VALTIX

TLS RAMIFICATIONS FOR LEGACY NGFW VIRTUAL APPLIANCES

E-Book

Speed of the Cloud

The cloud is naturally evolving as the new business environment. Wider adoption is inevitable—and accelerating. Yet on-premises security architectures can no longer adequately support the dynamic, elastic nature of today's (and tomorrow's) cloud. Only a native security solution can unify single and multi-cloud security for greater visibility, scalability, and performance. It's time for cloud security to move at the "speed of cloud."

VALTIX

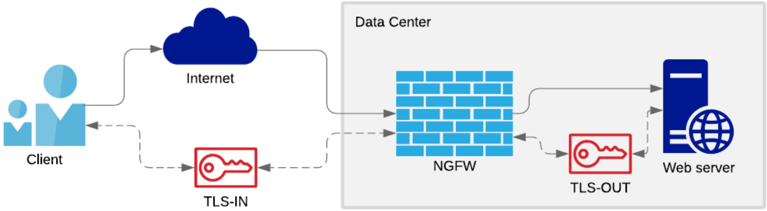
TLS Ramifications for Legacy NGFW Virtual Appliances

For Outside-In DC Traffic In Public Clouds

This e-Book empowers security architects with deployment consideration and practices to securely support TLS 1.2 and 1.3 simultaneously in Public Clouds, while mitigating the need for frontend and backend Load Balancers in TLS 1.2 deployments with PFS (Perfect Forward Secrecy) enabled & 1.3 environments.

Securing Communications with TLS

Consider a typical Next-Gen Firewall (NGFW) deployment for Outside-In traffic, where external clients are traversing the Internet to access a hosted service in a data center. Similar deployment exists to secure inter-VPC/VNet traffic traversing in and out of enterprise's VPCs/VNets.



Transport Layer Security (TLS) is the encryption protocol used to protect communications between the client and the web server going over the Internet. Before a client can talk to a server, they have to go through a TLS handshake to negotiate a shared secret which will be then used to form the basis for the symmetric encryption key used to secure the data transfers. Clients encrypt data with symmetric encryption keys and send it to destination servers which then decrypt with symmetric encryption keys to transform data into clear text for processing.

NGFW(s), checking for malware for incoming traffic from the Internet (client) or for applying any policy on Layer7 parameters, must decrypt the packets to do deep packet inspection (DPI). As a result, the NGFW must have a way to derive the client symmetric encryption key by snooping in on the TLS handshake process, which is where the symmetric keys are derived by the client and the server.

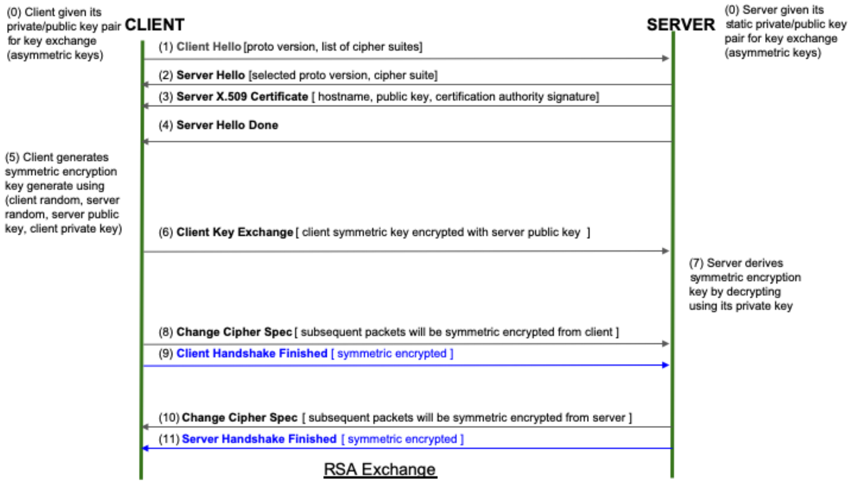
> “Keep fighting the MitM / decryption battles and you will win some and lose some, but will eventually lose the war. Will it be in 2021 or 2030? No idea when, but it will happen.”
>
> – Anton Chuvakin , Gartner Blog Network

TLS 1.2

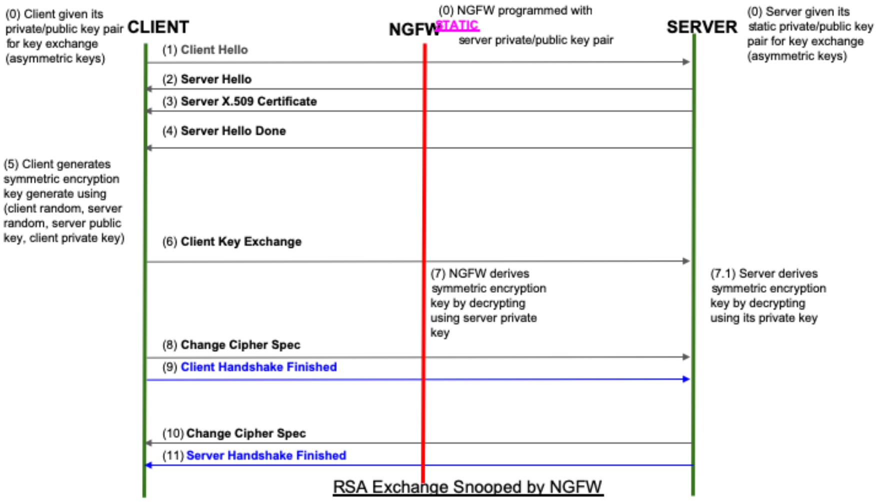
A TLS connection begins with the handshake - where the client and server verify their identities and negotiate which cipher suite and key algorithm they will use to secure their communications. Most today use either RSA or Diffie-Hellman key exchange.

TLS 1.2 supports static key exchange for both RSA and Diffie-Hellman. This means the private/public key pair generated on the server side is static (fixed forever). This means that the server key-pair can then be programmed to NGFW out-of-band for it to independently derive the symmetric encryption keys exactly like the server by NGFW snooping in on the handshake as shown in the following diagrams.

RSA Exchange

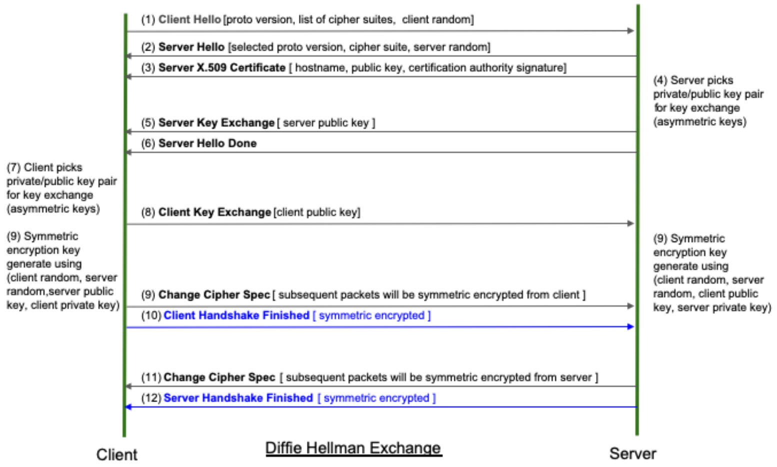

(0) Client given its private/public key pair for key exchange (asymmetric keys)
CLIENT
SERVER
(0) Server given its static private/public key pair for key exchange (asymmetric keys)
(1) Client Hello [proto version, list of cipher suites]
(2) Server Hello [selected proto version, cipher suite]
(3) Server X.509 Certificate [hostname, public key, certification authority signature]
(4) Server Hello Done
(5) Client generates symmetric encryption key generate using (client random, server random, server public key, client private key)
(6) Client Key Exchange [client symmetric key encrypted with server public key]
(7) Server derives symmetric encryption key by decrypting using its private key
(8) Change Cipher Spec [subsequent packets will be symmetric encrypted from client]
(9) Client Handshake Finished [symmetric encrypted]
(10) Change Cipher Spec [subsequent packets will be symmetric encrypted from server]
(11) Server Handshake Finished [symmetric encrypted]

RSA Exchange

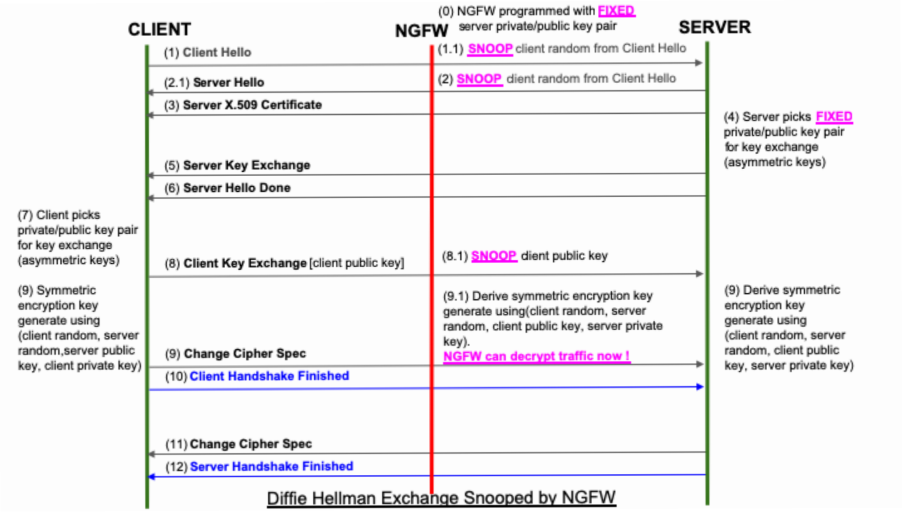

(0) Client given its private/public key pair for key exchange (asymmetric keys)
CLIENT
NGFW
(0) NGFW programmed with STATIC server private/public key pair
SERVER
(0) Server given its static private/public key pair for key exchange (asymmetric keys)
(1) Client Hello
(2) Server Hello
(3) Server X.509 Certificate
(4) Server Hello Done
(5) Client generates symmetric encryption key generate using (client random, server random, server public key, client private key)
(6) Client Key Exchange
(7) NGFW derives symmetric encryption key by decrypting using server private key
(7.1) Server derives symmetric encryption key by decrypting using its private key
(8) Change Cipher Spec
(9) Client Handshake Finished
(10) Change Cipher Spec
(11) Server Handshake Finished

RSA Exchange Snooped by NGFW

Diffie Hellman Exchange


(1) Client Hello [proto version, list of cipher suites, client random]
(2) Server Hello [selected proto version, cipher suite, server random]
(3) Server X.509 Certificate [hostname, public key, certification authority signature]
(4) Server picks private/public key pair for key exchange (asymmetric keys)
(5) Server Key Exchange [server public key]
(6) Server Hello Done
(7) Client picks private/public key pair for key exchange (asymmetric keys)
(8) Client Key Exchange [client public key]
(9) Symmetric encryption key generate using (client random, server random,server public key, client private key)
(9) Change Cipher Spec [subsequent packets will be symmetric encrypted from client]
(9) Symmetric encryption key generate using (client random, server random, client public key, server private key)
(10) Client Handshake Finished [symmetric encrypted]
(11) Change Cipher Spec [subsequent packets will be symmetric encrypted from server]
(12) Server Handshake Finished [symmetric encrypted]
Client
Server

Diffie Hellman Exchange


CLIENT
NGFW
(0) NGFW programmed with FIXED server private/public key pair
SERVER
(1) Client Hello
(1.1) SNOOP client random from Client Hello
(2.1) Server Hello
(2) SNOOP dient random from Client Hello
(3) Server X.509 Certificate
(4) Server picks FIXED private/public key pair for key exchange (asymmetric keys)
(5) Server Key Exchange
(6) Server Hello Done
(7) Client picks private/public key pair for key exchange (asymmetric keys)
(8) Client Key Exchange [client public key]
(8.1) SNOOP dient public key
(9) Symmetric encryption key generate using (client random, server random,server public key, client private key)
(9.1) Derive symmetric encryption key generate using(client random, server random, client public key, server private key).
NGFW can decrypt traffic now !
(9) Derive symmetric encryption key generate using (client random, server random, client public key, server private key)
(9) Change Cipher Spec
(10) Client Handshake Finished
(11) Change Cipher Spec
(12) Server Handshake Finished

Diffie Hellman Exchange Snooped by NGFW

TLS 1.3/TLS 1.2 with Perfect Forward Secrecy (PFS) Throws A Spanner In The Works

Perfect Forward Secrecy (PFS) is quickly becoming a mandatory security measure from enterprises.The intent of PFS is to prohibit hackers, who have previously stored older encrypted conversations between a client and server, from decrypting such communications utilizing older private key.

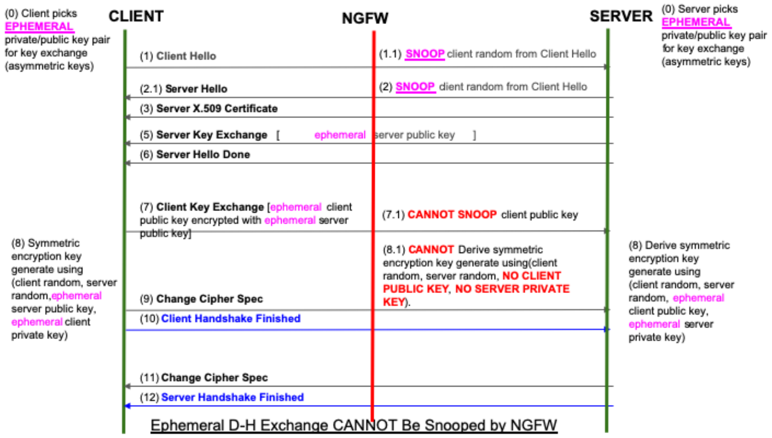
TLS 1.3 deprecated static RSA and static Diffie-Hellman exchanges for Perfect Forward Secrecy (PFS) reasons.

Ephemeral becomes the new mantra for TLS 1.3. What this means is that instead of using fixed/static private/public key pair, asymmetric private/

public key pair is being generated on a per TLS session basis. This key-pair is ephemeral in the sense that it becomes obsolete once the session is torn down. DH(E) and ECDH(E) have become the standard key exchanges for TLS 1.3 (E for Ephemeral).

Even if enterprises are slow to adopt TLS 1.3, a lot of enterprises are now mandating support of TLS 1.2 protocol with ciphersuites that only support PFS.

From the diagram below, it becomes very clear why NGFW operating in forwarding mode cannot decrypt traffic with PFS friendly key exchanges.


(0) Client picks EPHEMERAL private/public key pair for key exchange (asymmetric keys)
CLIENT
NGFW
SERVER
(0) Server picks EPHEMERAL private/public key pair for key exchange (asymmetric keys)
(1) Client Hello
(1.1) SNOOP client random from Client Hello
(2.1) Server Hello
(2) SNOOP dient random from Client Hello
(3) Server X.509 Certificate
(5) Server Key Exchange [ephemeral server public key]
(6) Server Hello Done
(7) Client Key Exchange [ephemeral client public key encrypted with ephemeral server public key]
(7.1) CANNOT SNOOP client public key
(8) Symmetric encryption key generate using (client random, server random, ephemeral server public key, ephemeral client private key)
(8.1) CANNOT Derive symmetric encryption key generate using(client random, server random, NO CLIENT PUBLIC KEY, NO SERVER PRIVATE KEY).
(8) Derive symmetric encryption key generate using (client random, server random, ephemeral client public key, ephemeral server private key)
(9) Change Cipher Spec
(10) Client Handshake Finished
(11) Change Cipher Spec
(12) Server Handshake Finished
Ephemeral D-H Exchange CANNOT Be Snooped by NGFW

> “32% of US companies and 16% of European companies received failing grades for their SSL/TLS implementations. Only around 15% of the companies had an SSL/TLS configuration that was compliant with current PCI DSS requirements.“
>
> - High-Tech Bridge 2018

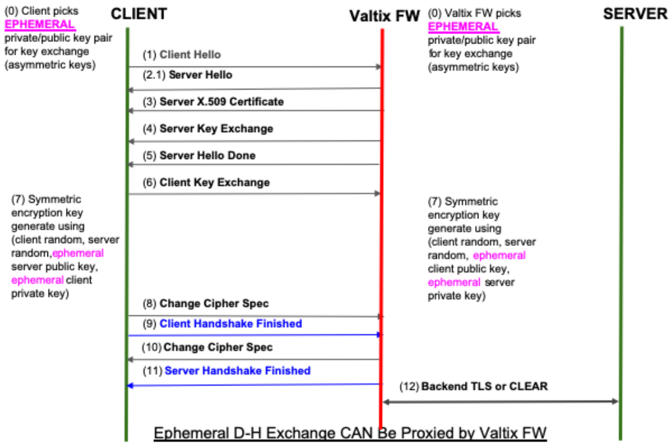
Solution for PFS

TLS 1.3 lets you run in “downgraded” TLS 1.2 with non-PFS cipher suites. This means PFS is broken. This design was put in place so that existing middleboxes (ref: NGFWs) can continue to work in conjunction with no PFS, which should be a major security concern for any enterprise.

The optimal solution is to use a TLS proxy with one of the following:

- TLS 1.2 PFS cipher suites (TLS-ECDHE-* or TLS-DHE-*) , or
- TLS 1.3 where the ephemeral key exchanges are the standard.

The Valtix Cloud Firewall natively supports both of these options with accelerated elliptic curve multiplication as well as the RSA sign/verify operations without performance degradation. More will be shared soon about how this is accomplished.



Ephemeral D-H Exchange CAN Be Proxied by Valtix FW

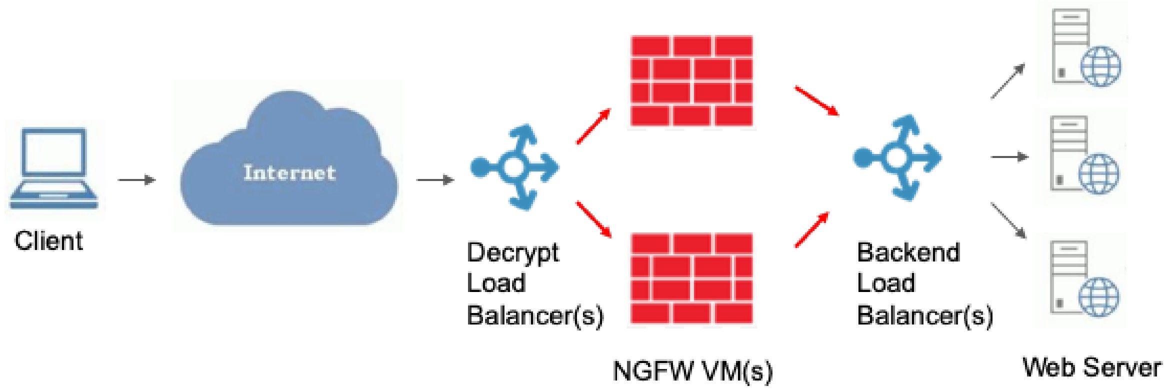
> “In contrast to TLS 1.2, TLS 1.3 provides additional privacy for data exchanges by encrypting more of the negotiation handshake to protect it from eavesdroppers. This enhancement helps protect the identities of the participants and impede traffic analysis. TLS 1.3 also enables forward secrecy by default which means that the compromise of long term secrets used in the protocol does not allow the decryption of data communicated while those long term secrets were in use. As a result, current communications will remain secure even if future communications are compromised.”
>
> *– IETF | Internet Engineering Task Force*

TLS Everywhere

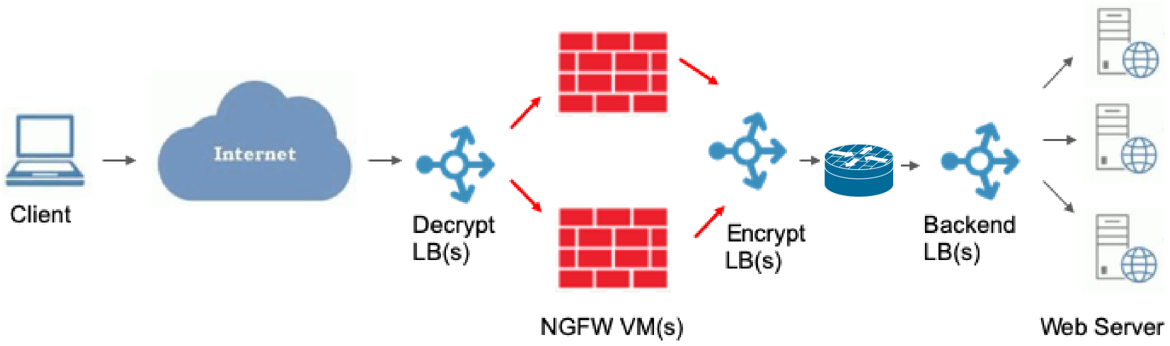
Load Balancer (LB) “Sandwich”

Since legacy firewalls have challenges scaling to cloud scale, the recommended public cloud security practice is to prescribe load balancer sandwich. This increases operational & support costs as well as the complexity of environments.

TLS termination (TLS handshake overhead) comes with a heavy processing overhead. TLS symmetric encryption/decryption
i
l
l
a



A dedicated load balancer for decryption is needed for TLS termination. However, with the Valtix Cloud Firewall this is not needed because this is performed natively.



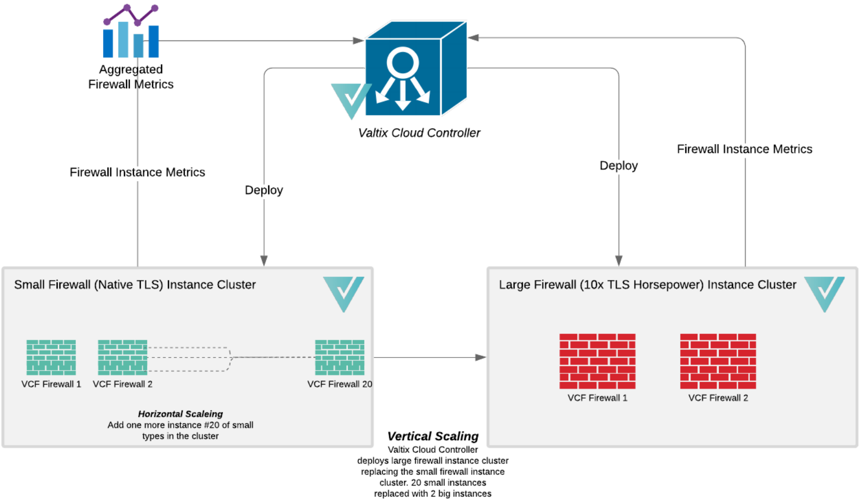
For enterprises with strict TLS everywhere mandates, the paths from the Decrypt load balancer to the virtual NGFW instance and the Encrypt load balancer are in violation creating TLS air gaps. These paths are noted red in the above two figures.

Summary

The Valtix Cloud Firewall offers native decryption/re-encryption of TLS with PFS and full proxy capabilities that greatly simplify your deployment architectures and most importantly ensuring the best security practices with no TLS air gaps helping the TLS everywhere mandate of enterprises.

Valtix Security Platform eliminates the need for “dedicated” and redundant network components required for TLS decryption/encryption via native TLS supporting both:

- Vertical scaling from basic to advanced parallel instances accelerating TLS, and
- Horizontal scaling by adding more instances in a single cluster to address increased TLS processing requirements.


Aggregated Firewall Metrics
Valtix Cloud Controller
Firewall Instance Metrics
Deploy
Deploy
Firewall Instance Metrics
Small Firewall (Native TLS) Instance Cluster
VCF Firewall 1
VCF Firewall 2
VCF Firewall 20
Horizontal Scaleing
Add one more instance #20 of small types in the cluster
Large Firewall (10x TLS Horsepower) Instance Cluster
VCF Firewall 1
VCF Firewall 2
Vertical Scaling
Valtix Cloud Controller deploys large firewall instance cluster replacing the small firewall instance cluster. 20 small instances replaced with 2 big instances

This gives security, DevOps and IT teams a complete and secure solution for advanced inspection, monitoring and compliance in their public cloud workloads when deploying in Public Cloud.

> “Calls on protocol designers, developers, and operators to make encryption the norm for Internet traffic...We recommend that encryption be deployed throughout the protocol stack since there is not a single place within the stack where all kinds of communication can be protected. We strongly encourage developers to include encryption in their implementations, and to make them encrypted by default. We similarly encourage network and service operators to deploy encryption where it is not yet deployed, and we urge firewall policy administrators to permit encrypted traffic.”
>
> – *The Internet Architecture Board (IAB)*

About Valtix

Valtix is the industry's first, cloud-native network security platform for enterprises. Comprised of Valtix Cloud Controller and Valtix Cloud Firewall, the solution revolutionizes cloud network security with innovations that make visibility and enforcement automatic at the pace of the apps they protect. The centralized multi-cloud controller supports deployments for AWS and Azure (and GCP later this year). The firewall is architected with built-in auto scale, app-aware security policy and a single-pass pipeline for TLS, advanced FW, IPS, advanced WAF and more, which operates on a variety of cloud instance types from basic to the most advanced. For more information, contact us at sales@valtix.com or visit www.valtix.com.

VALTIX

Valtix, Inc.
2901 Tasman Drive, #222 • Santa Clara, CA 95054

650.420.6014 • sales@valtix.com

www.valtix.com