Version 1.0 | Date: August 2025

Author: Rahul Kumar

IPsec Deployment and Validation Guide

🌐 What is IPsec?

IPsec (Internet Protocol Security) is a suite of protocols that provides **secure communication** over IP networks (like the internet or private WANs).

Think of it as a **security wrapper for IP packets**:

It encrypts the data (confidentiality)
It checks that data isn't modified (integrity)
It verifies who sent the data (authentication)
It can prevent replay attacks (anti-replay protection)

🔑 How IPsec Works

IPsec protects traffic by establishing **tunnels** between two devices (firewalls, routers, VPN concentrators).

It has **two main phases**:

1. IKE (Internet Key Exchange)
Negotiates which algorithms to use (encryption, hashing, DH group)
Authenticates peers (pre-shared key, certificates, etc.)
Builds the secure channel for key exchange (Phase 1 SA)

2. IPsec (ESP or AH)
Actually encrypts and protects the data traffic (Phase 2 SA)
Commonly uses **ESP (Encapsulating Security Payload)** for encryption + integrity

🔒 Two IPsec Modes

Transport Mode
Protects only the payload of IP packets
Original IP header remains
Used in end-to-end systems (host-to-host)

Tunnel Mode
Entire original IP packet is encrypted and encapsulated inside a new IP packet
A new IP header is added (between VPN gateways)
Most common in site-to-site VPNs (like SRX to SRX in your lab)

📦 IPsec Packet Flow (simplified)

Original packet:
```
[Src IP][Dst IP][TCP/UDP payload]
```

After ESP encryption (Tunnel mode):
```
[New IP header: SRX-A public → SRX-B public]
[ESP header]
[Encrypted original IP packet]
[ESP trailer + auth]
```

🔎 Why NAT-T is needed?

IPsec ESP (Encapsulating Security Payload) normally uses **protocol number 50** (not TCP/UDP).
Many NAT devices/firewalls don't understand or properly handle protocol 50.
NAT also modifies IP headers (and sometimes ports), which can **break the IPsec integrity checks** (hash mismatch).
➡️ Result: Without NAT-T, **IPsec tunnels fail** if there's a NAT device in the path (common for home broadband, ISPs, cloud environments).

🔑 How NAT-T works

NAT-T encapsulates IPsec ESP packets **inside UDP packets (port 4500)**.
This makes the traffic look like normal **UDP traffic**, which NAT devices can handle easily.
Peers detect NAT in the IKE negotiation (during Phase 1) and **switch to UDP/4500** automatically.

📦 IPsec with NAT-T Packet Flow

Normal IPsec (no NAT):
[Outer IP header][ESP header][Encrypted Payload]
Protocol = 50 (ESP)
With NAT-T (NAT in path):
[Outer IP header][UDP header (port 4500)][ESP header][Encrypted Payload]
Protocol = 17 (UDP)
→ NAT treats it as UDP traffic and doesn't break IPsec integrity.

⚙️ In Juniper SRX (example config)

NAT-T is usually enabled by default, but you can explicitly allow it:
set security ike gateway IKE-GW nat-keepalive
This sends periodic keepalives so NAT devices don't close the UDP/4500 session.

✅ In short

NAT-T = NAT Traversal
Encapsulates ESP (protocol 50) inside **UDP/4500**
Ensures IPsec works through NAT devices/firewalls
Widely used in site-to-site and remote-access VPNs

🔑 IKEv1 vs IKEv2 (used in IPsec VPNs)

Feature	IKEv1 (RFC 2409)	IKEv2 (RFC 4306 / 7296)
Standard Age	Old (1998), considered legacy	New (2005), modern & recommended
Exchange Types	- Phase 1 (Main Mode / Aggressive Mode)- Phase 2 (Quick Mode)	Only **two exchanges** (simpler) - IKE_SA_INIT and IKE_AUTH
Message Count	6 messages for Main Mode + 3 for Quick Mode (total ~9)	Only 4 messages to set up tunnel (faster)
Authentication	Pre-shared keys, certificates	Pre-shared keys, certificates, EAP (better remote user support)
NAT Traversal	Supported, but added later (NAT-T extensions)	Built-in NAT-T support (UDP/4500)
Rekeying	Complex – Phase 1 and Phase 2 must be renegotiated separately	Simple – rekeys can happen without full re-negotiation

Mobility Support	No built-in mobility	MOBIKE (Mobility & Multihoming Extension) supported
Error Handling	Limited diagnostics, hard to debug	Better error reporting, clearer notifications
Multiple Tunnels	Each tunnel requires separate negotiation	Supports multiple Child SAs under one IKE SA (more efficient)
Status Today	Legacy, still in use for compatibility	Recommended for all new deployments

🧠 Easy Way to Remember

IKEv1 = old, verbose, clunky (two phases, multiple modes, lots of messages).
IKEv2 = modern, fast, efficient, with built-in NAT and mobility support.

📌 Real SRX Impact

In your config earlier, you forced:
```
set security ike gateway IKE-GATEWAY version v1-only
```

That means the SRX is using **IKEv1**.
If you switch to:
```
set security ike gateway IKE-GATEWAY version v2-only
```

→ The tunnel will use **IKEv2**, which is recommended unless you need to interop with a legacy peer.

✅ In short:

IKEv1 = older, two-phase, heavier.
IKEv2 = newer, streamlined, secure, NAT/mobility friendly → use whenever possible.

Nice one Rahul 👍 — this is exactly where the difference between **IKEv1** and **IKEv2** becomes super clear:
Let's break down the **message exchanges** step by step.

🔑 Messages in IKEv1

IKEv1 is **two-phase** (Phase 1 = IKE SA, Phase 2 = IPsec SA).

Phase 1 (IKE SA)

Two modes: **Main Mode (6 messages)** or **Aggressive Mode (3 messages)**

Main Mode (6 messages):

```
Initiator → Responder: Proposals (encryption, hash, DH group)
Responder → Initiator: Selected proposal
Initiator → Responder: DH public value, nonce
Responder → Initiator: DH public value, nonce
Initiator → Responder: Identity (encrypted) + authentication
Responder → Initiator: Identity (encrypted) + authentication
```

➡️ Result: Secure IKE SA established.

Aggressive Mode (3 messages):

All of the above crammed into 3 messages (faster, less secure).

Phase 2 (Quick Mode, 3 messages)

```
Initiator → Responder: Proposals for IPsec SA (ESP/AH, keys, lifetimes)
Responder → Initiator: Selected proposal
Initiator ↔ Responder: Key material exchange, SA confirmation
```

➡️ Result: IPsec SA (tunnel) established.

🔢 **Total Messages (Main Mode)** = 6 (Phase 1) + 3 (Phase 2) = **9 messages**

🔑 Messages in IKEv2

IKEv2 is **simplified** — only **4 messages total** for tunnel setup.

IKE_SA_INIT (2 messages)

```
Initiator → Responder: Cryptographic proposals (encryption,
integrity, DH group), nonce
Responder → Initiator: Selected proposal, DH public value,
nonce
```

➡️ Negotiates crypto + does DH exchange.

IKE_AUTH (2 messages)

```
Initiator → Responder: Identity, authentication (PSK or cert),
traffic selectors (local subnets)
Responder → Initiator: Identity, authentication, traffic
selectors
```

➡️ Authenticates both peers + establishes **first IPsec SA**.

🔢 **Total Messages (IKEv2) = 4 messages**

🆚 **IKEv1 vs IKEv2 Message Count**

Phase	IKEv1 (Main Mode)	IKEv2
IKE SA setup	6 msgs	2 msgs
IPsec SA setup	3 msgs	2 msgs (inside IKE_AUTH)
Total	~9 msgs	4 msgs

🧠 **Easy Analogy**

IKEv1 = Like filling two long forms at the bank (Phase 1 form + Phase 2 form).
IKEv2 = One short online form (simpler, faster, fewer messages).

🛠️ Example: BGP over IPsec

In this case:

SRX-A and SRX-B have **public IPs (1.1.1.1, 1.1.1.2)**
They build an **IPsec tunnel (st0.0)** between them
Inside this tunnel, they run **BGP** using tunnel IPs (10.0.0.1 ↔ 10.0.0.2)
So BGP packets are encapsulated into **ESP** and sent securely over the internet

SRX-A (AS 100, public 1.1.1.1, st0: 10.0.1.1)

Interfaces

```
set interfaces ge-0/0/0 unit 0 family inet address 1.1.1.1/24
set interfaces st0 unit 0 family inet address 10.0.1.1/24
set interfaces lo0 unit 0 family inet address 192.168.1.1/32
```

optional example route to advertise

Security Zones & Host-Inbound (device-facing services)

```
set security zones security-zone UNTRUST interfaces ge-0/0/0.0
set security zones security-zone UNTRUST interfaces ge-0/0/0.0
host-inbound-traffic system-services ike
set security zones security-zone UNTRUST interfaces ge-0/0/0.0
host-inbound-traffic system-services ping
set security zones security-zone UNTRUST interfaces st0.0
set security zones security-zone UNTRUST interfaces st0.0
host-inbound-traffic system-services ping
set security zones security-zone UNTRUST interfaces st0.0
host-inbound-traffic protocols bgp
```

(For BGP destined to the SRX itself, host-inbound on st0.0 is the key. No inter-zone policy is needed for BGP-to-box. Add security policies later if you want to pass transit traffic between LANs.)

IKE (Phase 1)

```
set security ike proposal IKE-PROPOSAL authentication-method
pre-shared-keys
set security ike proposal IKE-PROPOSAL dh-group group14
set security ike proposal IKE-PROPOSAL authentication-
algorithm sha-256
set security ike proposal IKE-PROPOSAL encryption-algorithm
aes-256-cbc
set security ike policy IKE-POLICY mode main
set security ike policy IKE-POLICY proposals IKE-PROPOSAL
set security ike policy IKE-POLICY pre-shared-key ascii-text
"Juniper"
set security ike gateway IKE-GATEWAY ike-policy IKE-POLICY
set security ike gateway IKE-GATEWAY address 1.1.1.2
set security ike gateway IKE-GATEWAY external-interface ge-
0/0/0.0
set security ike gateway IKE-GATEWAY version v1-only
set security ike gateway IKE-GATEWAY dead-peer-detection
interval 10
set security ike gateway IKE-GATEWAY dead-peer-detection
threshold 3
```

IPsec (Phase 2)

```
set security ipsec proposal IPSEC-PROPOSAL protocol esp
set security ipsec proposal IPSEC-PROPOSAL authentication-
algorithm hmac-sha-256-128
set security ipsec proposal IPSEC-PROPOSAL encryption-
algorithm aes-256-cbc
set security ipsec policy IPSEC-POLICY proposals IPSEC-
PROPOSAL
set security ipsec vpn IPSEC-VPN bind-interface st0.0
set security ipsec vpn IPSEC-VPN ike gateway IKE-GATEWAY
set security ipsec vpn IPSEC-VPN ike ipsec-policy IPSEC-POLICY
set security ipsec vpn IPSEC-VPN establish-tunnels immediately
set security ipsec vpn IPSEC-VPN perfect-forward-secrecy keys
group14
```

BGP (over st0)

```
set routing-options autonomous-system 100
set protocols bgp group EBGP type external
set protocols bgp group EBGP peer-as 200
set protocols bgp group EBGP neighbor 10.0.1.2
```

SRX-B (AS 200, public 1.1.1.2, st0: 10.0.0.2)

Interfaces

```
set interfaces ge-0/0/1 unit 0 family inet address 1.1.1.2/24
set interfaces st0 unit 0 family inet address 10.0.1.2/24
set interfaces lo0 unit 0 family inet address 192.168.2.2/32
```

Security Zones & Host-Inbound

```
set security zones security-zone UNTRUST interfaces ge-0/0/0.0
set security zones security-zone UNTRUST interfaces ge-0/0/0.0
host-inbound-traffic system-services ike
set security zones security-zone UNTRUST interfaces ge-0/0/0.0
host-inbound-traffic system-services ping
set security zones security-zone UNTRUST interfaces st0.0
set security zones security-zone UNTRUST interfaces st0.0
host-inbound-traffic system-services ping
set security zones security-zone UNTRUST interfaces st0.0
host-inbound-traffic protocols bgp
```

IKE (Phase 1)

```
set security ike proposal IKE-PROPOSAL authentication-method
pre-shared-keys
set security ike proposal IKE-PROPOSAL dh-group group14
set security ike proposal IKE-PROPOSAL authentication-
algorithm sha-256
set security ike proposal IKE-PROPOSAL encryption-algorithm
aes-256-cbc
set security ike policy IKE-POLICY mode main
set security ike policy IKE-POLICY proposals IKE-PROPOSAL
set security ike policy IKE-POLICY pre-shared-key ascii-text
"Juniper"
set security ike gateway IKE-GATEWAY ike-policy IKE-POLICY
set security ike gateway IKE-GATEWAY address 1.1.1.1
set security ike gateway IKE-GATEWAY external-interface ge-
0/0/0.0
set security ike gateway IKE-GATEWAY version v1-only
set security ike gateway IKE-GATEWAY dead-peer-detection
interval 10
set security ike gateway IKE-GATEWAY dead-peer-detection
threshold 3
```

IPsec (Phase 2)

```
set security ipsec proposal IPSEC-PROPOSAL protocol esp
set security ipsec proposal IPSEC-PROPOSAL authentication-
algorithm hmac-sha-256-128
```

```
set security ipsec proposal IPSEC-PROPOSAL encryption-
algorithm aes-256-cbc
set security ipsec policy IPSEC-POLICY proposals IPSEC-
PROPOSAL
set security ipsec vpn IPSEC-VPN bind-interface st0.0
set security ipsec vpn IPSEC-VPN ike gateway IKE-GATEWAY
set security ipsec vpn IPSEC-VPN ike ipsec-policy IPSEC-POLICY
set security ipsec vpn IPSEC-VPN perfect-forward-secrecy keys
group14
```

BGP (over st0)

```
set routing-options autonomous-system 200
set protocols bgp group EBGP type external
set protocols bgp group EBGP peer-as 100
set protocols bgp group EBGP neighbor 10.0.1.1
```

Security Policies for Transit Traffic

If you will pass **LAN↔LAN** through the VPN (beyond BGP to the box), put st0.0 in a dedicated zone (e.g., VPN) and add policies. Minimal permissive example (same-zone example if you keep both in UNTRUST is not needed for BGP-to-box, but is needed for transit):

Example: create VPN zone and move st0.0 there (recommended for clarity)

```
set security zones security-zone VPN interfaces st0.0
set security zones security-zone VPN host-inbound-traffic
system-services ping
set security zones security-zone VPN host-inbound-traffic
protocols bgp
```

Example policies (adjust source/destination subnets)

```
set security policies from-zone VPN to-zone UNTRUST policy
VPN-TO-UNTRUST match source-address any destination-address
any application any
set security policies from-zone VPN to-zone UNTRUST policy
VPN-TO-UNTRUST then permit
set security policies from-zone UNTRUST to-zone VPN policy
UNTRUST-TO-VPN match source-address any destination-address
any application any
set security policies from-zone UNTRUST to-zone VPN policy
UNTRUST-TO-VPN then permit
```

Quick Bring-Up Validation (immediately after commit)

IKE / IPsec up? (from SRX-A)

```
device@vSRX-0> show security ike security-associations
```

```
Index   State  Initiator cookie  Responder cookie  Mode
Remote Address
3544408 UP     622e26f4b005aba1  721ff1b2bfa849a6  Main
1.1.1.1

device@vSRX-0> show bgp summary
Threading mode: BGP I/O
Default eBGP mode: advertise - accept, receive - accept
Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp
State    Pending
inet.0
                       0          0          0          0
0          0
Peer                     AS      InPkt     OutPkt    OutQ
Flaps Last Up/Dwn State|#Active/Received/Accepted/Damped...
10.0.1.1                100          4          4       0
0        1:10 Establ
  inet.0: 0/0/0/0

device@vSRX-0> show security ipsec security-associations
  Total active tunnels: 1     Total Ipsec sas: 1
  ID    Algorithm       SPI      Life:sec/kb  Mon lsys Port
Gateway
  <131073 ESP:aes-cbc-256/sha256 91e4cf8d 3514/ unlim - root
500 1.1.1.1
  >131073 ESP:aes-cbc-256/sha256 5b944da 3514/ unlim - root
500 1.1.1.1
```

IKE / IPsec up? (from SRX-A)

```
device@vSRX-0> show security ike security-associations
Index   State  Initiator cookie  Responder cookie  Mode
Remote Address
3544408 UP     622e26f4b005aba1  721ff1b2bfa849a6  Main
1.1.1.1
```

```
device@vSRX-0> show bgp summary
Threading mode: BGP I/O
Default eBGP mode: advertise - accept, receive - accept
Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp
State    Pending
inet.0
                       0          0          0          0
0          0
Peer                     AS      InPkt     OutPkt    OutQ
Flaps Last Up/Dwn State|#Active/Received/Accepted/Damped...
10.0.1.1                100          4          4       0
0        1:10 Establ
  inet.0: 0/0/0/0

device@vSRX-0> show security ipsec security-associations
  Total active tunnels: 1     Total Ipsec sas: 1
  ID    Algorithm       SPI      Life:sec/kb  Mon lsys Port
Gateway
  <131073 ESP:aes-cbc-256/sha256 91e4cf8d 3514/ unlim - root
500 1.1.1.1
  >131073 ESP:aes-cbc-256/sha256 5b944da 3514/ unlim - root
500 1.1.1.1
```

Tunnel reachability (from SRX-A & SRX-B)

```
device@vSRX-0> ping 10.0.1.2 source 10.0.1.1
PING 10.0.1.2 (10.0.1.2): 56 data bytes
64 bytes from 10.0.1.2: icmp_seq=0 ttl=64 time=0.846 ms
64 bytes from 10.0.1.2: icmp_seq=1 ttl=64 time=0.466 ms
^C
--- 10.0.1.2 ping statistics ---
8 packets transmitted, 8 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.402/0.502/0.846/0.132 ms

jcluser@vSRX-0> ping 10.0.1.1 source 10.0.1.2
PING 10.0.1.1 (10.0.1.1): 56 data bytes
64 bytes from 10.0.1.1: icmp_seq=0 ttl=64 time=0.832 ms
64 bytes from 10.0.1.1: icmp_seq=1 ttl=64 time=0.445 ms
64 bytes from 10.0.1.1: icmp_seq=2 ttl=64 time=0.479 ms
^C
--- 10.0.1.1 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.445/0.585/0.832/0.175 ms
```

BGP up?

```
device@vSRX-0> show bgp neighbor 10.0.1.2
Peer: 10.0.1.2+52485 AS 200    Local: 10.0.1.1+179 AS 100
  Group: EBGP                  Routing-Instance: master
  Forwarding routing-instance: master
  Type: External    State: Established    Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Options: <PeerAS Refresh>
  Options: <GracefulShutdownRcv>
  Holdtime: 90 Preference: 170
  Graceful Shutdown Receiver local-preference: 0
  Number of flaps: 0
  Peer ID: 192.168.2.2     Local ID: 192.168.1.1       Active
Holdtime: 90
  Keepalive Interval: 30         Group index: 0    Peer index:
0    SNMP index: 0
  I/O Session Thread: bgpio-0 State: Enabled
  BFD: disabled, down
  Local Interface: st0.0
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-unicast
  Peer supports Refresh capability (2)
  Stale routes from peer are kept for: 300
  Peer does not support Restarter functionality
  Restart flag received from the peer: Notification
  NLRI that restart is negotiated for: inet-unicast
  NLRI of received end-of-rib markers: inet-unicast
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer does not support LLGR Restarter functionality
  Peer supports 4 byte AS extension (peer-as 200)
  Peer does not support Addpath
  Table inet.0 Bit: 20000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              0
    Received prefixes:            0
    Accepted prefixes:            0
    Suppressed due to damping:    0
    Advertised prefixes:          0
  Last traffic (seconds): Received 8    Sent 8    Checked 978
  Input messages:  Total 39     Updates 1       Refreshes 0
Octets 789
  Output messages: Total 38     Updates 0       Refreshes 0
Octets 726
  Output Queue[1]: 0            (inet.0, inet-unicast)
```

Conclusion:

This document explained IPsec fundamentals, the differences between IKEv1 and IKEv2, and how to configure and validate BGP over IPsec on Juniper SRX devices. IPsec remains a cornerstone of network security, ensuring confidentiality, integrity, and authentication of data across untrusted networks.

While IKEv1 continues to serve legacy deployments, IKEv2 should be the preferred choice for all modern implementations because of its efficiency, scalability, and built-in support for NAT traversal and mobility.

For successful deployments:
- Use strong cryptographic proposals (AES-256, SHA-256, DH group14 or higher).
- Adopt IKEv2 wherever supported.
- Validate both configuration and runtime states using the provided checklists.

By following these best practices, organizations can achieve secure, stable, and future-ready VPN connectivity.

In this document, we explored the fundamentals of IPsec, its working modes, and the difference between IKEv1 and IKEv2. We saw how IPsec provides confidentiality, integrity, and authentication, making it one of the most reliable technologies for secure communication over untrusted networks.

While **IKEv1** is still in use for legacy systems, it is complex and less efficient due to its multi-phase message exchange. **IKEv2**, on the other hand, offers simplified negotiation, faster setup, better NAT traversal, and built-in support for mobility—making it the preferred standard for modern deployments.

For practical deployments, it is recommended to:

Use **IKEv2** whenever supported, for efficiency and security.
Carefully select encryption, authentication, and hashing algorithms based on organizational policies.
Test interoperability between devices before production rollout.

In conclusion, IPsec continues to remain a cornerstone of network security, and choosing the right version (preferably IKEv2) ensures both **strong protection and operational simplicity** for enterprises and service providers alike.