Anytun - Secure Anycast Tunneling

Christian Pointner

19. Mai 2015

Überblick

OpenVPN

- eines der ältesten SSL/TLS basierten VPNs
- viele Features
- Multi-Platform
- große Community
- wird von Coverity regelmäßig geprüft
- Firma die Support bietet

OpenVPN

- eines der ältesten SSL/TLS basierten VPNs
- viele Features
- Multi-Platform
- große Community
- wird von Coverity regelmäßig geprüft
- Firma die Support bietet

OpenVPN

- eines der ältesten SSL/TLS basierten VPNs
- viele Features
- Multi-Platform
- große Community
- wird von Coverity regelmäßig geprüft
- Firma die Support bietet

OpenVPN

- eines der ältesten SSL/TLS basierten VPNs
- viele Features
- Multi-Platform
- große Community
- wird von Coverity regelmäßig geprüft
- Firma die Support bietet

OpenVPN

- eines der ältesten SSL/TLS basierten VPNs
- viele Features
- Multi-Platform
- große Community
- wird von Coverity regelmäßig geprüft
- Firma die Support bietet

OpenVPN

- eines der ältesten SSL/TLS basierten VPNs
- viele Features
- Multi-Platform
- große Community
- wird von Coverity regelmäßig geprüft
- Firma die Support bietet

tinc

- neueres Projekt - weniger Legacy Code
- basiert auf SSL/TLS
- unterstützt automatisches Meshing (P2P)
- kleinere Community
- Multi-Platform (weniger als OpenVPN)

tinc

- neueres Projekt - weniger Legacy Code
- basiert auf SSL/TLS
- unterstützt automatisches Meshing (P2P)
- kleinere Community
- Multi-Platform (weniger als OpenVPN)

tinc

- neueres Projekt - weniger Legacy Code
- basiert auf SSL/TLS
- unterstützt automatisches Meshing (P2P)
- kleinere Community
- Multi-Platform (weniger als OpenVPN)

tinc

- neueres Projekt - weniger Legacy Code
- basiert auf SSL/TLS
- unterstützt automatisches Meshing (P2P)
- kleinere Community
- Multi-Platform (weniger als OpenVPN)

tinc

- neueres Projekt - weniger Legacy Code
- basiert auf SSL/TLS
- unterstützt automatisches Meshing (P2P)
- kleinere Community
- Multi-Platform (weniger als OpenVPN)

strongSwan

- IPsec Implementierung für Linux
- basiert auf freeS/WAN
- unterstützt IKEv1 und IKEv2
- getestet mit vielen verschiedenen IPsec Implementierungen

strongSwan

- IPsec Implementierung für Linux
- basiert auf freeS/WAN
- unterstützt IKEv1 und IKEv2
- getestet mit vielen verschiedenen IPsec Implementierungen

strongSwan

- IPsec Implementierung für Linux
- basiert auf freeS/WAN
- unterstützt IKEv1 und IKEv2
- getestet mit vielen verschiedenen IPsec Implementierungen

strongSwan

- IPsec Implementierung für Linux
- basiert auf freeS/WAN
- unterstützt IKEv1 und IKEv2
- getestet mit vielen verschiedenen IPsec Implementierungen

ipsec-tools

- Key Exchange Daemons/Tools für IPsec im Linux Kernel
- raccoon: unterstützt nur IKEv1
- setkey: für "statisches" keying

ipsec-tools

- Key Exchange Daemons/Tools für IPsec im Linux Kernel
- raccoon: unterstützt nur IKEv1
- setkey: für "statisches" keying

ipsec-tools

- Key Exchange Daemons/Tools für IPsec im Linux Kernel
- raccoon: unterstützt nur IKEv1
- setkey: für “statisches” keying

Überblick

Warum Anytun

- OpenVPN und tinc basieren auf SSL/TLS
  - Angriffe gegen SSL/TLS treffen eventuell auch das VPN
  - kein standardisiertes Protokoll
- IPSec kann nicht mit Anycast umgehen (keine Replay Protection oder hoher Synchronisationsaufwand)
- NAT Transversal ist in IPSec umständlich und schwierig
- Anycast bietet viele Vorteile bei Load Balancing und Redundanz
- ein Protokoll das Anycast unterstüzt kann in vielen Szenarien eingesetzt werden.

Warum Anytun

- OpenVPN und tinc basieren auf SSL/TLS
  - Angriffe gegen SSL/TLS treffen eventuell auch das VPN
  - kein standardisiertes Protokoll
- IPSec kann nicht mit Anycast umgehen (keine Replay Protection oder hoher Synchronisationsaufwand)
- NAT Transversal ist in IPSec umständlich und schwierig
- Anycast bietet viele Vorteile bei Load Balancing und Redundanz
- ein Protokoll das Anycast unterstüzt kann in vielen Szenarien eingesetzt werden.

Warum Anytun

- OpenVPN und tinc basieren auf SSL/TLS
  - Angriffe gegen SSL/TLS treffen eventuell auch das VPN
  - kein standardisiertes Protokoll
- IPSec kann nicht mit Anycast umgehen (keine Replay Protection oder hoher Synchronisationsaufwand)
- NAT Transversal ist in IPSec umständlich und schwierig
- Anycast bietet viele Vorteile bei Load Balancing und Redundanz
- ein Protokoll das Anycast unterstüzt kann in vielen Szenarien eingesetzt werden.

Überblick

SATP

- Name des Protokolls: Secure Anycast Tunneling Protocol
- dokumentiert als Internet Draft
- spezifiziert nur die Payload Kommunikation und das NAT Transversal
- agnostisch gegenüber dem Key Exchange Protokoll

SATP

- Name des Protokolls: Secure Anycast Tunneling Protocol
- dokumentiert als Internet Draft
- spezifiziert nur die Payload Kommunikation und das NAT Transversal
- agnostisch gegenüber dem Key Exchange Protokoll

SATP

- Name des Protokolls: Secure Anycast Tunneling Protocol
- dokumentiert als Internet Draft
- spezifiziert nur die Payload Kommunikation und das NAT Transversal
- agnostisch gegenüber dem Key Exchange Protokoll

SATP

- Name des Protokolls: Secure Anycast Tunneling Protocol
- dokumentiert als Internet Draft
- spezifiziert nur die Payload Kommunikation und das NAT Transversal
- agnostisch gegenüber dem Key Exchange Protokoll

SATP Details

- Ähnlich zu IPSec/ESP und GRE
- geringer Overhead (10 Byte + MAC)
- über UDP oder direkt auf IP
- unterstützt jedes Ethertype Protocol als Payload (Ethernet, IPv4, IPv6, ...)
- verschiedene Anycast Sender können identifiziert werden somit funktioniert die Replay Protection auch ohne viel Synchronisationsaufwand
- Crypto basiert auf sRTP (RFC 3711)

SATP Details

- Ähnlich zu IPSec/ESP und GRE
- geringer Overhead (10 Byte + MAC)
- über UDP oder direkt auf IP
- unterstützt jedes Ethertype Protocol als Payload (Ethernet, IPv4, IPv6, ...)
- verschiedene Anycast Sender können identifiziert werden somit funktioniert die Replay Protection auch ohne viel Synchronisationsaufwand
- Crypto basiert auf sRTP (RFC 3711)

SATP Details

- Ähnlich zu IPSec/ESP und GRE
- geringer Overhead (10 Byte + MAC)
- über UDP oder direkt auf IP
- unterstützt jedes Ethertype Protocol als Payload (Ethernet, IPv4, IPv6, ...)
- verschiedene Anycast Sender können identifiziert werden somit funktioniert die Replay Protection auch ohne viel Synchronisationsaufwand
- Crypto basiert auf sRTP (RFC 3711)

SATP Details

- Ähnlich zu IPSec/ESP und GRE
- geringer Overhead (10 Byte + MAC)
- über UDP oder direkt auf IP
- unterstützt jedes Ethertype Protocol als Payload (Ethernet, IPv4, IPv6, ...)
- verschiedene Anycast Sender können identifiziert werden somit funktioniert die Replay Protection auch ohne viel Synchronisationsaufwand
- Crypto basiert auf sRTP (RFC 3711)

SATP Details

- Ähnlich zu IPSec/ESP und GRE
- geringer Overhead (10 Byte + MAC)
- über UDP oder direkt auf IP
- unterstützt jedes Ethertype Protocol als Payload (Ethernet, IPv4, IPv6, ...)
- verschiedene Anycast Sender können identifiziert werden somit funktioniert die Replay Protection auch ohne viel Synchronisationsaufwand
- Crypto basiert auf sRTP (RFC 3711)

SATP Details

- Ähnlich zu IPSec/ESP und GRE
- geringer Overhead (10 Byte + MAC)
- über UDP oder direkt auf IP
- unterstützt jedes Ethertype Protocol als Payload (Ethernet, IPv4, IPv6, ...)
- verschiedene Anycast Sender können identifiziert werden somit funktioniert die Replay Protection auch ohne viel Synchronisationsaufwand
- Crypto basiert auf sRTP (RFC 3711)

Anytun

- erste Implementierung von SATP
- unterstützt Cluster Synchronisation
- geschrieben in C++ / Boost (Multi-Threaded)
- läuft unter Linux, Windows, FreeBSD

Anytun

- erste Implementierung von SATP
- unterstützt Cluster Synchronisation
- geschrieben in C++ / Boost (Multi-Threaded)
- läuft unter Linux, Windows, FreeBSD

Anytun

- erste Implementierung von SATP
- unterstützt Cluster Synchronisation
- geschrieben in C++ / Boost (Multi-Threaded)
- läuft unter Linux, Windows, FreeBSD

Anytun

- erste Implementierung von SATP
- unterstützt Cluster Synchronisation
- geschrieben in C++ / Boost (Multi-Threaded)
- läuft unter Linux, Windows, FreeBSD

uAnytun

- kleine Implementierung von SATP
- keine Cluster Synchronisation
- geschrieben in C mit wenig Abhängigkeiten (nur Crypto)
- läuft unter Linux, FreeBSD und OpenBSD

uAnytun

- kleine Implementierung von SATP
- keine Cluster Synchronisation
- geschrieben in C mit wenig Abhängigkeiten (nur Crypto)
- läuft unter Linux, FreeBSD und OpenBSD

uAnytun

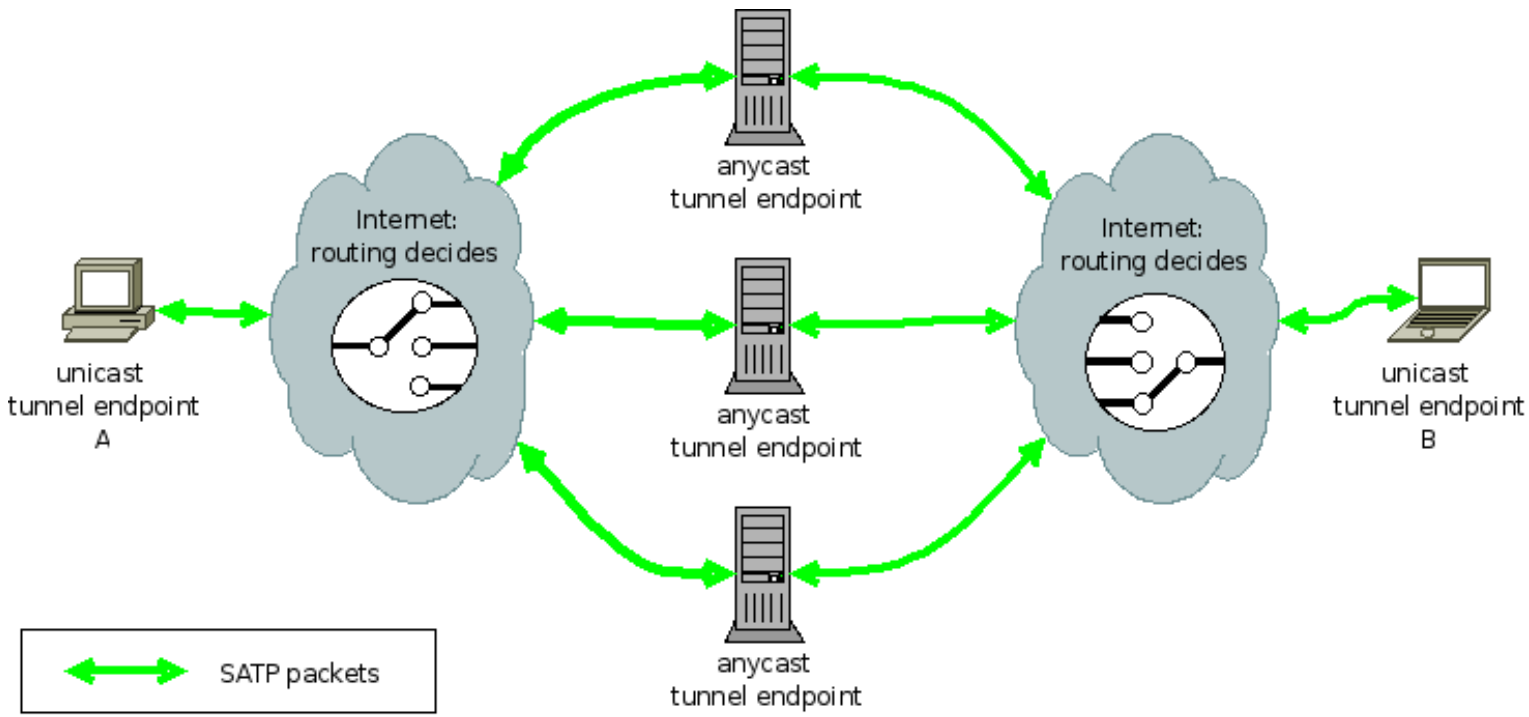
- kleine Implementierung von SATP
- keine Cluster Synchronisation
- geschrieben in C mit wenig Abhängigkeiten (nur Crypto)
- läuft unter Linux, FreeBSD und OpenBSD

uAnytun

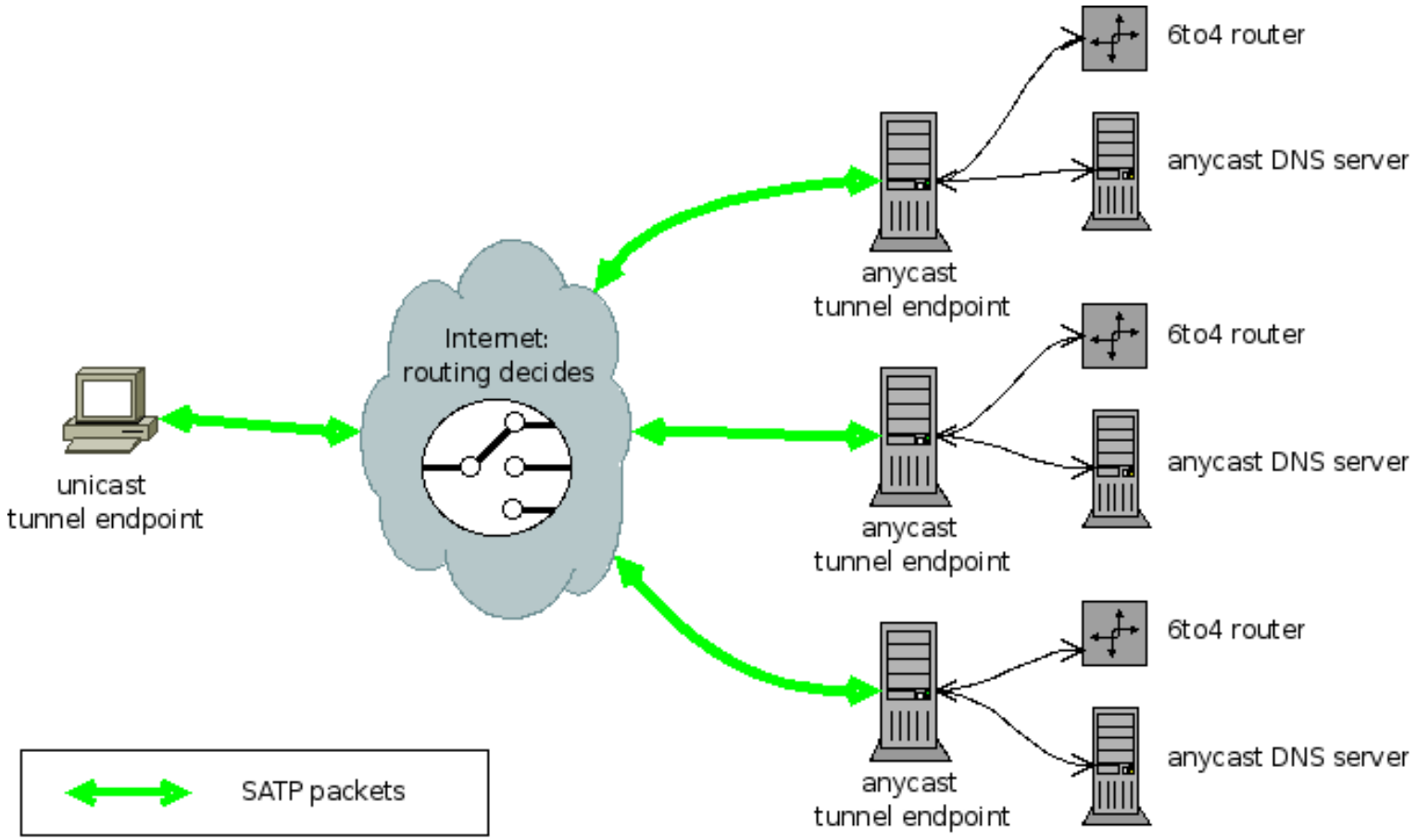
- kleine Implementierung von SATP
- keine Cluster Synchronisation
- geschrieben in C mit wenig Abhängigkeiten (nur Crypto)
- läuft unter Linux, FreeBSD und OpenBSD

Überblick

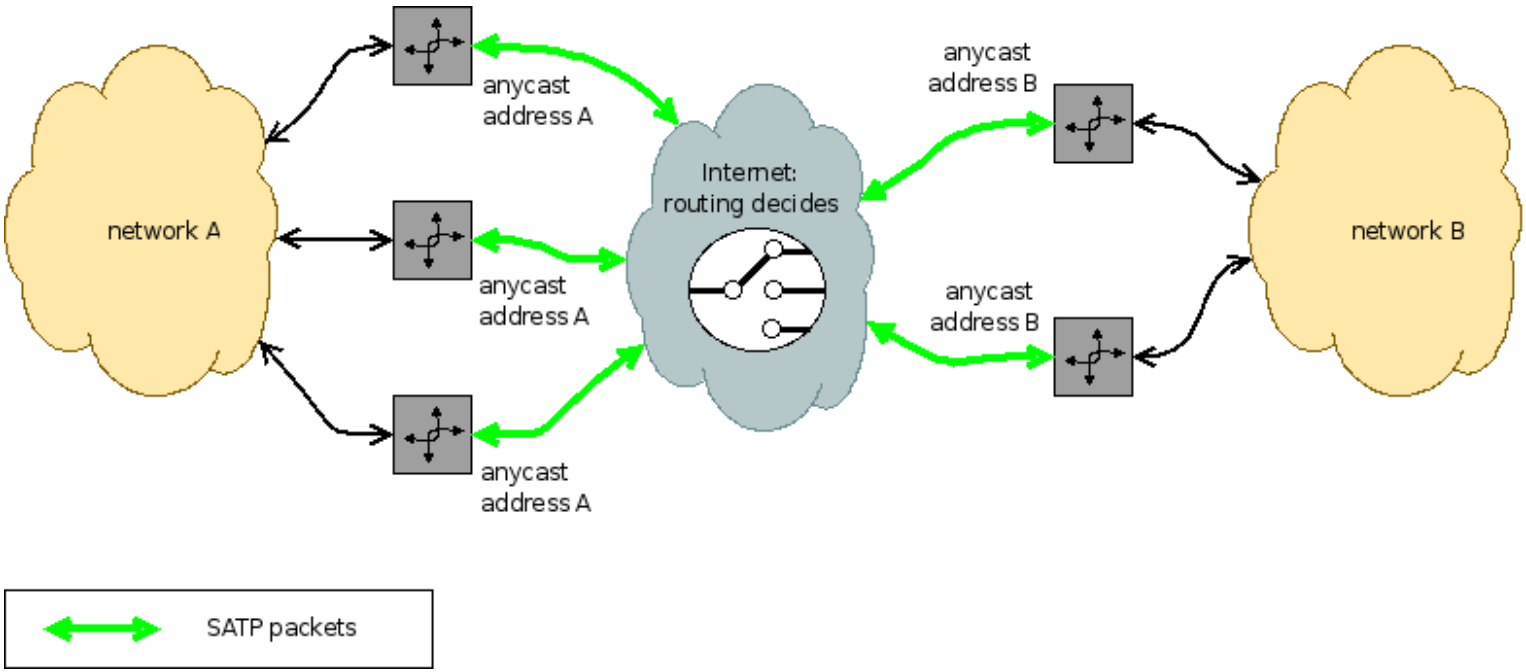
Anycast VPN Cluster



Secure Anycast Service/Application



Mulit-Path Interconnect



Überblick

RAIL

- Redundant Array of Inexpensive Links
- SATP Pakete werden mehrmals über verschiedene Pfade gesendet
- Empfänger verwirft alle doppelten Pakete
- Redundanz und oder gesteigerte Bandbreite
- erste Test Implementierung in uAnytun funktioniert
- ohne Key Exchange sehr umständlich zu verwenden :(

RAIL

- Redundant Array of Inexpensive Links
- SATP Pakete werden mehrmals über verschiedene Pfade gesendet
- Empfänger verwirft alle doppelten Pakete
- Redundanz und oder gesteigerte Bandbreite
- erste Test Implementierung in uAnytun funktioniert
- ohne Key Exchange sehr umständlich zu verwenden :(

RAIL

- Redundant Array of Inexpensive Links
- SATP Pakete werden mehrmals über verschiedene Pfade gesendet
- Empfänger verwirft alle doppelten Pakete
- Redundanz und oder gesteigerte Bandbreite
- erste Test Implementierung in uAnytun funktioniert
- ohne Key Exchange sehr umständlich zu verwenden :(

RAIL

- Redundant Array of Inexpensive Links
- SATP Pakete werden mehrmals über verschiedene Pfade gesendet
- Empfänger verwirft alle doppelten Pakete
- Redundanz und oder gesteigerte Bandbreite
- erste Test Implementierung in uAnytun funktioniert
- ohne Key Exchange sehr umständlich zu verwenden :(

RAIL

- Redundant Array of Inexpensive Links
- SATP Pakete werden mehrmals über verschiedene Pfade gesendet
- Empfänger verwirft alle doppelten Pakete
- Redundanz und oder gesteigerte Bandbreite
- erste Test Implementierung in uAnytun funktioniert
- ohne Key Exchange sehr umständlich zu verwenden :(

RAIL

- Redundant Array of Inexpensive Links
- SATP Pakete werden mehrmals über verschiedene Pfade gesendet
- Empfänger verwirft alle doppelten Pakete
- Redundanz und oder gesteigerte Bandbreite
- erste Test Implementierung in uAnytun funktioniert
- ohne Key Exchange sehr umständlich zu verwenden :(

Key-Exchange, RFC, ...

- zurzeit untstützt Anytun und uAnytun nur statische Keys :(
- Interface für externes Keyexchange um Inbound Daten schicken zu können
- Interface um Keys im Daemon zu installieren
- Update der Crypto Primitives in SATP
- RFC für SATP
- Threading Modell von Anytun

Key-Exchange, RFC, ...

- zurzeit untstützt Anytun und uAnytun nur statische Keys :(
- Interface für externes Keyexchange um Inbound Daten schicken zu können
- Interface um Keys im Daemon zu installieren
- Update der Crypto Primitives in SATP
- RFC für SATP
- Threading Modell von Anytun

Key-Exchange, RFC, ...

- zurzeit untstützt Anytun und uAnytun nur statische Keys :(
- Interface für externes Keyexchange um Inbound Daten schicken zu können
- Interface um Keys im Daemon zu installieren
- Update der Crypto Primitives in SATP
- RFC für SATP
- Threading Modell von Anytun

Key-Exchange, RFC, ...

- zurzeit untstützt Anytun und uAnytun nur statische Keys :(
- Interface für externes Keyexchange um Inbound Daten schicken zu können
- Interface um Keys im Daemon zu installieren
- Update der Crypto Primitives in SATP
- RFC für SATP
- Threading Modell von Anytun

Key-Exchange, RFC, ...

- zurzeit untstützt Anytun und uAnytun nur statische Keys :(
- Interface für externes Keyexchange um Inbound Daten schicken zu können
- Interface um Keys im Daemon zu installieren
- Update der Crypto Primitives in SATP
- RFC für SATP
- Threading Modell von Anytun

Key-Exchange, RFC, ...

- zurzeit untstützt Anytun und uAnytun nur statische Keys :(
- Interface für externes Keyexchange um Inbound Daten schicken zu können
- Interface um Keys im Daemon zu installieren
- Update der Crypto Primitives in SATP
- RFC für SATP
- Threading Modell von Anytun