



Please give me back my Network Cables!

On Networking Limits in AWS

STEFFEN GEBERT
MIKLOS TIRPAK

SRECON 2025 AMERICAS
2025-03-26



What's this?

- "100G network cable"



100G Ethernet Throughput Limits

- 148,809,524pps!
@64 bytes/frame
- Per direction!



Cloudy Throughput Limits



EC2 Instance Types

Throughput Limits

General Purpose m7g

Instance Size	vCPU	Memory (GiB)	Instance Storage (GB)	Network Bandwidth (Gbps)	EBS Bandwidth (Gbps)
m7g.medium	1	4	EBS-Only	Up to 12.5	Up to 10
m7g.large	2	8	EBS-Only	Up to 12.5	Up to 10
m7g.xlarge	4	16	EBS-Only	Up to 12.5	Up to 10
m7g.2xlarge	8	32	EBS-Only	Up to 15	Up to 10
\$950/mo	16	64	EBS-Only	Up to 15	Up to 10
m7g.8xlarge	32	128	EBS-Only	15	10
m7g.12xlarge	48	192	EBS-Only	22.5	15
m7g.16xlarge	64	256	EBS-Only	30	20
m7g.metal	64	256	EBS-Only	30	20

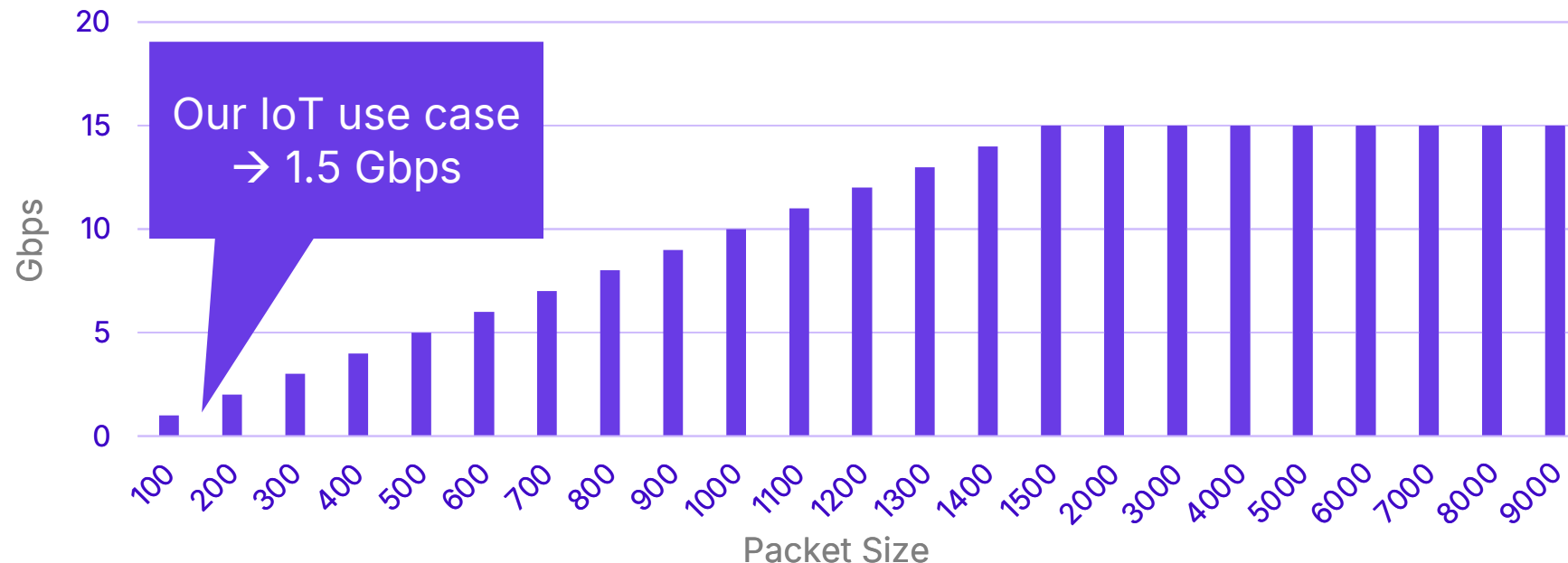
Compute-optimized + network-enhanced c7gn

Instance Size	vCPU	Memory (GiB)	Instance Storage (GB)	Network Bandwidth (Gbps)***	EBS Bandwidth (Gbps)
c7gn.medium	1	2	EBS-Only	Up to 25	Up to 10
c7gn.large	2	4	EBS-Only	Up to 30	Up to 10
c7gn.xlarge	4	8	EBS-Only	Up to 40	Up to 10
c7gn.2xlarge	8	16	EBS-Only	Up to 50	Up to 10
\$1450/mo	32	64	EBS-Only	50	Up to 10
c7gn.8xlarge	32	64	EBS-Only	100	Up to 20
c7gn.12xlarge	48	96	EBS-Only	150	Up to 30
c7gn.16xlarge	64	128	EBS-Only	200	Up to 40
c7gn.metal	64	128	EBS-Only	200	Up to 40

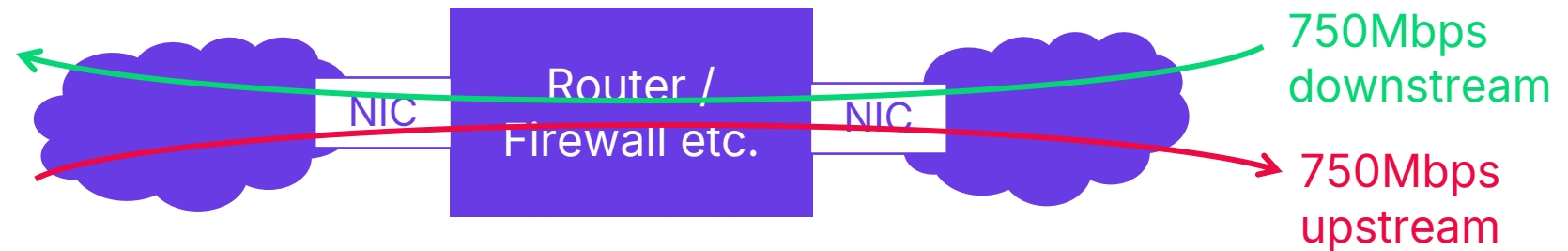
Real 15 Gbps – or again “up to”?

Throughput Limits

- AWS limits throughput based on
 - Bandwidth (data rate): 15 Gbps regardless of packet size (default MTU is 9001 byte)
 - Packets per Second (PPS), i.e., rate of 1500 byte packets to reach 15 Gbps: 1.25Mpps



Network Functions Throughput Limits



With a 15 Gbps instance, we can transfer bi-directional 750 Mbps - peak!

One Note...

Disclaimer

- We like working with AWS, it makes our life so much easier!
- But we want to share our journeys through sleepless nights etc.

Thanks to..

- Our colleagues, who went with us through this
- AWS account team and network specialists

Limits are critical in multi-tenant environments – they protect (also us) from noisy neighbors!

emnify IoT SuperNetwork

+ Cellular connectivity for the Internet of Things (IoT)

+ Global IoT SIM card

 Headquartered in Germany
 Engineering EU remote

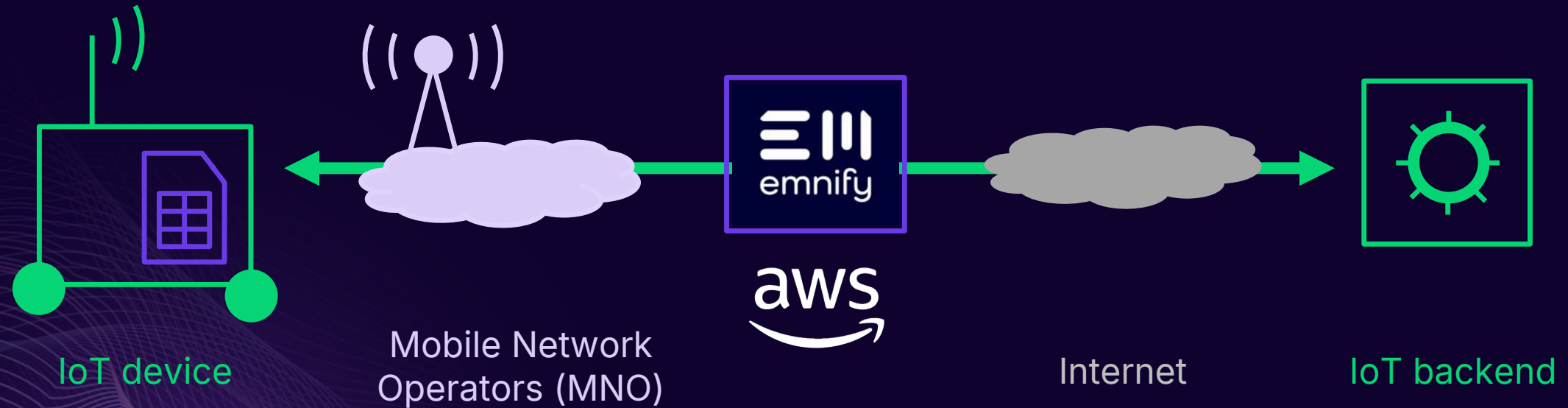


+ Deployed in 7 breakout regions

+ IoT-specific security features

+ Integration into customer networks

Why are AWS limits so relevant for us?





Unpredictable traffic

Customer pays for the traffic

EC2 instance type economics



 customer-owned

 emnify-contracted

 emnify-owned

 nobody owns it

Burstable Instance Types ("up to X Gbps")

Throughput Limits

- Documentation is clear about baseline / burst bandwidth
- Credits-based mechanism
- On a best effort basis

Network specifications

Note

C7g instance types support configurable bandwidth weightings. With these instance types, you can optimize an instance's bandwidth for either networking performance or Amazon EBS performance. The following table shows the default networking bandwidth performance for these instance types. For the supported configurable weightings, see [Configurable bandwidth weighting preferences](#).

Instance type	Baseline / Burst bandwidth (Gbps)	EFA	ENA	ENA Express	Network cards	Max. network interfaces	IP addresses per interface	IPv6
C7g								
c7g.medium ¹	0.52 / 12.5	X	✓	X	1	2	4	✓
c7g.large ¹	0.937 / 12.5	X	✓	X	1	3	10	✓
c7g.xlarge ¹	1.876 / 12.5	X	✓	X	1	4	15	✓
c7g.2xlarge ¹	3.75 / 15.0	X	✓	X	1	4	15	✓
c7g.4xlarge ¹	7.5 / 15.0	X	✓	X	1	8	30	✓
c7g.8xlarge	15 Gigabit	X	✓	X	1	8	30	✓
c7g.12xlarge	22.5 Gigabit	X	✓	✓	1	8	30	✓
c7q.16xlarge	30 Gigabit	✓	✓	✓	1	15	50	✓

Note

¹ These instances have a baseline bandwidth and can use a network I/O credit mechanism to burst beyond their baseline bandwidth on a best effort basis. Other instances types can sustain their maximum performance indefinitely. For more information, see [instance network bandwidth](#).

For `32xlarge` and `metal` instance types that support 200 Gbps, at least 2 ENIs, each attached to a different network card, are required on the instance to achieve 200 Gbps throughput. Each ENI attached to a network card can achieve a max of 170 Gbps.

Monitoring Throughput Limits

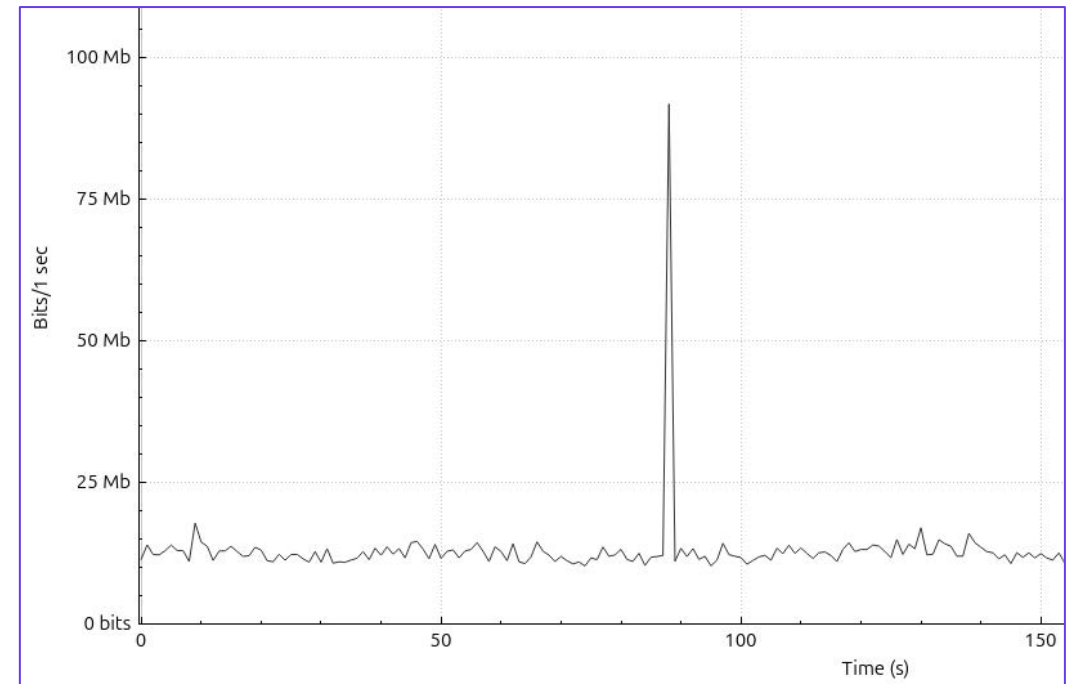
- Metrics
 - `bw_in_allowance_exceeded`
 - `bw_out_allowance_exceeded`
 - `pps_allowance_exceeded`
- Indicate queuing (and potential packet drops)
- ~~Amazon CloudWatch~~
- `ethtool -S ens5` to read from NIC driver (ENA)
 - CloudWatch Agent install + configuration
 - Prometheus *node_exporter*, with *ethtool* collector (disabled by default)
- Monitoring interval: Beware of bursts

Microbursts

Throughput Limits



- Packet drops on a c6i.xlarge with very low traffic
 - Bandwidth utilization is 15 Mbps
 - `bw_out_allowance_exceeded` increases every 5 min
- File upload to S3 every 5 min
 - File size: 9 MB
 - Data transmit completes within 26 ms
 - → 9 MB/26 ms = 2.7 Gbps peak


Instance type	Baseline / Burst bandwidth (Gbps)
c6i.xlarge ¹	1.562 / 12.5



Microbursts

Throughput Limits

 AWS Support

 Hello! We're here to help.

Amazonian: You write that you observe *bandwidth out allowance exceeded* metric increasing on an instance that is barely used, which should have enough network credits.

EM: Yes, metric increases every 5min.

Amazonian: Your packet size is only 1500 byte. Can you add a VPCe?

EM: Okay, now it increases even faster.

Amazonian: Probably microburst on PPS limit.

EM: The PPS metric is not increased.

Amazonian: Scale up the instance!

EM: No, you won't get more money!

- Limits are implemented much more granular than per second.
- `bw_out_allowance_exceeded` does not always mean dropped packets, only enqueued.
- Make sure you monitor your application behavior.
- "Bandwidth is a shared resource"
Instance burst is on a best effort basis, even when the instance has credits available, as burst bandwidth is a shared resource.

Countermeasures

Throughput Limits

Increasing Limits

- ~~Add more network interfaces?~~
- Change the instance type
 - Scale up! 📈
 - Use network-enhanced types 📈
 - Use newer generation (7th gen increased limits)
- EC2 instance bandwidth weighting: 25% more network capacity (requires. 8th gen instance)
- Use jumbo frames inside AWS or via DX


Avoid Microbursts


- AWS-CLI S3: `default.s3.max_bandwidth`
- `SO_MAX_PACING_RATE` socket option for own applications
- Limit bandwidth using `tc`
- Increase the Tx ring size with `ethtool`

Issues

Fragmented Packets

- `pps_allowance_exceeded` increased, while far away from the PPS limit

aws  AWS Support

 Hello! We're here to help.

Amazonian: You write that everything is fine but you have packet loss metrics increasing.

EM: Yes, totally.

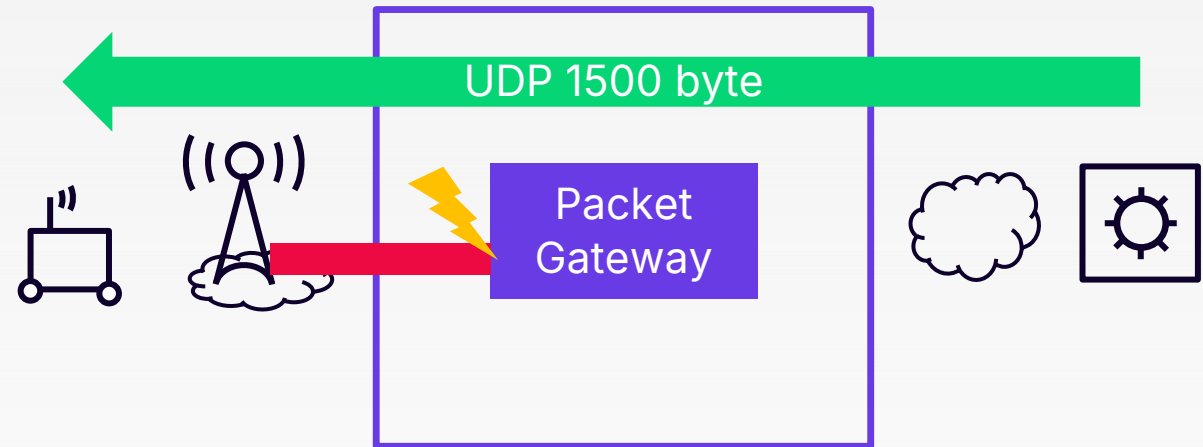
Amazonian: Fragmented packets?

EM: Oh, yes! How many can I?

Amazonian: Can't tell ya

EM: Hold my beer. Like 1000?

Amazonian: Exactly. And btw. you won't get more when you scale up.



Understanding Fragmented Packets

- Not using Nitro's hardware acceleration (fast path)
- Differs between
 - Ingress: Nitro standard rate (slow path)*
 - Egress: 1024 pps (intentionally slow path)
- Same limit (1024pps) as link-local traffic, but different bucket.

Application design considerations

There are aspects of application design and configuration that can affect your processing efficiency. The following list includes some important considerations.

Packet size

Larger packet sizes can increase throughput for the data that an instance can send and receive on the network. Amazon EC2 supports jumbo frames of 9001 bytes, however other services may enforce different limits. Smaller packet sizes can increase the packet process rate, but this can reduce the maximum achieved bandwidth when the number of packets exceed PPS allowances.

If the size of a packet exceeds the Maximum Transmission Unit (MTU) of a network hop, a router along the path might fragment it. **The resulting packet fragments are considered exceptions, and are normally processed at the standard rate (not accelerated).** This can cause variations in your performance. However, you can override the standard behavior for outbound fragmented packets with the fragment proxy mode setting. For more information, see [Maximize network performance on your Nitro system](#). We recommended that you evaluate your topology when you configure MTU.

Monitoring Fragmented Packets

- `pps_allowance_exceeded` increased, while far away from the PPS limit
- Running `tcpdump 'ip[6] = 32'`
- Ask AWS support!

Countermeasures

Fragmented Packets

Avoid Fragmentation

- Don't fragment!
- VPN / encapsulation use cases
 - Lower MTU of tunnel interfaces: Fragment the inner packets (hide them from AWS)
 - Make sure path MTU discovery works
 - TCP MSS clamping

Increase Throughput

- Move fragmentation to TGW, if applicable
- Use ENA's fragment bypass support (v2.13.3, released last week)
 - `enable_frag_bypass`
 - Increases egress throughput from 1024 pps to the "standard rate" (not accelerated)
 - Competes for Nitro CPU resources



Conntrack

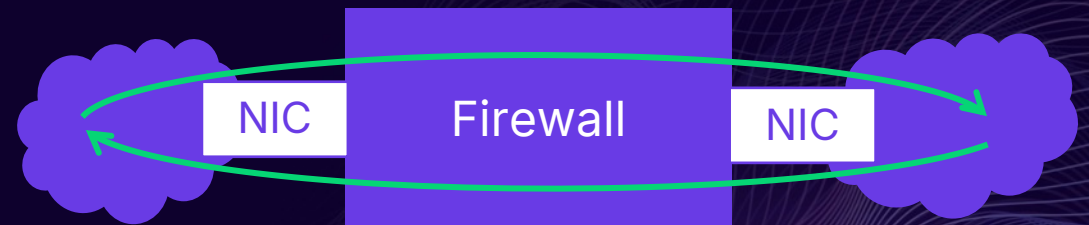
CONNECTION TRACKING

Introduction

Connection Tracking

State of Connections

- Stateful firewall:
 - Allow the returning flow of a connection
 - Similar to NAT table, but slightly different
- In Linux, mostly using `iptables`



Source IP Address	Source Port	Destination IP Address	Destination Port	Protocol	State
1.2.3.4	54321	5.6.7.8	80	TCP	ESTABLISHED
1.2.3.5	54322	5.6.7.8	80	TCP	SYN_SENT
1.2.3.5	43210	6.7.8.9	123	UDP	SEEN_REPLY

Introduction

Connection Tracking

State of Connections

- Stateful firewall:
 - Allow the returning flow of a connection
 - Similar to NAT table, but slightly different
- In Linux, mostly using `iptables`

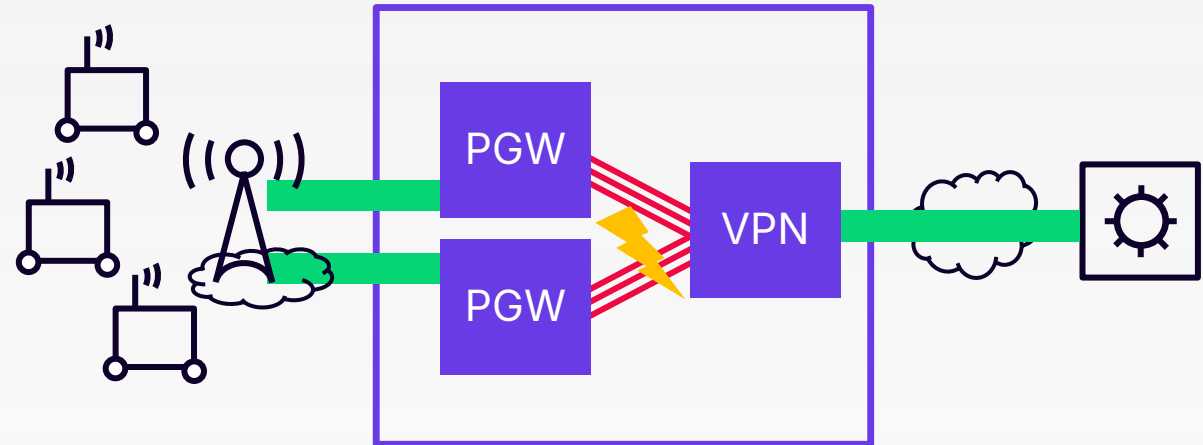
AWS

- Security groups: What makes them *stateful*
- Limited number of entries
 - Varies per EC2 instance type
 - No official documentation of limits

Consequences

Connection Tracking

- New TCP / UDP connections are blocked
 - Incoming connections might fail
 - DNS might (sometimes) not work
 - AWS SSM might (sometimes) not work



Monitoring Connection Tracking

Capacity Exceeded

- `conntrack_allowance_exceeded`
- since 2021

Remaining Capacity

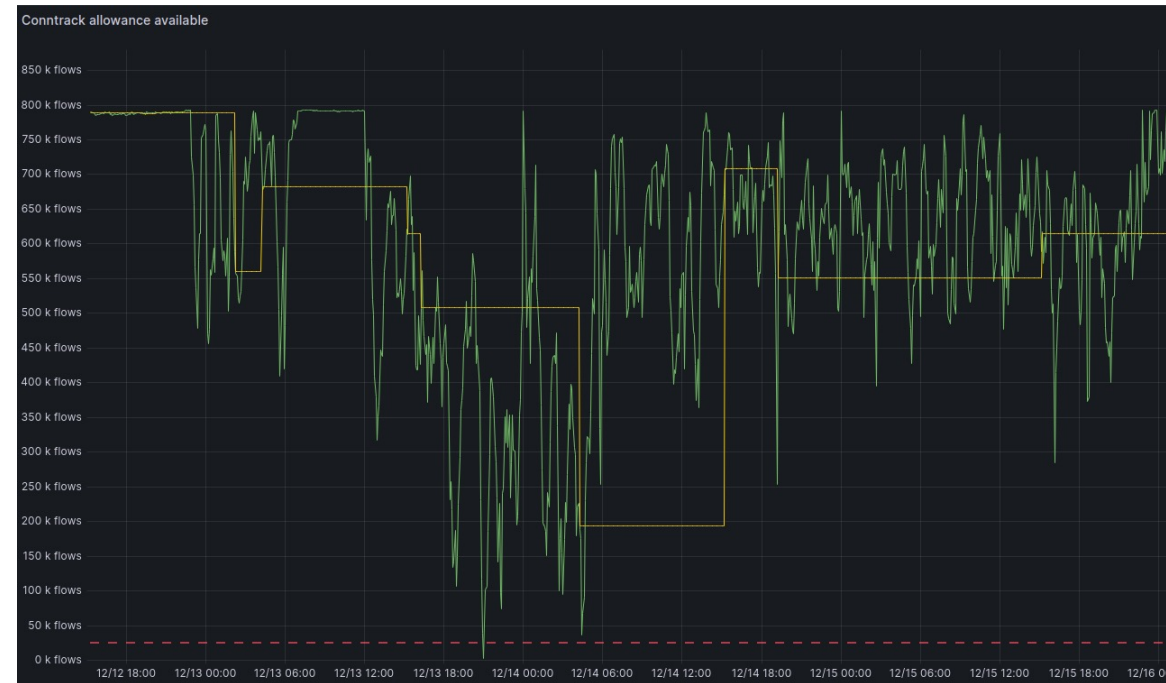
- `conntrack_allowance_available`
- since Jan 2023

	c5	..n	c6g	..n	c7g	..n	c8g	..n
large	136k	136k	153k	153k	153k	205k	153k	N/A
xlarge	273k	273k	307k	307k	307k	410k	307k	N/A
			+12% compared to previous generation			+33%		?

Monitoring Challenges

Connection Tracking

- Requires young enough ENA driver version, not included in Ubuntu 24.04 LTS. We have own version compiled.
- Counters to be interpreted differently
 - *Exceeded* counter: refused connections per interface (use the *sum* of all interfaces)
 - *Available* counter: each interface shows the same value, the remaining for the EC2 instance (use *avg* or pick a single interface)
- Observed different values for the available counter on different interfaces – only 1st interface was up-to-date (bug #291, fixed)
- Flow logs can help, but setup is cumbersome



Countermeasures

Connection Tracking

Disable connection tracking

- Have a rule in the reverse direction allowing traffic from all IP addresses¹:

Ingress Rules		Egress Rules	
Local Port	Source IP	Destination IP	Remote Port
80	0.0.0.0/0	0.0.0.0/0	*

- Network functions, imagine NAT instance

Ingress Rules		Egress Rules	
Local Port	Source IP	Destination IP	Remote Port
*	0.0.0.0/0	0.0.0.0/0	*

- **Beware of the security implications!**

¹) EXCEPT "AUTOMATICALLY TRACKED"

Automatically Tracked Connections

INTERNET ARCHIVE
Wayback Machine
36 captures
27 Feb 2021 - 5 Mar 2025

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/security-group-connection-tracking.html

AWS > Documentation > Amazon EC2 > User Guide for Linux Instances

Amazon Elastic Compute Cloud

User Guide for Linux Instances

Recently added to this guide

- [Preview](#)
- [Prerequisites for Capacity Blocks](#)
14 February 2025
- [Reference AMIs using Systems Manager parameters](#)
10 February 2025
- [Reference the latest AMIs using Systems Manager public parameters](#)
10 February 2025

Automatically tracked connections

Connections made through the following are automatically tracked, even if the security group configuration does not require tracking. These connections must be tracked to ensure symmetric routing.

- Egress-only internet gateways
- Gateway Load Balancers
- Global Accelerator accelerators
- NAT gateways
- Network Firewall firewall endpoints
- Network Load Balancers
- AWS PrivateLink (interface VPC endpoints)
- Transit gateway attachments
- AWS Lambda (Hyperplane elastic network interfaces)

Connections made through the following are automatically tracked, even if the security group configuration does not require tracking.

- Egress-only internet gateways
- Global Accelerator accelerators
- NAT gateways
- Network Firewall firewall endpoints
- Network Load Balancers
- AWS PrivateLink (interface VPC endpoints)
- AWS Lambda (Hyperplane elastic network interfaces)

INTERNET ARCHIVE
Wayback Machine
37 captures
17 Feb 2021 - 5 Nov 2025

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/security-group-connection-tracking.html

aws

Search in this guide

English

Sign In to the Console

AWS > Documentation > Amazon EC2 > User Guide for Linux Instances

Untracked connections

Not all flows of traffic are tracked. If a security group rule permits TCP or UDP flows for all traffic (0.0.0.0/0 or :::/0) and there is a corresponding rule in the other direction that permits all response traffic (0.0.0.0/0 or :::/0) for all ports (0-65535), then that flow of traffic is not tracked. The response traffic is therefore allowed to flow based on the inbound or outbound rule that permits the response traffic, and not on tracking information.

An untracked flow of traffic is immediately interrupted if the rule that enables the flow is removed or modified. For example, if you have an open (0.0.0.0/0) outbound rule, and you remove a rule that allows all (0.0.0.0/0) inbound SSH (TCP port 22) traffic to the instance (or modify it such that the connection would no longer be permitted), your existing SSH connections to the instance are immediately dropped. The connection was not previously being tracked, so the change will break the connection. On the other hand, if you have a narrower inbound rule that initially allows the SSH connection (meaning that the connection was tracked), but change that rule to no longer allow new connections from the address of the current SSH client, the existing connection will not be broken by changing the rule.

Example

In the following example, the security group has specific inbound rules for TCP and ICMP traffic, and outbound rules that allow all outbound IPv4 and IPv6 traffic.

Inbound rules		
Protocol type	Port number	Source IP
TCP	22 (SSH)	203.0.113.1/32
TCP	80 (HTTP)	0.0.0.0/0
TCP	80 (HTTP)	:::/0
ICMP	All	0.0.0.0/0

Outbound rules		
Protocol type	Port number	Destination IP
All	All	0.0.0.0/0
All	All	:::/0

- TCP traffic on port 22 (SSH) to and from the instance is tracked, because the inbound rule allows traffic from 203.0.113.1/32 only, and not all IP addresses (0.0.0.0/0).
- TCP traffic on port 80 (HTTP) to and from the instance is not tracked, because both the inbound and outbound rules allow all traffic (0.0.0.0/0 or :::/0).
- ICMP traffic is always tracked, regardless of rules.
- If you remove the outbound rule from the security group, all traffic to and from the instance is tracked, including traffic on port 80 (HTTP).

Throttling

Amazon EC2 defines the maximum number of connections that can be tracked per instance. After the maximum is reached, any packets that are sent or received are dropped because a new connection cannot be established. When this happens, applications that send and receive packets cannot communicate properly.

To determine whether packets were dropped because the network traffic to your instance exceeded the maximum number of connections that can be tracked, use the `conntrack_olowance_exceeded` network performance metric. For more information, see [Monitor network performance for your EC2 instance](#).

Connections made through the following are automatically tracked, even if the security group configuration does not require tracking:

- Gateway Load Balancers
- Global Accelerator accelerators
- NAT gateways
- Network Firewall firewall endpoints
- Network Load Balancers
- Transit gateway attachments

With Elastic Load Balancing, if you exceed the maximum number of connections that can be tracked per instance, we recommend that you scale either the number of instances registered with the load balancer or the size of the instances registered with the load balancer.

Countermeasures

Connection Tracking

Disable connection tracking

- Have a rule in the reverse direction allowing traffic from all IP addresses¹:

Ingress Rules		Egress Rules	
Local Port	Source IP	Destination IP	Remote Port
80	0.0.0.0/0	0.0.0.0/0	*

- Network functions, imagine NAT instance

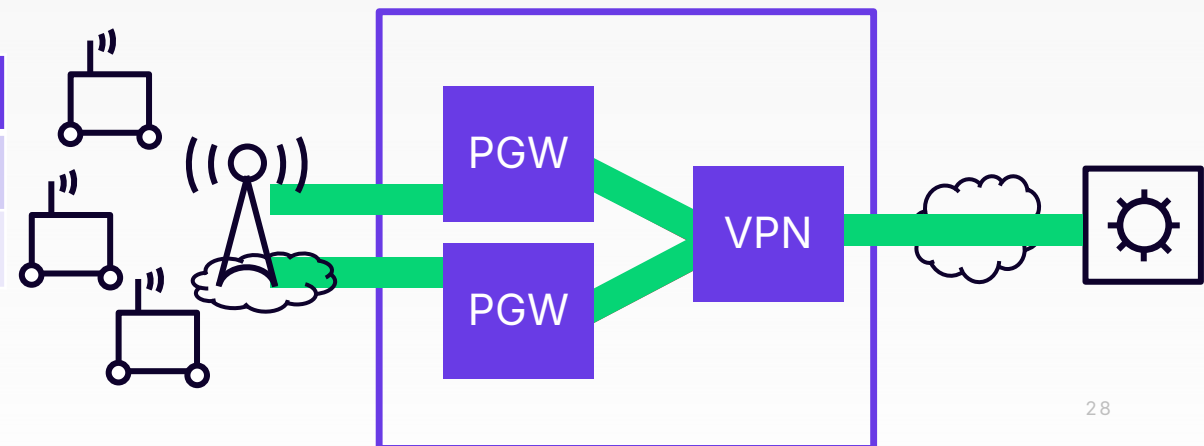
Ingress Rules		Egress Rules	
Local Port	Source IP	Destination IP	Remote Port
*	0.0.0.0/0	0.0.0.0/0	*

- Beware of the security implications!**



Live with it

- Change EC2 instance
 - Scale up instance size, as it doubles 💰
 - Change instance type (cf. table before)
- Lower idle timers of network interface
- Lower cardinality (easier said than done)

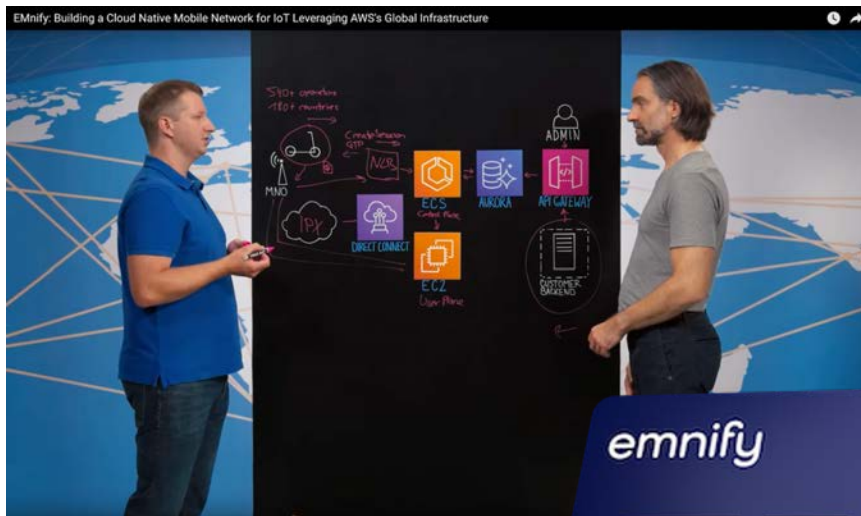


Summary

- The cloud has limits to protect from noisy neighbors
 - Often not clear, what the limits are
 - AWS documentation got a lot better (we should read it more often!)
- Monitoring capabilities are helpful
 - We can't rescue every packet
 - Sometimes still a mystery
- Other clouds?

Resources

- [This is My Architecture: emnify](#)



- [Evaluation SIM Cards](#)



- [AWS re:Invent 2024 - EC2 Nitro networking under the hood \(NET402\)](#)

- [EC2 User Guide – Instance network bandwidth](#)

- [AWS Blogs - Amazon EC2 instance-level network performance metrics uncover new insights](#)

- [AWS re:post - Why does my Amazon EC2 instance exceed its network limits when average utilization is low?](#)