Whitepaper - Next Generation Firewalls

Table of Contents

Chapter 1. Some unssorted coments

- Scale up vs Scale out
    - use cases for centralized firewalls
        - IIIS as example
            - scale up reaches limits with rule bases too large
                - requires scale out on top of scale up
    - use case for decentral firewalls
        - existing CF as example
            - Can be vistualized and scale out inside a HW
                - HW can also scale out
- Distributed vs Central Choke point
    - central Choke point requires high performance HW with high costs
    - Decentral requires spare capacity in clusters of compute or dedicated clusters
        - can be part of hypervisor (uncommon)
        - can be special VM in cluster (common)
        - can be stand alone and not mixed with compute (scale up limits)
        - cloud capable but looses some features that rely on custom silicon (FPGA/ASIC/ETC)
            - tend to be higher latency than HW based Firewalls through serialization delay on limited number of CPUs
            - tend to support subset of features or limited combination of features
- HW vs Virtual
    - Virtual matches cloud OPEX model
    - Virtual Fw partially improves ISSU features
    - Virtual Allows in band distribution in Compute env.
    - Virtual in individual functional FW context allows for routing domain separation which is required to deal with overlapping RFC1918 address space
    - Virtual Tends to be both feature and throughput limited by number and type of CPUs and memory and HW bus of commodity platform
        - example IPSEc limitations on number of concurrent tunnels etc
    - HW tends to scale up better
        - requires higher initial CAPEX and matches traditional CAPEX model
    - HW can also scale out but often that is not required. (up and out vs jsut out with Virt)
    - HW can include
- NAT vs next gen.
    - a standard feature for years in V4 to combat address shortage through use of RFC1918 space

  - used primarily to work with shared applications reaching overlapping routing domains.
    - also Soho to Internet
  - used to bridge differences with cloud backend addressing (v4) and client V4
  - Used to abstract routing domains (hiding)
  - noT supported in V6
    - V6 requires different approach (BEWARE)

Chapter 2. Definition of Next Gen Firewalls

- Micro segmentation
- Distributed
- HW accelerated
- Virtual
- SDN (central control of de-central)
- Platform X (commodity compute)
- application aware
- encryption handling strategies

Chapter 3. Advantages of distributed Next Gen Firewalls

- Thrhoughput scales out over the environmet so that they can handle capacities much higher than a central choke point in sum
- Decentral
- better currency than traditional
- potential for partitioning in security zone (micro segmentation)
  - can be in addition to central choke point (augment with less features)
    - example ACI ACL is no replacement for a stateful firewall but it is an augmentation and provides security in depth.

Chapter 4. Disadvantages of distributed NExt Gen Firewalls

- Throughput does not scale up on individual nodes only out across the environment
- Decentral - logging can be problem
- Micro segmentation can be contrary to traditional security processes and policy and tooling etc. see below.

Chapter 5. Applicability of Next Gen Firewall Technology

- Zone based Rules

Chapter 6. Impact to security standards

decentralization / Distributed

- classical design assumes central choke point
    - distributed firewalls CAN mean distribution of rule set so that features and rules are only deployed where required.
        - Policy checking needs to be done at central configuration point.
            - Trust in distribution algorithms

Micro segments

- Use of micro segmentation requires deeper knowledge of application function.
- Micro segmentation when done right should be same zone partitionung and not multi zone in subnet.
- Zone based Security standards like the GSNI security standard allow for micro segmentation but assume that a zone is a subnet or group of subnets.
    - firewall policy checking becomes impossible with mixed zones in one subnet.
        - only assisted checks are then possible as the flow complexity gets too high.

Chapter 7. Tooling considerations

Every platform brings with it a set of tools and automations and APIs that can be used at both a tooling or a device level.

7.1. Vendor tooling

The tools each platform brings are potentially those we'll only use as an aside. The tools we should be using every day are the ones that can be used to push data to multiple platforms.

7.2. Overarching toolsets

Other tooling like automated firewall policy checking etc. also needs to be able to reach all platforms eventually. Time and resources being constrained the choice comes down to investing in multiple vendors tools or central tools or a mix.

Easy integration will make the difference.

If a device is easily integrated into a central solution the vendor tooling can be used in parallel.

If Vendor tooling is more easily integrated into a central solution than the individual devices then this is to be preferred.

7.3. Firewall rule revalidation

Firewall rule revalidation is also a major consideration as this makes up a large part of recurring work. One thing we also need to consider is potentially being able to keep control of rules across platforms so that integrating each vendors mgmt. platform might not be possible and it might be better to, for example, use those to cross check and do manual interactions but to do the daily work across ansible or something equivalent that takes input from whatever change mgmt. tooling the rule requests get pushed through.

7.4. Tooling summary

- For tools like change management and firewall rule revalidation it makes a lot of sense to use central tools.
- For device specific configuration tasks it makes a lot of sense to use the vendor provided tools.
- Vendor tools should only be used as a path from central tools to devices if integration of the devices directly for central tasks is more costly.

Chapter 8. recommendations

In general based on different environments different approaches are recommend

8.1. Use case central choke point in colo

Example is a central access firewall to bridge from blue to yellow (forgive the old fashioned colour)

- Scale up and out
  - HW not virtual
    - should support routing domains and partitioning even though Blue is just one domain…. ah no it's two (V4 and V6)

8.2. use case Customer firewall traditional DC

- scale out
  - NAT required to maintain traditional environment without modifications
  - virtual instance
  - medium sized hw not large hw

8.3. use case internet

- Scale up if in traditional DC
  - scale out not often required but possible based on destination ranges in local DC or cloud
- HW acceleration for IPSEC a boon if mixing transport and internet access
  - transport vs internet access are different use cases as such but can be mixed if careful
    - potentially separate instances to allow for easier inspection
      - Also easier to limit Internet without pulling plug on transport
- NAT not normally required
- V6 a must
  - here customers care more about V6 as it's client facing
- WAF
  - traditional firewall effectively is without function here as most attacks are application based.

8.4. use case cloud

- commodity HW
  - virtual FW on bare metal (limited scale up)

    - virtual FW in cluster (scale out)
    - Mixed use case with backend, frontend, and middle firewall.
        - can require different mix of software firewalls in compute and on bare metal.
        - more use of BGP etc to traditional firewalls.
        - more use of IPSEC/GRE etc

8.5. use case tooling

- low bandwidth
- single routing domain for tools
    - multiple routing domains if shared tooling to clients (multiple client domains)
        - NAT or V6
    - less attack surface as normally already behind further outward fw
    - micro segmentation can be boon in addition to central choke point to prevent spread of malware in same security zone through partitioning
        - combination of fabric based and traditional firewall

8.6. Use case backup

- High throughput
    - LAN to LAN (local DC traffic)
- overlapping IP ranges (if cloudy software defined setup to allow traditional client to come in with own IPs - recommended)
    - approach with RFC1918 ranges or public IBM IPs require an architect to create a solution based on the context of the client routing domain overlap with IBM and other clients
    - or use public IBM V6 if IBM manages the backup and client! Better! use native not dual stack.
        - NAT(traditional approach) OR V6(preferred)

Appendix A: UNSORTED Brainstorming

1. the "potential problem"
2. the risks
3. the impacts
4. how we should respond
5. how we should prevent

A.1. MFA

[NextGenFWWhitePaper b830a] | *images/NextGenFWWhitePaper-b830a.png*

Figure 1. Multi Factor Authentication across Stepping Stones with DUO

The above figure shows an example of firewalls in context with jump servers and dynamic rules and conventional static firewalls.

A.2. Putting on The BLACK hat and Brainstorming after notpetya. A new age of Malware?

A real nasty would follow Established flows with credential stealing like the NotPEtya but slightly worse by getting into admin networks and then working from the top down. The attack would be on infra only and not on physical or Vms as such. Hypervisors, routers, etc. One in those the flows could quickly be collected and as it attacks admins would be trying to solve it hence making it worse. Also one thing to keep in the back of your mind right now is that the above may be triggered with a small attack that seems easy (or hard) to solve. And while solving it's acctually preparing the larger nastier surprise at the end of the road… A timing factor would play into this. Either having two unrelated attacks so that antivirus etc only look at the current one. or doing it within a day to be ahead of the analysis. One attack doing damage and one that does nothing nasty and just collects and follows flows. I'm not 100% sure how to totally evade being seen but in the midst of an attack is a good idea. Leaving admin workstations in tact is then also a good idea. So targetting web servers would probably be the best tactic to keep one attack apart from the rest. Primary aim of misleading attack would be servers only. Secondary total control attack would target only jump servers and the like. Potentially also firewalls but that's something I think isn'T even needed as with the jump servers the firewall is out of the picture. The atack rate should be slow and low with credential stealing the primary aim till a trigger hits. The next item of interest is if it should be done with coordination or clocks synchronized and shared nothing. Both are possible. Assumption is that admin workstations have access and hence can also reach one another. Jumpservers are assumed to be locked down so comms would be through admin workstations potentially as peer to peer or via some means of DNS or similar requests through the proxy of the admin stations. Clocks synchronized would mean setting an attack time up front. dis-similar to not petya setting the 40 min timer to first spread and then encrypt. So essentially this would be slower and in the range of hours as admins will need time to check all the systems one by one. But could also be triggered either for a set time or an offset from infection. Probably best set as a fixed time and offset as backup in case the infection is later than the fixed time or it's too far away. Offset probably about one day at most and at minimum 8 hrs. If past the fixed time it would be shorter as AV would be on to the attack. the less time spent on mucking up these routines the better. This part should be bullet proof as it wouldn't help to fail here. The main problem will be keeping stuff in the path from crashing and popping alarms bells on. So a fine line has to be walked in that unknown targets should not be popped open at all costs so as to avoid popping the clogs off something that points to admin traffic as the culprit. Take into account that if two or more attacks are running with only one or two in the foreground then the AV will concentrate on this. Potentially setting one or two triggers on stuff the AV will check for will allow the low and slow to trigger also when AV eventually figures out the kill switch on the misleading attack or the binaries used. The low and slow should therefore check if the primary is being scanned and then change it's tactics. Central comms could be instrumental here or just pop the clogs on that machine immediately. A Nice ruse might be to diguise as malware but and to actually collect ransome not like notpetya.. BUT to leave a vulnerability in it so that the AV folks spend time working on getting decryption routines working further detracting from the low and slow… if that wins an hour or two all the better for the secondary attack

If we're doing all this it might be good to have tools available as watering hole attacks for those who try to grab stuff to help. This is one risky avenue for the attackers but it might fit well if it's proven to decrypt the disk and fits with the flaw that one could install. This is something that might quickly explode in the attackers face though unless it can quickly be pushed to the top of the google search lists as it's likely to get looked at quickly by AV guys to up their own game and they're not likely to be infected by it. That tool would have to be propagated to all systems by admin workstations and would therefore be a further in-road and potentially a further infection vector using the pressure of the misleading attack. I'm not sure this one is still as good an idea as it would have been a few years ago but it's worth thinking about as a risk. The question is how to get that tool to behave well enough so as not to get flagged immediately. Or maybe that doesn't matter and it's designed to get a few of the stupids and is kept separate from the other attack inroads and adds to the overall mayhem. Hence if this is designed to cause utmost damage it probably doesn't matter if it get's flagged as bad as that will increase the overall nastiness and potentially still catch a few unaware while taking more attention off things the attacker wants to keep going till the final endgame.

The next nice thing to add to the fray would be the damage section. If we know that the network will be shut down we can potentially monitor for it by looking at the established connexions and seeing them all go stale. That way we don't need to probe the network. When that happens the routine would know to perform the end game. It make a lot of sense to get the end game running smoothly so as to have it run with minimal fuss and quickly. Notpetya encrypted some files and then added the final nail by doing the master file table but after a reboot.. why bother with the reboot? potentially to get running without AV and stuff getting in the way. Potentially this takes too long. If the aim is an attack then what matters is destroying trust in the data as opposed to destroying it outright. It might make sense to have routines that modify data in certain file types and plain overwrite others. I think with large file it makes sense to look at destroying structure as opposed to the acctual data. So the routing might not bother doing much damage but concentrate on doing structural damage if pushed and doing complete damage if left to work. so a bit like petya/notPetya doing damage to certain file types up front and then taking out the rest last. If there are multiple attacks running in parallel they would also have different aims so those routines should be written in different tools and ways and have no obvious code overlap so as to avoid detection of one through detection of the other. I think the most we should expect is 2 to three forms as the development effort is too large for more. Also it would require too many in the know unless state sponsored. A small team could easily create an admin attack with another team creating the obfuscation attack to help amplify the admin attack. A third watering hole attack could be written on a budget as the propagation method would be the admins themselves so it doesn't need propagation methods. It needs to keep it's head down and needs a trigger of sorts but otherwise would look to be decrypting potentially very slowly till the main event kicks in. So let's say it starts the decryption and begins on small file or selected file and goes slow. That would seem acceptable if it slowly provides results and can then be left to run on it's own while checking it's back and awaiting the crescendo so as to perform the final back stab. It might of course help if it also scans for infection….. Happiness more systems to deploy on. Potentially doing nastiness to AV servers as a further vector but that one is quite risky so might not be a good idea and might be one step too far and out of reach of a small to mid sized team hence limiting it to real real nasty. But then again if we work on the assumption that this is state sponsored it would do well to look at what Russia is asking for for AV vendors to operate on Russian soil.. They ask to see the source code. That makes the scenario of hitting AV as a target or even a vector a more realistic possibility for an amplification attack. I'm wondering if we should treat each one separately or consider them in combination as a worst case. I think the latter is the best way to protect from each. The cost to protect rises though and trust is reduced in AV and other tools we currently use to clean up hence causing us to have to think about independant means of cleaning up and detecting. Potentially out of band detection in combination with detection on the boxes in combination with a new backup monitoring to prevent large scale backup destruction or tampering to add to boot. the backup systems would be fair game as admins would check those also. So if for instance the jump servers are a propagation vector then the backup servers become a primary aim in order to maximise damage. Credential stealing or even session hijacking would make that simple as pie. Essentially needing something to handle RDP and SSH key logging on jump servers would allow the baddy to see what's going on and to thereby see what kind of system is being dealt with. then allowing it to pick the best method to attack. Backup being a juicy target and even a low hanging fruit as it's easy to do a lot of damage there in short order by deleting indexes etc. The backup servers databases as opposed to the tapes would be a primary target. Tapes would take too long to do in as if one thinks about a large setup the ratio of tapes to drives alone would just wind up taking too long to do max damage that way. destroying the index would cripple any recovery in short order. I need to go over this and put some of my notes form a few years ago in too. I think stuff like team size and attack types possibly might give early pointers to what we're dealing with. With notpetya the email address and the way the decryption was to work raised an early flag as to what WTF is this doing? There should be lots of such pointers that could be looked for so as to raise warning flags. Another warning flag as to notpetya not being run of the mill was that it really didn't try that hard to keep the victim happy. But then again maybe that's good as I get the creeps visiting pages I know are written by those who are attacking me in order to extort money. Wouldn't it be good to have a page seemingly unrelated to look at instead. Need to give them something to search for so as to pimp results up front or fast. Might be some string in the ransom page for instance that is a good handle to search for on google. Potentially a snappy name that isn't easily miss-spelt to lead to a neat watering hole. notpetya is a wonderful name. How about notnimda or notcodered? just some examples. Notiloveyou or in fact some miss-spelt word on the ransomware splash screen that lends itself to easily search for or some l33t word but that might be too much. maybe a name with attached date like ransom-june-2017. whatever. enough criminal thought to work on protecting from it. Essentially need to be VERY careful not to fall flat into a watering hole.

Back to jumpservers and client comms. The jump server often breaks the encryption or is the end of a tunnel. It may not always have visibility into the data stream. So this is a two edged sword. It may be a good ingress point and it may not depending on how it's used. One thing is clear that the admin workstation initiates traffic so it's the best place to grab credentials. Potentially the jump servers are then a good vector to map the admin estate and the server estate without being able to attack directly from these so using them might require feedback to central systems that may be detected. but packing it in standard admin flows will make that detection harder. There are ways of using traffic analyses to tell humans apart from machine comms so that might be something one should put into place. Essentially the system checks not the content but the size and timing of comms to see if it's regular or interesting traffic. I'd hate to have to work against a determined team made up of dedicated good programmers equipped with malice and insider knowledge and potentially motivation to do real damage. Add to that some bastard who holds it all together and it's a bad mix. Especially if the aim is not to gain funds through the attack and if this is well funded and all the families and the team are well looked out for and have zero to worry about. also if the team is small enough to be well knit and is equipped with the knowledge that they are doing country/religion/whatever a huge service.

If that team is looking for insider knowledge what better place to look for it than for ex employees who are disgruntled and out of work after massive layoffs due to outsourcing. Happy hunting but good thing that most people are good enough not to do nasty stuff. Building an IT team out of such people would be interesting. It has challenges though so not likely in my eyes but worth discussing. I wonder how that can be prevented. I also wonder if outfits like the IS can easily catch disgruntled employees. Or if it's easier for state sponsored teams to do so. IS being a sort of state but appealing to a different audience I think. People supporting a family are less likely to up and go working for IS. They might be interested in working for a firm that will guarantee that they and their family is well taken care of without having to cut other's heads off. A nice pension scheme etc could help added with a bit of help the homeland as opposed to hurt others as primary target. I wonder what implications that has for the employer and employees and if it makes them detectable in the mid to long term. Example is The stuxnet team: why don't we see them even after the reasonably long time since the attacks. Are they happy? Probably. No point in saying it was me as that would only cause attention and hence potetntially loss of pension etc.. So you don't need show offs in that team and need to prevent them from wanting to show off.

Re segmentation: Yes a very good idea. would also need to isolate Admin jumpservers from reaching normal stuff. So hypervisor should not be managed by workstation that manages customer images. Potentially through separate jump servers as in GSNI with IR and SR segments. But also on customer site. Two types of out of band at minimum. service facing admins and infra facing admins. Admins for low level stuff and infra needing rock solid authetication and infrastructure services. Not sharing fualt domains between infra and service. in the case of APMM for example the switches use APMM AD for access…. that is sort of good and sort of bad. we should cut that out. Finding for APMM review.

Restore considerations: Incremental backups take ages to restore. may be able to consolidate those before restoring while in between incident and restore while collecting information and making plans. Other ideas need to be added here. Like reducing backup schedules for restored systems and how to deal with unafected systems to make sure they are not pulled down by running out of backup volume.

Another ineteresting thing also done in the past on local hosts was to bend the DNS.. why not do this centrally if we own the AD and or DNS DHCP servers. That way either preventing certain updates or bending them to something of our choice (our=nasties). how do we prevent that and what advantages doe it have for an attacker. preventing updates or slowing them down is one thing that might be an aim. But then again an aim might be to misdirect servers to update from a tainted source.

Talking about server is also leading away from the infrastructure question. what is critical infrastructure? The routers and switches are, so hitting them would also cause a few days of outage. That could easily be done through credential harvesting and hitting them through central systems in the out of band network. how easily and what do we need to do to line that up? I think that is an interesting thing. Potentially looking at access patterns through central logging of access could slow that down but maybe not. I think not. Question how do we make absolutely sure that the entire estate doesn't get firmware erased remotely? And if we can't prevent it how can we get it back up quickly. If network firmware get's flashed it's a case of sending people on site with flash drives and terminals. That should be prepared and can be done in parallel in theory.

Backup might also be a central ingress point. If you get in there you can destroy it and attack the servers performing the backups. That is also a central fulcrum and should be segmented also. N.B. The central question is which single point of failure services and apps do we have. mgmt included. Potentially looking at graphing all this and then looking at it through the node cluster lens. There are quite a few clusters of nodes and in theory all of then have to be partitioned. the methods of partitioning will be aimed at separating the application node clusters with the separation going all the way down the chain to layer 2. Graphing this would be a service and I think that's worth a patent or two if combined with the partitioning logic and methods. I had started writing a patent on something like that a while back with Bertus Eggen. And another one allowing one way logging from an air gapped network. This seems all the more relevant now.

propogation mechanisms would be interesting and important to understand

One method is to hack a vendor potentially an AV vendor (hard) or some other vendor with lots of software out there and push an update with the payload. then trigger it later with another upload. I wonder would ACI and microsegmentation have prevented this. I think not as it used allowed microsoft flows to get around (notpetya) Hence segmentation has to be done above the network layers at layer 7. firewalls also will not help here. what might help is segregating through having real separate routing domains and having certain apps/servers not being ale to reach anything but the minimum to work. Also fencing third party apps off. A DR site then needs to be on a different update cycle to keep out of the danger zone. But that puts it in another danger zone so it would mean keeping it disconnected with out of band data synchronization so as to avoid getting infected in-band with executable code through API calls.

I did some segmentation of routing domains for EY in a previous design to ensure that tooling was kept apart. That caused initial consternation of both firewall and server and tooling teams but in the end ensured that tooling at least was kept apart and failure of one site would not have the tooling on the other site dependent on it. BUT here's the crux AD had to be cross site because it's central. So AD is a problem here and it's what was used in this (notpetya) attack as the fulcrum. So this is a risk in which central software is the central point of failure/infection. That means replicating that out to a separate site/network and keeping it out of sync. I wonder is it possible to put a one day delay on that kind of replication. ie that one can then stop the replication or delay it if something happens as it will be in transit for a set amount of time before hitting the DR system. Maybe worth a patent? Essentially the equivalent of bouncing the backup off mars to add time delay. The equivalent of really slow mercury delay lines as it where. then when the shit hits the fan (in this case malware) hitting the pause button and potentially bringing the Dr systems up with older data while the newer data is still paused in transit and potentially the badness is caught in flight and the arrow doesn’t strike it’s mark. Might already be a patent :-) I’m sure we’re not the first doing this line of thought. I had this idea back in 2001 instilled by the first audit experience. Joking idea was to send all the nasty files to a mirror on mars and hence claim it’s not reachable only to get it back after it bounces. It may be an idea here.. but without mars. the good old tapes being sent offsite was a good idea in this case and would have helped if the restore was always being done but time delayed in the DR site. That could be a nasty wear on material though. SSD would have problems with that type of behavior potentially (maybe (needs consideration)). That would be a DR concept with a twist. But we already have issues getting customers to do DR so the target audience is small unless we can do it in the cloud or whatever. That would be good. A bit like the old BCRS idea or rolling in a replacement mainframe and installing the backup on it. In this case having a small private or public cloud with then lined up restores in FIFO order with a long enough pipe to handle the delay.

A.3. Micro segmentation

I have a "general vision" that we need to completely reconsider how we manage our internal zones. eg. we have lots of subnets, but no isolation between disparate systems: Workstations, can talk to Domain Controls, and to File Servers, and to App Servers, and to Mail Servers For all the subnets, just one big flat intranet / LAN

A.4. Summary

So - looking into a new age of malware - do we need to rethink this and look at Firewalls and/or application aware firewalls between logical services