Case study:

Radiator 10 performance with EAP-TLS

Table of contents

Executive summary

This case study describes the performance tests and results for Radiator Software's Radiator 10 RADIUS server. The tests have been conducted by Radiator Software, on standard-sized cloud processing resources with out-of-the-box Radiator 10 software.

The testing was done with EAP-TLSv1.3 authentication with two deployments, one for RADIUS/UDP and one for RADIUS/TLS (RadSec). Both deployments were done in a Google Compute Engine virtual machine, with one 4-core 8 vCPU Radiator 10 instance. RADIUS traffic was generated using a shell script that executed eapol_test tool repeatedly, using the same certificate for all requests.

The tests had two parts, identification of peak performance level, and load test at identified peak performance level for an extended period of time. This test structure proves that the identified peak performance level is sustainable with sustained traffic.

The tests concluded that on the test setup, Radiator 10 could process over 4200 RADIUS EAP-TLSv1.3 requests per second. With parallel RadSec connections from four proxy instances, Radiator processed over 9900 EAP-TLSv1.3 authentications per second. With an average EAP-TLS request requiring 8.4 total RADIUS packet exchanges, this means that Radiator 10 exchanged over 83 000 RADIUS packets per second over the 3 500 000 EAP-TLS authentication test set.

Product	Radiator 10 Radiator Core and Radiator Policy Server	Competitor 1	Competitor 2
Deployment	Google Cloud Compute Engine c2d-highcpu-8	On-site server rack hardware	On-site server rack hardware
Processor	AMD EPYC 7B13	AMD EPYC 9124	Intel 4316 2.3GHz
Cores	4	16	20
TPS (EAP-TLS)	RADIUS/UDP 4255 RadSec 9943	628	200

Based on the results and comparison with available data on RADIUS EAP-TLS authentication performance, it is safe to say that Radiator 10 sets a new industry standard for RADIUS server performance.

Introduction

With people's ever-increasing online activity, communication service providers are faced with increasingly growing performance requirements for their networks. And while computing power grows as well, not all software can utilise the resources and scale to meet these increasing performance requirements.

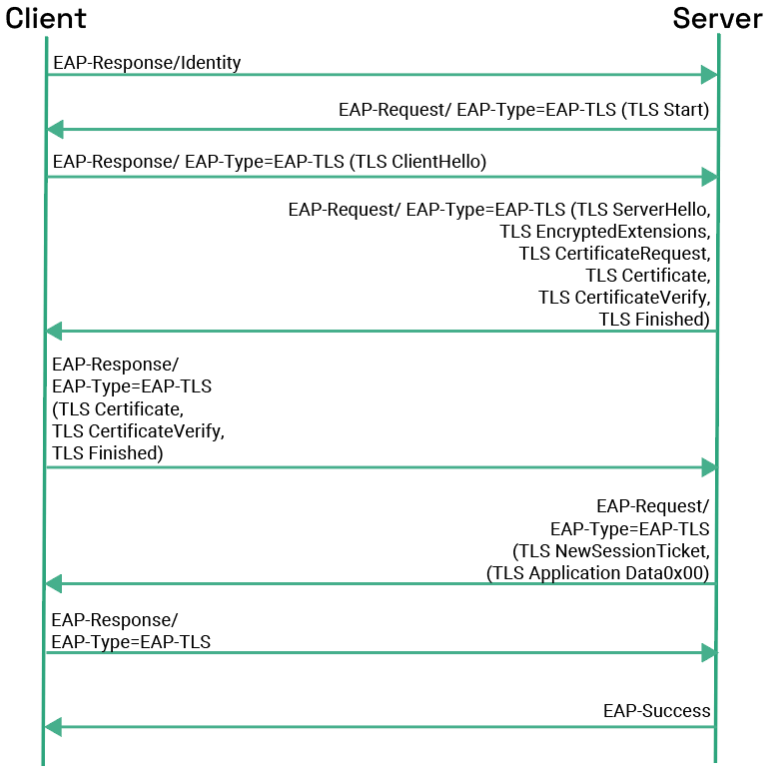
We've closely monitored the feedback from our existing and prospective customers, and designed and built a new policy engine, Radiator 10, from ground up to handle the highest performance requirements, setting the bar for what the performance of a modern RADIUS server should look like.
This case study examines how our product excels in EAP-TLS authentication, demonstrating its ability to process an industry-leading number of authentication transactions per second. In this case study, we showcase how our solution not only enhances security but also delivers unparalleled authentication speed, ensuring smooth and secure access for thousands of users simultaneously.

In the following sections, we'll explain the used authentication method with call flow and request contents, used hardware, deployed setup and testing model, test results and lastly conclusions

Authentication method, call flow

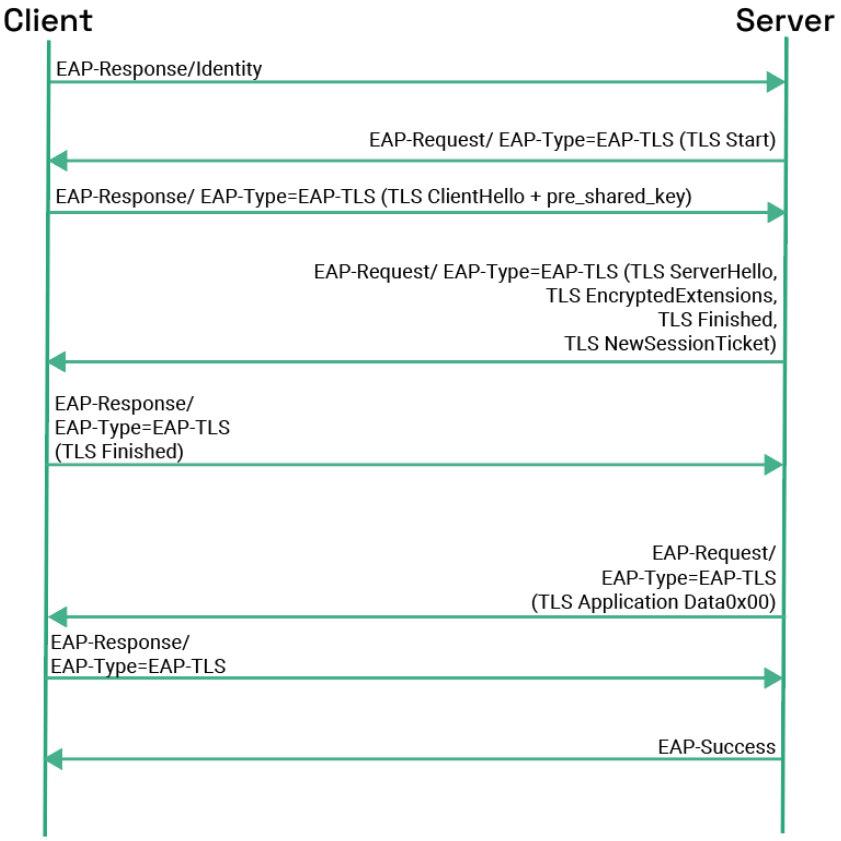
EAP-TLSv1.3 flow

The basis for this case study is EAP-TLSv1.3 authentication. There are two reasons for this: First, we chose a certificate-based method as we wanted to illustrate specifically Radiator 10's performance without a backend like LDAP or SQL, as database performance is a subject of its own. Second, we chose an EAP-based method as it defines one authentication transaction nicely within the EAP context.



1. EAP-TLS authentication begins when the authenticator, such as a wireless access point or switch, sends an EAP Request Identity message to the EAP-TLS client, which responds with its identity. The authenticator and this initial step are not shown in the diagram.
2. A full TLS handshake then proceeds as shown in the diagram, with both client and server using certificates to authenticate each other. RADIUS or RadSec is used to transfer EAP messages to the EAP server.
3. Upon successful authentication, encryption keys are derived from the TLS session. The encryption keys are sent to the authenticator with the EAP-Success message.

To further optimize the authentication process, TLS Session Resumption allows previously authenticated supplicants to reconnect without performing a full TLS handshake. Instead of exchanging certificates again, the server and client reuse a previous full handshake, significantly reducing authentication time and computational overhead.



This is achieved through pre-shared keys (PSKs) that are delivered with NewSessionTicket messages, enabling rapid reauthentication while maintaining security. In large-scale deployments, such as enterprise Wi-Fi networks, TLS Session Resumption enhances performance by minimizing handshake latency and reducing the load on Wi-Fi access points and controllers, and RADIUS servers, ensuring seamless and efficient user experiences.

Definition of TPS

When the term Transactions Per Second (TPS for short) is used in this case study, it refers to one EAP-TLS authentication. In this case study, for every one full TLS handshake, there are nine TLS Session Resumptions. Due to packet fragmentation, one full TLS handshake has twelve RADIUS packets (six requests from Client to Server, six responses from Server to Client), and one Session Resumption has eight RADIUS packets (four requests from Client to Server, four responses from Server to Client). This means that ten EAP-TLS authentications have a total of 84 messages, 42 requests from Client to Server, 42 responses from Server to Client, which means that for the purposes of this case study, an EAP-TLS authentication over the duration of the test has an average of 4.2 RADIUS packets sent to and from Radiator.

In the results and conclusion section of this case study we showcase the performance results in terms of both EAP-TLS authentications per second (transactions per second, TPS), and RADIUS packets sent and received per second. This is to avoid confusion when

comparing with performance tests where results are displayed as RADIUS packets rather than number of authentications.

Resources

The focus of these performance tests is Radiator 10's performance as authenticating RADIUS server, and so all test components are sized with enough capacity for no bottlenecks until Radiator 10's peak sustained performance is reached.

The performance tests for this Case Study were conducted using Google Cloud Compute Engine virtual machines. The authenticating Radiator 10 RADIUS server was run on instance type c2d-highcpu-8 with following specifications:

- 8 vCPUs (4 cores)
- Memory: 16 GB
- CPU:
  AMD EPYC Milan 3rd Generation
  Processor SKU: AMD EPYC™ 7B13
  Base frequency (GHz): 2.45
  Effective frequency (GHz): 2.8
  Max boost frequency (GHz): 3.5
- Disk:
  SSD persistent disk. IOPS varies based on disk size.
  At 500 GB, write IOPS per instance is 15 000
- Connection between instances:
  10 Gbit/s connection with ~0.6 ms ping latency

- Cloud Region: europe-west3-a
- OS Boot image: ubuntu-2404-noble-amd64-v20250214

Proxy servers were run on their own Google Compute Engine c2d-highcpu-4 instances.

Clients were run on their own Google Compute Engine n2d-highcpu-8 instances.

Test structure and deployed setups

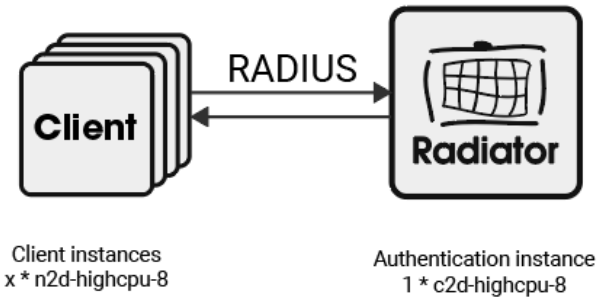
The purpose of this Case study is to give readers a rough idea of how Radiator 10 products can perform out-of-the-box. The performance tests were conducted on Google Cloud Computing machines. Each test had two parts: Performance level identification and load testing with sustained traffic.

In the peak performance identification phase, traffic towards Radiator was generated using separate servers which were programmed to send EAP-TLS requests continuously to Radiator, as fast as the machine's performance allowed. The traffic was increased by increasing the number of servers generating traffic, until first dropped requests were observed.

Once the highest performance level was identified, a stable load at peak sustained capacity was run consecutively for at least five minutes to ensure the identified performance point was sustainable and not just a theoretical one-second peak value.

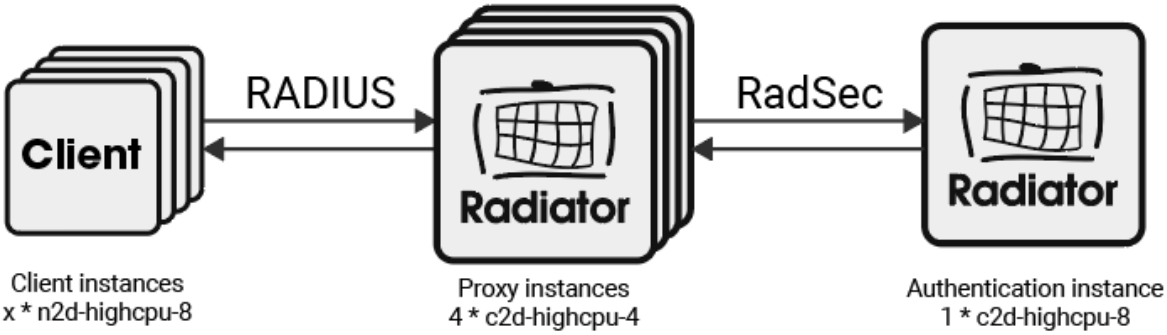
The tests were conducted with two scenarios:

RADIUS/UDP test setup



EAP-TLS with RADIUS over UDP: RADIUS requests were sent from Clients to the authenticating RADIUS Server running Radiator 10.

RADIUS/TLS RadSec test setup



EAP-TLS with RADIUS over UDP and RadSec tunnel: RADIUS requests were sent from Clients to 4 Radiator 10 instances acting as RADIUS-RadSec Proxies. The Proxy instances forwarded RADIUS traffic to the authenticating RADIUS Server over secure RadSec tunnels. Each proxy instance formed its own RadSec connection(s) over TCP to the authenticating Server running Radiator 10.

As shown in the deployment illustrations, all Clients were run on their own Google Compute Engine n2d-highcpu-8 instances, the proxy servers were run on their own Google Compute Engine c2d-highcpu-4 instances and the authenticating Radiator 10 server was run on a Google Compute Engine c2d-highcpu-8 instance (see above for detailed specification).

The Proxy instances were designed and sized with adequate resources, meaning they were never the bottleneck in the setup. In both test scenarios, Client instances were

programmed to simulate EAP clients using a shell script executing eapol_test tool repeatedly. During testing, all clients sent EAP-TLS requests to Radiator instances using the same certificate, which resulted in a successful authentication. The certificates used were self-signed prime256v1 (EC) type certificates with no intermediate CA. For both test scenarios, the amount of traffic was increased by increasing the number of client instances, until a point was found where Radiator could not handle all incoming requests.

The test configuration also included authorisation with two RADIUS reply-attributes added, and extensive JSON logging for authentication, to ensure these test results reflect real-world scenarios and are not purely theoretical.

Results

RADIUS/UDP

In the first test phase, RADIUS/UDP from Client instances, the identified peak sustained performance was at three client instances, as with four client instances Radiator failed to process all requests.

In the load testing phase, it was identified that three client instances produced a stable load of 4255 TPS, which Radiator handled without a single error over extended periods of time.
In a second data set, a similar amount of traffic was sent to Radiator but from six slower clients servers (equalling to roughly same computing power as three more powerful servers) instead, processing speed remained the same. This means that for the RADIUS/UDP deployment, the number of traffic sources did not have an observed effect on performance.

Without load-balancing this is roughly the limit of how much traffic one Radiator 10 instance can handle, as the bottleneck of one listening port can not be extended further. The server's processing capacity was not reached and the same instance could have been used for other processing, such as RADIUS accounting. However, as showcased in the next test set, with parallel connections from RadSec, the authentication performance can be increased vastly.

RADIUS/TLS (RadSec) and TLS1.3

In the second test phase, RADIUS/UDP traffic from Client instances was sent through four Radiator 10 Proxy instances with RadSec connections to Radiator 10 authenticating RADIUS Server. TLS1.3 was using self-signed prime256v1 (EC) type certificates with no intermediate CA.

The identified peak sustained performance was at seven client instances, as with eight client instances Radiator was not able to process all requests successfully.

In the load testing phase, the sustainable load was tested with sets of 3 500 000 EAP-TLS authentications (seven client instances, 500 000 authentications each). Radiator 10 processed the 3 500 000 EAP-TLS authentications in 352 seconds, resulting in 9943 TPS.

This means that with one Radiator 10 instance, nearly 10 000 EAP-TLS connections were successfully authenticated per second, over an extended period of time. As each EAP-TLS connection consisted of an average of 4.2 RADIUS messages sent and received, this means that one Radiator 10 instance received over 41 000 and sent over 41 000 RADIUS packets per second for a total of over 82 000 RADIUS packets of communication passing through in a second.

As the server was configured with 8 vCPUs, the results equaled ~1243 TPS per vCPU and 2486 TPS per core, or 5220 sent RADIUS packets per second. The results are illustrated in the table below.

Setup	EAP-TLS authentications per second, TOTAL	EAP-TLS authentications per second, per vCPU	RADIUS packets sent per second, TOTAL	RADIUS packets sent per second, per vCPU
RADIUS/UDP 3 Clients	4255	532	17900	2234
RADIUS/TLS 7 Clients 4 RADIUS/RadSec proxy instances	9943	1243	41760	5220

But we think a better illustration would be a comparison between Radiator 10 and other RADIUS server products (based on their latest available material). The comparison is done between EAP-TLS deployments where certificates are stored locally. In this comparison, highest values are used.

Product	Radiator 10 Radiator Core and Radiator Policy Server	Competitor 1	Competitor 2
Deployment	Google Cloud Compute Engine c2d-highcpu-8	On-site server rack hardware	On-site server rack hardware
Processor	AMD EPYC 7B13	AMD EPYC 9124	Intel 4316 2.3GHz
Cores	4	16	20
TPS (EAP-TLS)	RADIUS/UDP 4255 RadSec 9943	628	200

Considerations

This case study was done to give readers a rough idea of Radiator 10's performance out-of-the-box. The biggest point to consider is the peak performance identification model. The increase in traffic was done by increasing client instances which generated traffic. In the RADIUS/UDP test, the peak sustained performance was identified to be between 3 and 4 client instances. This is a 33% increase in traffic, and the peak sustained performance is somewhere between 3 and 4, and not directly at 3. The same goes for RADIUS/TLS test, though not to the same extent. The difference between seven and eight client instances is ~14% in traffic, and having seven faster and one slower client instances would have pushed the results over 10 000 TPS.

Other considerations:

1. For use cases with authentication, authorisation or accounting data backends (SQL, LDAP, REST, ...), backend performance can be a limiting factor, so RADIUS server can not be guaranteed to reach its peak performance in real-life scenarios. Each use-case needs to be tested separately to confirm precise end-to-end performance.
2. On the other hand, EAP-TLS is not the lightest authentication protocol, as it requires multiple challenge/response steps between the Client and the Server during the TLS handshake.
3. The configuration used in these tests included authorisation factors and produced extensive logging. Theoretical numbers with minimal logging and no authorisation factors could exceed numbers produced in this testing scenario.
4. The performance of cloud computing resources varied depending on the time of day. During this test the perceived performance was ~5% slower during busiest hours. It might have been possible to achieve a ~5% performance gain on the quietest time of the day as well. This is the nature of cloud computing when using virtual servers on shared hardware. Operating the servers on dedicated hardware mitigates this variation.
5. Performance may vary depending on the cloud computing environment or physical hardware used. When running on physical hardware, performance stays consistent, while on virtual machines other provisioned usage may affect performance if resources are not reserved for the VM instance.
6. Radiator 10 products (Radiator Policy Server and Radiator Core) are recently released and under active development, so results are subject to change as the products evolve.

Conclusions

The key takeaway from this case study is in the comparison chart showcased at the bottom of the results section.

Product	Radiator 10 Radiator Core and Radiator Policy Server	Competitor 1	Competitor 2
Deployment	Google Cloud Compute Engine c2d-highcpu-8	On-site server rack hardware	On-site server rack hardware
Processor	AMD EPYC 7B13	AMD EPYC 9124	Intel 4316 2.3GHz
Cores	4	16	20
TPS (EAP-TLS)	RADIUS/UDP 4255 RadSec 9943	628	200

When comparing available performance test data from some of the largest RADIUS server vendors' products, during this performance test Radiator 10's performance was on another level compared to rival products. Radiator's performance reached over 15 times Competitor 1's and nearly 50 times Competitor 2's performance figures in EAP-TLS authentication from locally stored credentials.

This is a remarkable discovery for companies that struggle to scale up their systems due to RADIUS server software not being able to keep up. While this is a good start, it should also be noted that this is a straight-forward test with an out-of-the-box product, showcasing how a one instance without a complicated architecture can process the request amount of a handsomely sized internet service provider.

Want to know more?

If you want to know more about the Radiator 10 performance testing process, have any questions or want to discuss if Radiator 10 products, Radiator Policy Server and Radiator Core, could help you scale your deployment, please do not hesitate to contact **sales@radiatorsoftware.com**