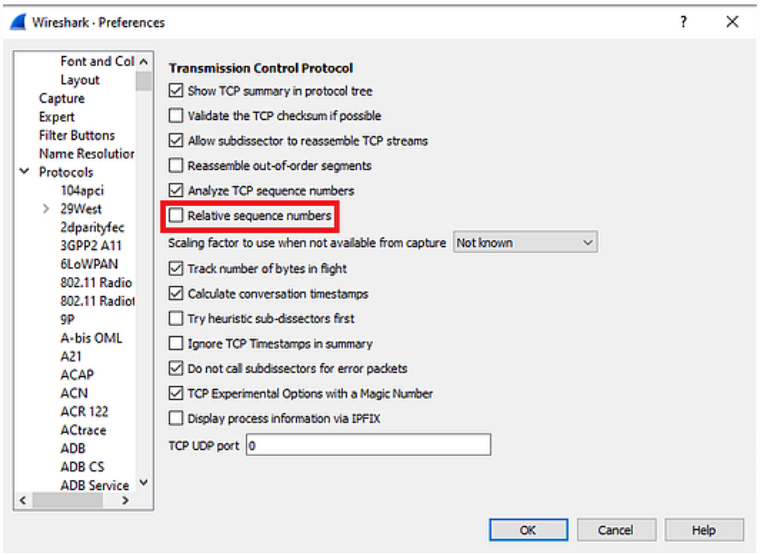
Executing and researching the attack was beyond the scope of this post but in my opinion it seems to have a high amount of effort for low reward. For instance to guess the 4 tuple accurately you would pretty much need to be sniffing traffic or performing a man in the middle to get accurate info which takes some effort as it is, plus you'd have to successfully

predict the acknowledgements. To me it seems a starvation attack or phishing would be easier and more likely to yield results, depending on what the goal is of course.

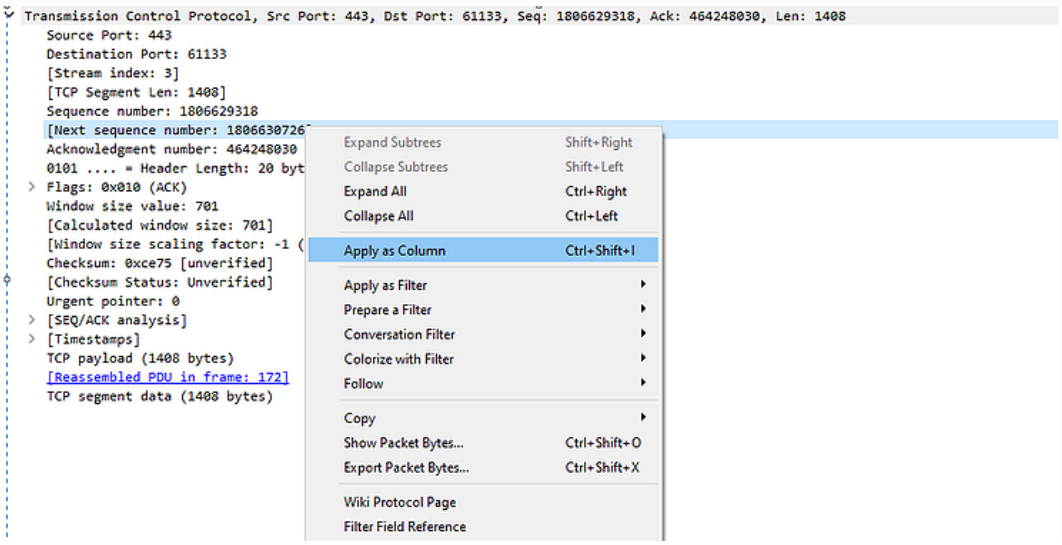
Apparently ISPs and governments have also used similar methods of sending RSTs which is essentially this type of attack. Just something to be aware of when having connection problems in the international landscape.

Wireshark Tips

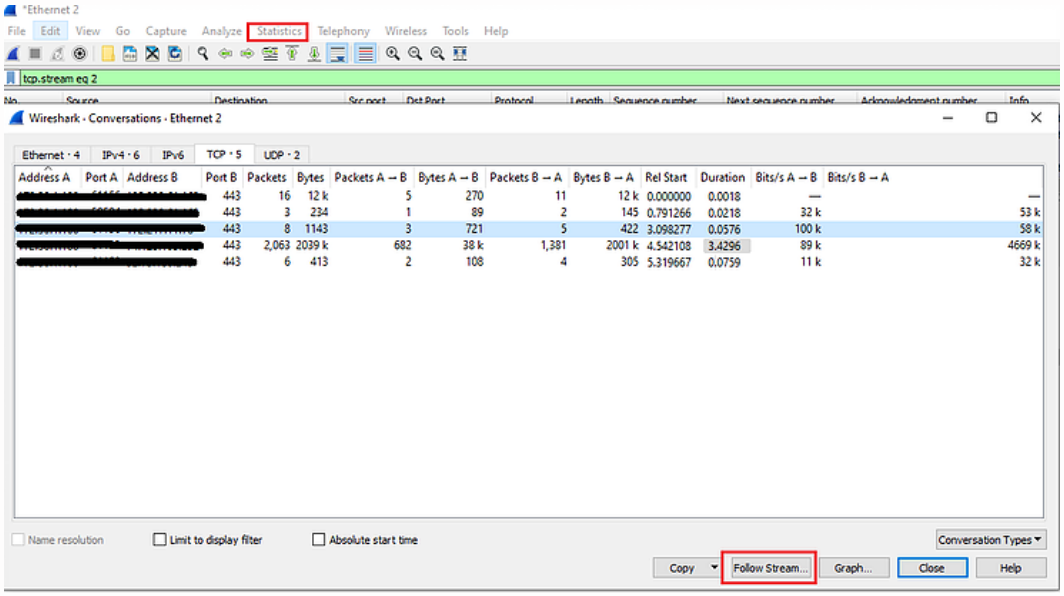
As far as wireshark in order to fully view this issue you need to turn off relative sequence numbers so you can assess the sequence numbers in their true form. In the windows version this can be found in edit > preferences > protocols > TCP. By default this feature will be enabled which causes wireshark to show sequence numbers that are more human friendly; however sometimes things will not match up, therefore its recommended to disable this feature for this scenario.



Another tip is to create columns for each of the sequence numbers. To do this click on any type of TCP packet and open it, then click on the data you want, finally right-click and select "apply as column". So in this case we would want to create a column for the sequence number, acknowledgement number, and next-sequence number. I generally always run these columns in my captures when troubleshooting. Note: you can do this for many different parameters - try it.



When doing a packet capture on a firewall you will most likely setup filters. This is so you can only capture the desired traffic. Although there are ways to do this in wireshark, when capturing on a client machine you might not have this luxury, so a quick way to find the flow you are looking for is by going to the conversations tab and filter there. Statistics > Conversations > click the flow you want after the window pops up > then click follow stream. note: this is also useful to find the largest flow by sorting by # of packets or Bytes etc. if you were looking for a large download or upload.



Closing Notes

With the way the internet landscape is these days it's not surprising that you'd run into something like this. It's understandable that the server side of things would want to protect themselves from different transport control attacks or have a mechanism to free up socket space in the TCP stack when receiving large amounts of requests. However if you are running a solution like this where the challenge ACK is explicitly configured or something - like on a load-balancer, I feel you should have some sort of article or note in your service system in order to allow your help desk to possibly assist in identifying this is the issue. This would be valuable in helping the user/customer.

The fact we couldn't get anywhere with support was somewhat frustrating, so hopefully operators can add additional info for this in the future, even though this vulnerability is over 10 years old. Like I had mentioned try googling for this problem and you will have mixed results.

Although there are some arguments about not enabling some of the features that allow non-standard TCP behavior the likeliness of this type of attack is low compared to some others, and the threat sometimes is there with the feature enabled or not. There are definitely TCP related configurations I would never enable on internet facing interfaces though like full state-bypass. In my case the behavior was already allowed but after a firewall upgrade to a higher version a new feature needed to be enabled and committed to allow it. At first there was some head scratching but as you can see from the packet capture and the references the nature of the flow is to be expected in some cases.

Although an acceptable root cause was determined, there were still some things left unanswered for me in this case; however, without further visibility into the provider network or help from support I likely will not dig any further to analyze this. Plus the customer is happy now that they can access the web page - another ticket closed.

Hopefully this article helps you if you run into this type of issue or something similar. If you have any further information on vendor mechanisms for RFC 5961 please share them here for educational purposes but also on the web to improve the google results.

Thank you

Further reading:
ASA troubleshooting slide deck
Palo Alto troubleshooting via CLI
Palo Alto in-depth flow explanation
CVE info

#firewall #cisco #paloalto #dataconservation #cybersecurity #TCP #netsec

Thank you for reading. Please like, share, and join the mailing list!

Brandon Hitzel is a network engineer hailing out of California. He holds multiple industry certifications including SSCP, CCDP, and CCNP. He has been in the industry for a number of years and enjoys everything networking, defense, and tech related.

Don't forget to vote for network defense blog in the Cisco IT blog awards!

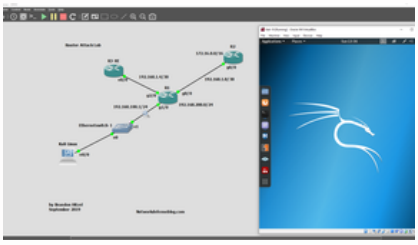
Much appreciated!

Networking • Study/Certications

Recent Posts

See All

Cisco IT Blog Award Finalist in "Best Analysis" Vote Now!



5 Easy Router Protection Techniques - includes Attack an...

CCDP Certification Study Material

Write a comment Write a comment Write a comment 1

Log in to leave a comment.

Contact Me

Professional | Personal | Consulting | Volunteering
Use the below form to drop me a line

Join the mailing list
Never miss an update

Email Address

Click if you're a Human

Subscribe Now

Name *

Email *

Subject *

Message *

Send

Site Work in Progress - Best viewed on the desktop