Virtual Private Networks

Qui je suis?

- **Background**
  - Master SRIV - Université Lyon 1
  - Ingénieur Système (Docker, K8S, Ansible)
  - Entrepreneur et développeur open-source blockchain
    - Rust, Substrate, Nodejs, React
- **Actuellement**
  - Doctorant - Inria / ENS de Lyon / Équipe Avalon
  - **Efficacité énergétique du cloud**
    - Dans le cadre de défi entre ***Inria*** et ***OVHcloud***
- **Page perso et mail**
  - https://vladost.com
  - pro@vladost.com

Vladimir Ostapenco

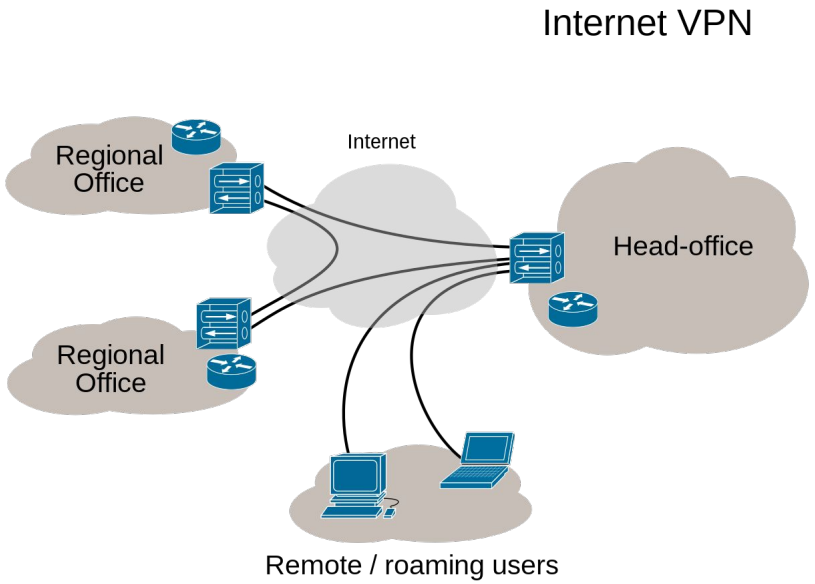
Planning

1. Introduction
2. GRE (Generic Routing Encapsulation)
3. IPSec
4. OpenVPN
5. WireGuard

- **CM**: 1h
- **TP**: 4h

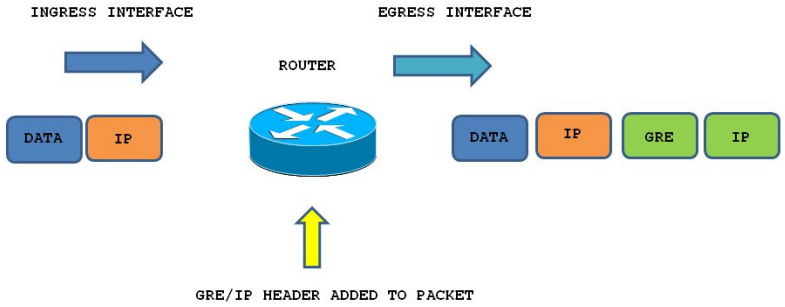
Introduction: Principe du VPN

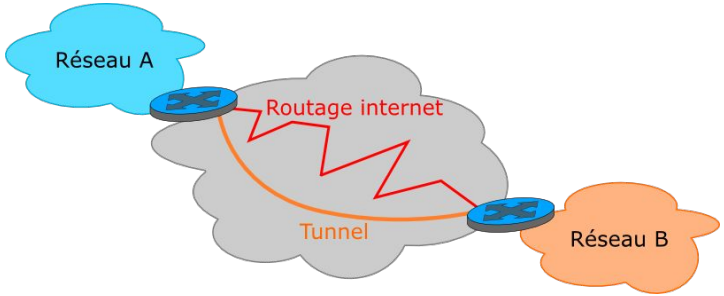
- Réseau privé virtuel (VPN) permet de
  - Simuler un réseau privé sur une infrastructure à accès partagé (Internet)
  - Interconnecter de réseaux répartis sur de grandes distances géographiques
  - Fournir un accès sécurisé au réseau de l'entreprise
- Repose sur un protocole appelé « **protocole de tunneling** »
  - Consiste à construire un chemin virtuel entre l'émetteur et le destinataire
  - Permet de faire circuler les informations de façon sécurisée d'un bout à l'autre du tunnel



Introduction: Exemple d’un tunnel

- Encapsulation d'un protocole dans un autre protocole de même niveau (IP dans IP)
- Tunnel est établi entre deux routeurs
- Première couche IP circule normalement sur l'internet et transporte une seconde couche IP
- Sur la première couche tout se passe comme si les deux routeurs communiquent directement
- Grâce à ce tunnel, tout nœud du réseau A peut communiquer avec tout nœud du réseau B, les deux réseaux étant construits avec des adresses IP **privées**



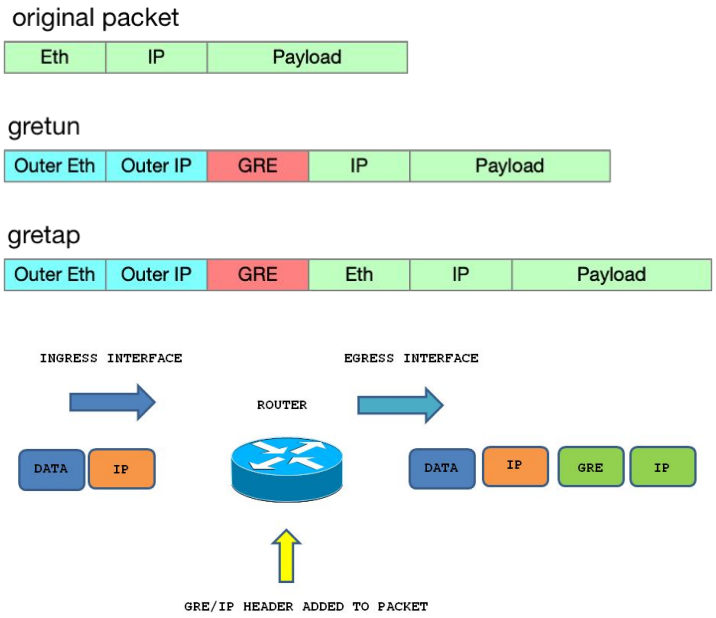


Introduction: Pourquoi faire du VPN?

- Interconnection des réseaux distants
- Connection sécurisé au réseau de l’entreprise
- Communications sécurisés
- Faire du L2 au dessus de L3
- Changer d’IP pour être identifié dans un autre pays
- Forcer le passage dans un équipement de sécurité

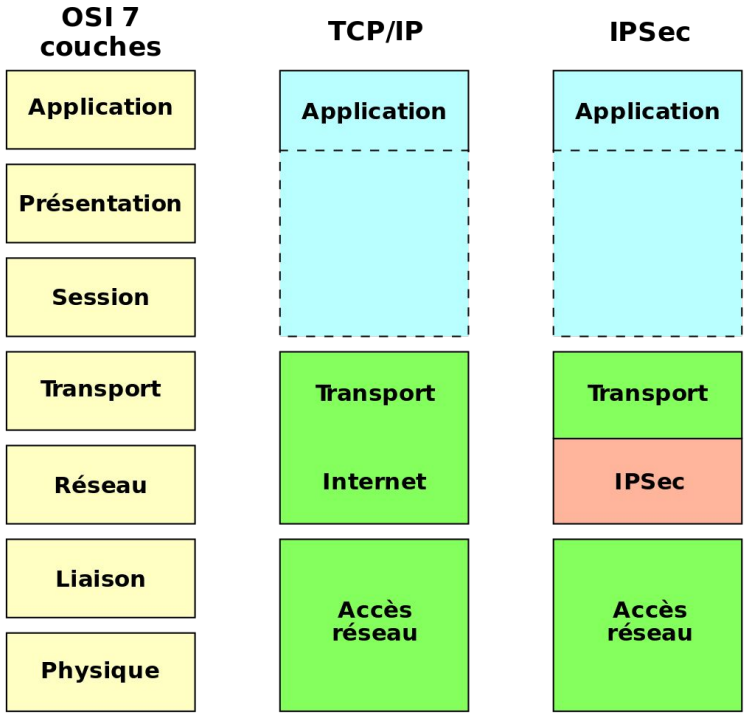
GRE (Generic Routing Encapsulation)

- Protocole générique de tunneling
- Dévélopé par Cisco
- Conçu pour encapsuler
  - IP (*gretun*)
  - Ethernet (*gretap*)
- N'implémente pas de chiffrement ou d'authentification
  - Doit être utilisé avec IPsec
- Est sans etat



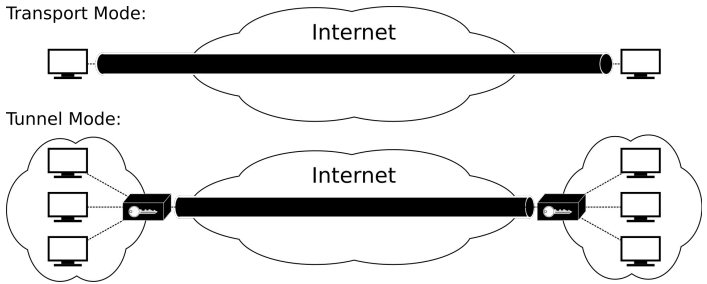
IPsec

- Framework ouvert pour assurer des communications privées sécurisées
- Assure la confidentialité, l'intégrité et l'authenticité des données sur un réseau public
- Norme prévue pour IPv6 mais adaptée pour IPv4
- Vise à sécuriser l'échange de données au niveau de la couche **Réseau** (niveau 3)
- Permet de créer des VPN sécurisés et sécuriser les accès distants
- Généralement implémenté dans le noyau



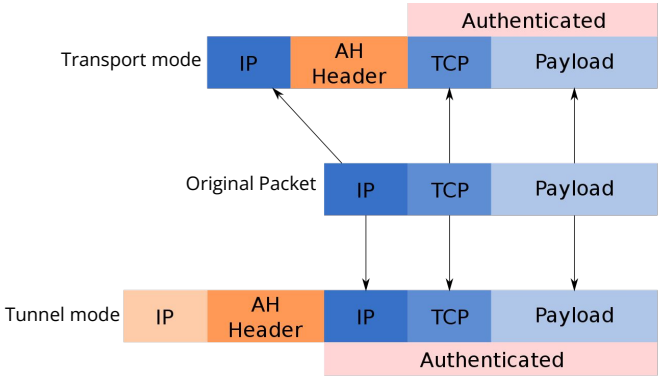
IPsec - Modes de fonctionnement

- **Mode transport**
  - Connection hôte à hôte
- **Mode tunnel**
  - Tunnel réseau
  - Utilisé pour
    - Communication de réseau à réseau (création des VPNs)
    - Communication d’hôte à réseau (accès à distance d'un utilisateur)
    - Communication hôte à hôte (messagerie privée)



IPsec - Modes de fonctionnement

- **Mode transport**
  - Uniquement les données transférées sont chiffrées et/ou authentifiées (la partie *payload* du paquet IP)
- **Mode tunnel**
  - Totalité du paquet IP qui est chiffré et/ou authentifié
  - Paquet est encapsulé dans un nouveau paquet IP avec un nouvel en-tête IP

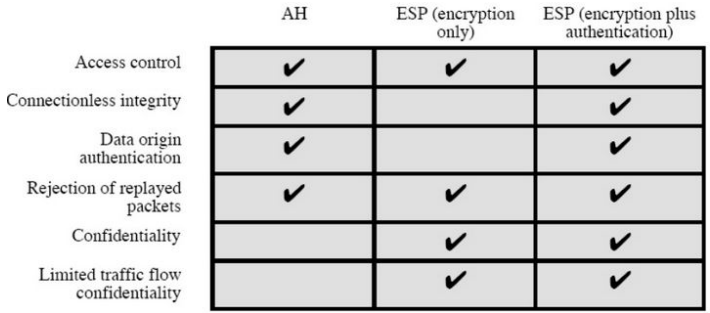


IPsec - Composants

- **Mécanismes de sécurité**
  - Authentication Header (AH)
  - Encapsulation Security Payload (ESP)
- **Bases de données internes**
  - Security Association Database (SAD)
  - Security Policy Database (SPD)
- **Protocole d'échange de clés**
  - Internet Key Exchange (IKE)

IPsec - Mécanismes de sécurité

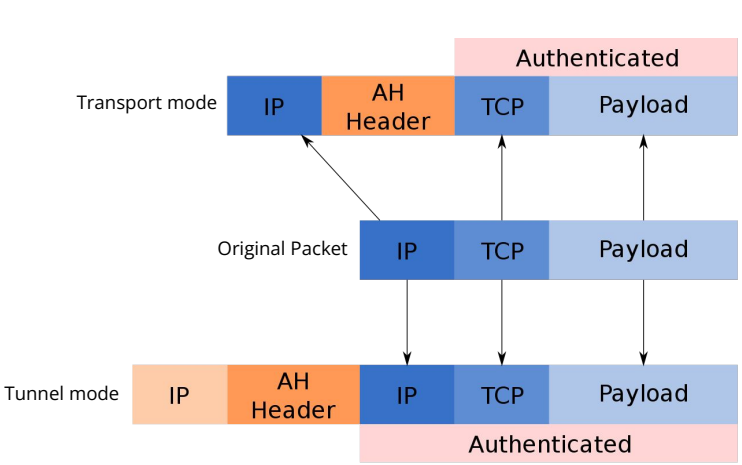
- **AH (Authentication Header)**
  - Intégrité des données - à l'aide d'un hash de message
  - Authenticité des données - à l'aide d'une clé secrète partagée
  - Unicité des données - en utilisant un champ de numéro de séquence dans l'en-tête
  - Pas de confidentialité!
  - Algorithmes d'authentification: *HMAC_MD5*, *HMAC_SHA1/2*, *AES128_XCBC*, *AES_GMAC*...
- **ESP (Encapsulating Security Payload)**
  - Confidentialité (Chiffrement des données)
  - Intégrité, authenticité et unicité des données
  - Algorithmes de chiffrement: *DES, 3DES, AES_CBC, AES_GCM...*



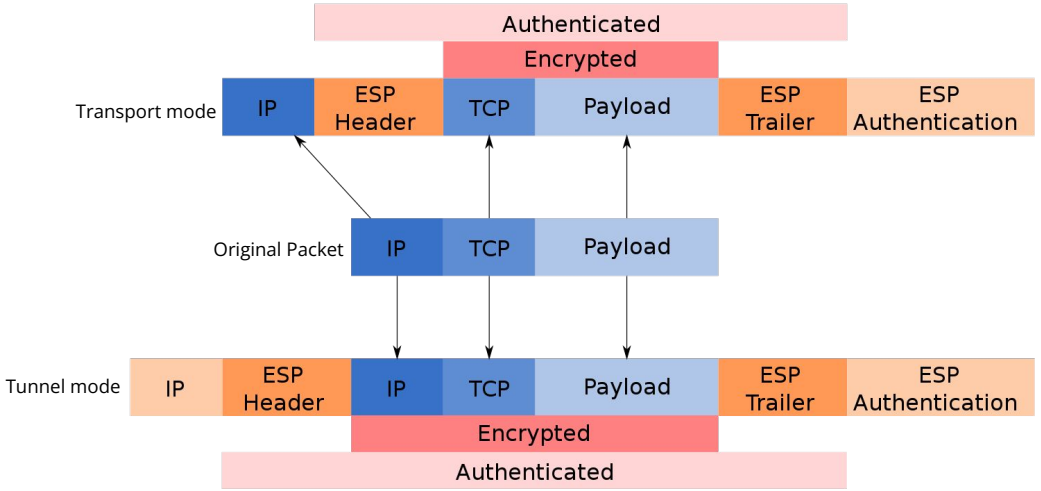
	AH	ESP (encryption only)	ESP (encryption plus authentication)
Access control	✔	✔	✔
Connectionless integrity	✔		✔
Data origin authentication	✔		✔
Rejection of replayed packets	✔	✔	✔
Confidentiality		✔	✔
Limited traffic flow confidentiality		✔	✔

IPsec - Mécanismes de sécurité

AH (Authentication Header)



ESP (Encapsulating Security Payload)



IPsec - Security Association (SA)

- IPsec utilise le concept de **Security Association (SA)** pour gérer **les paramètres et les clés des mécanismes de sécurité**
- **Security Association (SA)**
  - Stocke l'ensemble des paramètres associés à une communication donnée
  - Est unidirectionnelle (deux SA pour protéger deux sens de communications)
  - Une SA par protocole de sécurité (AH ou ESP)
  - Identifiée de manière unique à l'aide d'un triplet
    - Index des paramètres de sécurité (Security Parameter Index, SPI)
    - Adresse de destination des paquets
    - Identifiant du protocole de sécurité utilisé (AH ou ESP)
- SA actives sont stockés dans « base de données des associations de sécurité » (***Security Association Database, SAD***)

Security Associations

Direction	Outbound	Inbound
SPI	1000	1001
Destination Address	POP Server B	Node A
IPsec Protocol	ESP	ESP
Algorithm	3DES-CBC	3DES-CBC
Key	The secret key from A to B	The secret key from B to A
Mode	Transport	Transport

IPsec - Security Policy Database (SPD)

- Base de données **indiquant le niveau de protection à appliquer** aux paquets
    - Permet de décider, pour chaque paquet, s'il doit être sécurisé avec IPsec, autorisé à passer sans être sécurisé ou rejeté
- Contient un ensemble de règles (***Security Policies***) qui détermine si un paquet est soumis au traitement IPsec et gère les détails du traitement
    - Si le paquet doit être protégé par IPsec, il détermine également quelle SA spécifique le paquet doit utiliser
- Établie et maintenue par un utilisateur, un administrateur système ou une application

Règles SPD

Direction	Outbound	Inbound
Source Address	Node A	POP Server B
Destination Address	POP server B	Node A
Upper Layer Protocol	TCP	TCP
Upper Layer Source Port	Any	POP3
Upper Layer Destination Port	POP3	Any
IPsec Protocol	ESP	ESP
Mode	Transport	Transport

IPsec - Gestion des clés

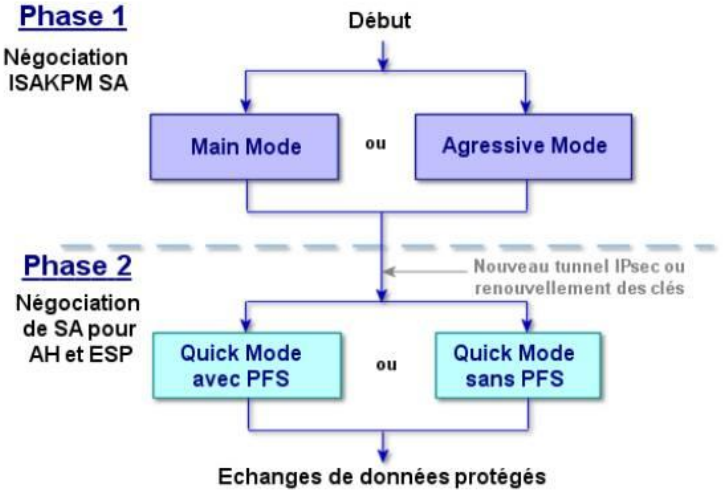
- Protocoles de sécurité ont recours à des algorithmes cryptographiques et ont donc besoin de clés
- **Problématique:** se mettre d'accord sur les algorithmes, les paramètres et clés à utiliser
- Solutions
  - Gestion des paramètres et des clés manuelle
  - Gestion dynamique à l'aide d'un protocole sécurisé (comme IKE)

IPsec - IKE (Internet Key Exchange)

- Protocole permettant de négocier un accord sur les protocoles, les algorithmes et les clés à utiliser
- Gère et échange des clés en toute sécurité
- Permet l’établissement de SA à travers un réseau non sécurisé
- Basé sur l'amélioration des protocoles ***ISAKMP/Oakley***
- Permet d’échanger les clés via
    - **Clés pré-partagées (PSK)**
    - **Certificats (X.509)**

IPsec - IKE - Fonctionnement

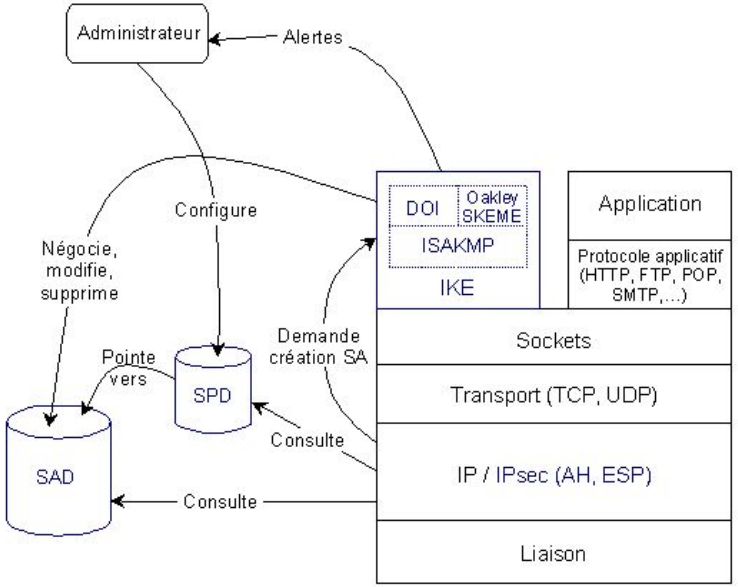
- **Phase 1 -** Négociation ISAKMP
  - Attributs suivants sont négociés
    - Algorithme de chiffrement, fonction de hachage, méthode d'authentification et groupe pour Diffie-Hellman
  - Trois clés sont générées
    - Une pour le chiffrement
    - Une pour l'authentification
    - Une pour la dérivation d'autres clés
- **Phase 2 -** Négociation de SA pour des protocoles de sécurité (AH et ESP)
  - Chaque négociation aboutit à deux SA
  - Une dans chaque sens de la communication



IPsec - Fonctionnement

Trafic sortant

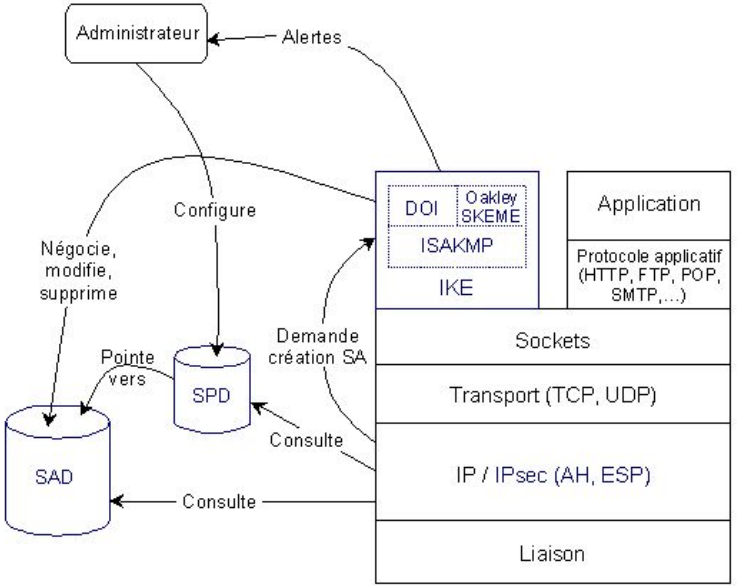
- IPsec reçoit des paquets à envoyer
- Consulte la base de données des politiques de sécurité (SPD) pour savoir comment traiter ces paquets
- Si il faut appliquer des mécanismes de sécurité, récupère les caractéristiques requises pour la SA correspondante et consulte la base des SA (SAD)
- Si la SA nécessaire existe déjà, elle est utilisée pour traiter le paquet en question
- Sinon, IPsec fait appel à IKE pour établir une nouvelle SA avec les caractéristiques requises



IPsec - Fonctionnement

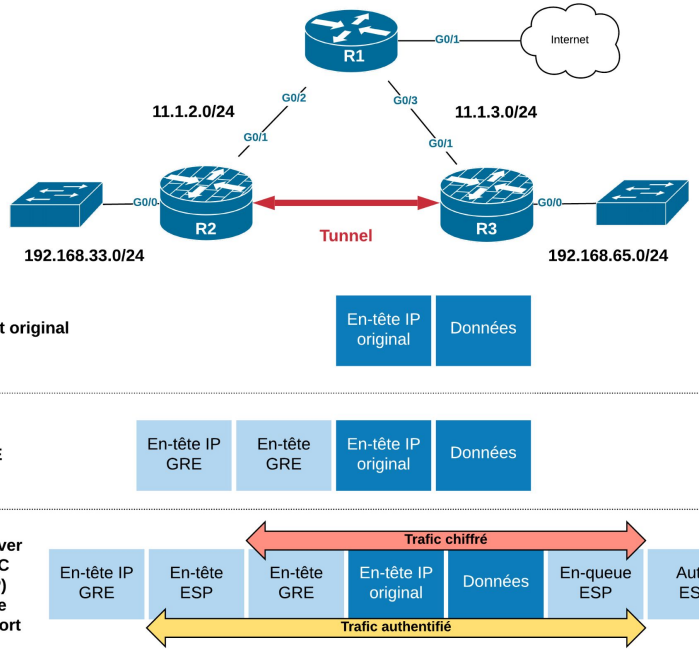
Trafic entrant

- IPsec reçoit un paquet en provenance du réseau
- Examine l'en tête pour savoir si ce paquet s'est vu appliquer un ou plusieurs mécanismes de sécurité
- Si oui, quelles sont les références de la SA
- Consulte la SAD pour connaître les paramètres à utiliser pour la vérification et/ou le déchiffrement du paquet
- Une fois le paquet vérifié et/ou déchiffré, la SPD est consultée pour savoir si l'association de sécurité appliquée au paquet correspondait bien à celle requise par les politiques de sécurité



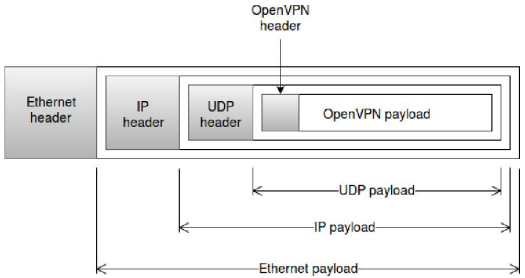
GRE avec IPsec

- **GRE IPsec Tunnel Mode**
  - Ensemble du paquet GRE (qui comprend le paquet d'en-tête IP d'origine) est encapsulé, chiffré et protégé dans un paquet IPsec
  - Surcharge importante ajoutée au paquet (76 octets)
- **GRE IPsec Transport Mode**
  - Paquet GRE est encapsulé et chiffré à l'intérieur du paquet IPsec, cependant, l'en-tête IP GRE est placé à l'avant
  - Moins de surcharge



OpenVPN

- Système VPN open source permettant de créer des connexions point-à-point ou site-à-site sécurisées
- Authentification et chiffrement avec OpenSSL SSL/TLS
  - Moins efficace que IPsec, mais plus facile à mettre en oeuvre
- Architecture client/serveur
- VPN espace utilisateur
  - Crée une interface réseau virtuelle dans l'espace utilisateur à chaque extrémité du réseau
- Supporte TCP et UDP pour transmettre les données
- **Méthodes d'authentification**
  - Clés secrètes pré-partagées (PSK)
  - Certificats (PKI)
  - Nom d'utilisateur/mot de passe (nécessite un module externe)



OpenVPN: Modes

- **Mode TAP**
  - Traffic Ethernet
- **Mode TUN**
  - Traffic IP
  - Topologies réseau disponibles
    - **Net30 (obsolète)** - ancienne topologie pour la prise en charge des clients Windows exécutant les clients 2.0.9 ou antérieurs, chaque client se voit attribuer un /30 virtuel
    - **P2P** - tous les nœuds sont configurés comme de vrais liens point à point, un seul /30 entre le client et le serveur, pas utilisable avec Windows
    - **Subnet** - topologie actuelle recommandée, comme dans un réseau classique, l'interface *tun* est configuré avec une adresse IP et un netmask, un réseau /24 dans lequel se trouvent les clients

OpenVPN: Communication entre plusieurs réseaux

- Comment un serveur peut annoncer un LAN derrière lui?
  - Configurer le serveur pour annoncer le réseau derrière lui aux clients
    - **push "route…"** dans la configuration du serveur
- Comment un client peut annoncer un LAN derrière lui?
  - Client doit avoir le routage IP activé
  - Configurer le serveur pour router le LAN du client sur le VPN
    - **route** et **push "route …"** dans la configuration du serveur
  - Indiquer au serveur quel client est responsable du réseau avec **iroute**
- **iroute** est une route interne à OpenVPN qui indique au serveur quel client est responsable de quel réseau
  - Ajoutée à une entrée **CCD** (Configuration du serveur pour un client spécifique)

WireGuard

- Solution VPN sécurisée, la plus facile à utiliser et la plus simple de l'industrie
  - Plus performante que OpenVPN
  - Plus simple que IPsec
- Implémenté dans le noyau Linux
  - Intégré dans le noyau Linux officiel à partir de la version 5.6
- Encapsulé en toute sécurité les paquets IP via UDP (pas de TCP)
- Authentification avec une paire des clés privées/publiques (x25519)
  - Ne dispose pas de mécanismes de distribution ou de configuration de clés
- Fonctionne en ajoutant une interface réseau virtuel (ou plusieurs), appelée **wgX**

WireGuard - Interface réseau

- Chaque interface réseau est configuré avec
  - Clé privée
  - Port d'écoute
  - Liste de pairs
- Chaque pair est configuré avec
  - Clé publique
  - Liste des adresses autorisés
  - Endpoint (*optionnel*)

Configuration de l'interface Wireguard

```
[Interface]
PrivateKey = gI6EdUSYvn8ugXOt8QQD6Yc+JyiZxIhp3GInSWRfWGE=
ListenPort = 51820

[Peer]
PublicKey = HIgo9xNzJMWLKASShiTqIybxZ0U3wGLiUeJ1PKf8ykw=
AllowedIPs = 10.192.122.4/32, 192.168.0.0/16
Endpoint = 192.95.5.69:51820
```

WireGuard - Cryptokey Routing

- Utilise le concept appelé **Cryptokey Routing** qui simplifie la gestion du réseau et le contrôle d'accès
  - Associe des clés publiques à une liste d'adresses IP autorisées à l'intérieur du tunnel
  - **Lors de l'envoi de paquets**
    - Liste des IP autorisées se comporte comme une sorte de table de routage
    - Clé publique est utilisée pour chiffrer le paquet
  - **Lors de la réception de paquets**
    - Clé publique est utilisée pour déchiffrer et authentifier le paquet
    - Liste des IP autorisées se comporte comme une sorte de liste de contrôle d'accès

Configuration de l'interface Wireguard

```
[Interface]
PrivateKey = yAnz5TF+lXXJte14tji3zlMNq+hd2rYUIgJBgB3fBmk=
ListenPort = 51820

[Peer]
PublicKey = TrMvSoP4jYQlY6RIzBgbssQqY3vxI2Pi+y71lOWWXX0=
AllowedIPs = 10.192.122.4/32, 192.168.0.0/16

[Peer]
PublicKey = gN65BkIKy1eCE9pP1wdc8ROUtkHLF2PfAqYdyYBz6EA=
AllowedIPs = 10.10.10.230/32
```

WireGuard - Fonctionnement

Sur le serveur

- Lorsque l'interface veut envoyer un paquet à un pair (un client)
  - Voir si IP de destination est dans la liste des IP autorisées pour voir à quel pair l'envoyer
  - Chiffrer le paquet à l'aide de la clé publique du pair
  - Envoyer au Endpoint le plus récent de ce pair
- Lorsqu'un paquet est reçu par le serveur du pair
  - Paquet est déchiffré et authentifié avec la clé publique du pair
  - Si son IP source est dans la liste des IP autorisées pour ce pair, alors le paquet est autorisé sur l'interface
    - Sinon le paquet est rejeté

Configuration Serveur

```
[Interface]
PrivateKey = yAnz5TF+lXXJte14tji3zlMNq+hd2rYUIgJBgB3fBmk=
ListenPort = 51820

[Peer]
PublicKey = TrMvSoP4jYQlY6RIzBgbssQqY3vxI2Pi+y71lOWWXX0=
AllowedIPs = 10.192.122.4/32, 192.168.0.0/16

[Peer]
PublicKey = gN65BkIKy1eCE9pP1wdc8ROUtkHLF2PfAqYdyYBz6EA=
AllowedIPs = 10.10.10.230/32
```

Configuration Client

```
[Interface]
PrivateKey = gI6EdUSYvn8ugXOt8QQD6Yc+JyiZxIhp3GInSWRfWGE=
ListenPort = 51820

[Peer]
PublicKey = HIgo9xNzJMWLKASShiTqIybxZ0U3wGLiUeJ1PKf8ykw=
Endpoint = 192.95.5.69:51820
AllowedIPs = 0.0.0.0/0
```

WireGuard - Fonctionnement

Sur le client

- Lorsque l'interface veut envoyer un paquet à son seul pair (le serveur) avec n'importe quelle adresse IP de destination
    - Chiffrer le paquet à l'aide de la clé publique du son seul pair (le serveur)
    - Envoyer au Endpoint le plus récent de ce pair
- Lorsqu'un paquet est reçu du serveur avec n'importe quelle adresse IP source
    - Paquet est déchiffré et authentifié avec la clé publique du serveur
    - Si le paquet est correctement déchiffré et authentifié, il est autorisé sur l'interface car les adresses IP autorisées du pair sont 0.0.0.0/0

Configuration Serveur

```
[Interface]
PrivateKey = yAnz5TF+lXXJte14tji3zlMNq+hd2rYUIgJBgB3fBmk=
ListenPort = 51820

[Peer]
PublicKey = TrMvSoP4jYQlY6RIzBgbssQqY3vxI2Pi+y71IOWWXX0=
AllowedIPs = 10.192.122.4/32, 192.168.0.0/16

[Peer]
PublicKey = gN65BkIKy1eCE9pP1wdc8ROUtkHLF2PfAqYdyYBz6EA=
AllowedIPs = 10.10.10.230/32
```

Configuration Client

```
[Interface]
PrivateKey = gI6EdUSYvn8ugXOt8QQD6Yc+JyiZxIhp3GInSWRfWGE=
ListenPort = 51820

[Peer]
PublicKey = HIgo9xNzJMWLKASShiTqIybxZ0U3wGLiUeJ1PKf8ykw=
Endpoint = 192.95.5.69:51820
AllowedIPs = 0.0.0.0/0
```

Merci pour votre attention!

Références

- https://irp.nain-t.net/doku.php/280vpn:start
- https://www.frameip.com/vpn/
- https://www.frameip.com/ipsec/
- https://www.sans.org/white-papers/1459/
- https://community.openvpn.net/openvpn/wiki/GettingStartedwithOVPN
- https://www.wireguard.com/