Discussion topics, Linux IPsec Workshop

Steffen Klassert
secunet Security Networks AG
Dresden

Linux IPsec Workshop, Dresden, March 26, 2018
Future of PFKEY in the kernel

Configurable system policy default (allow/drop)

Crypto layer problems

Hardware GRO
Future of PFKEY in the kernel

- PFKEY is buggy
- Google syscall fuzzer reports more and more (security related) bugs
- No active development since more that 10 years
- Do we still need to support PFKEY, and if yes how long?
- What do we need to do to be able to remove PKKEY from the kernel?
- How do we handle the PFKEY bug reports until we can remove it?
Future of PFKEY in the kernel

- PFKEY is buggy
  - Google syscall fuzzer reports more and more (security related) bugs
  - No active development since more that 10 years
  - Do we still need to support PFKEY, and if yes how long?
  - What do we need to do to be able to remove PFKEY from the kernel?
  - How do we handle the PFKEY bug reports until we can remove it?
Future of PFKEY in the kernel

- PFKEY is buggy
- Google syscall fuzzer reports more and more (security related) bugs
- No active development since more than 10 years
- Do we still need to support PFKEY, and if yes how long?
- What do we need to do to be able to remove PFKEY from the kernel?
- How do we handle the PFKEY bug reports until we can remove it?
Future of PFKEY in the kernel

- PFKEY is buggy
- Google syscall fuzzer reports more and more (security related) bugs
- No active development since more than 10 years
- Do we still need to support PFKEY, and if yes how long?
- What do we need to do to be able to remove PKKEY from the kernel?
- How do we handle the PFKEY bug reports until we can remove it?
Future of PFKEY in the kernel

- PFKEY is buggy
- Google syscall fuzzer reports more and more (security related) bugs
- No active development since more that 10 years
- Do we still need to support PFKEY, and if yes how long?
- What do we need to do to be able to remove PKKEY from the kernel?
- How do we handle the PFKEY bug reports until we can remove it?
Future of PFKEY in the kernel

- PFKEY is buggy
- Google syscall fuzzer reports more and more (security related) bugs
- No active development since more that 10 years
- Do we still need to support PFKEY, and if yes how long?
- What do we need to do to be able to remove PKKEY from the kernel?
- How do we handle the PFKEY bug reports until we can remove it?
Future of PFKEY in the kernel

- PFKEY is buggy
- Google syscall fuzzer reports more and more (security related) bugs
- No active development since more than 10 years
- Do we still need to support PFKEY, and if yes how long?
- What do we need to do to be able to remove PFKEY from the kernel?
- How do we handle the PFKEY bug reports until we can remove it?
The current default behaviour is to allow traffic if there is no matching policy.

A patch that make the default configurable (allow/drop) exists.

Each direction can be configured separately (input/output/forward).

When default is block, we need allow policies for all packet flows we accept.

Would this be usefull for the userspace?
The current default behaviour is to allow traffic if there is no matching policy

A patch that makes the default configurable (allow/drop) exists

Each direction can be configured separately (input/output/forward)

When default is block, we need allow policies for all packet flows we accept

Would this be useful for the userspace?
Configurable system policy default (allow/drop)

- The current default behaviour is to allow traffic if there is no matching policy
- A patch that make the default configurable (allow/drop) exists
- Each direction can be configured separately (input/output/forward)
- When default is block, we need allow policies for all packet flows we accept
- Would this be useful for the userspace?
The current default behaviour is to allow traffic if there is no matching policy.

A patch that makes the default configurable (allow/drop) exists.

Each direction can be configured separately (input/output/forward).

When default is block, we need allow policies for all packet flows we accept.

Would this be useful for the userspace?
The current default behaviour is to allow traffic if there is no matching policy

A patch that make the default configurable (allow/drop) exists

Each direction can be configured separately (input/output/forward)

When default is block, we need allow policies for all packet flows we accept

Would this be useful for the userspace?
The current default behaviour is to allow traffic if there is no matching policy.

A patch that make the default configurable (allow/drop) exists.

Each direction can be configured separately (input/output/forward).

When default is block, we need allow policies for all packet flows we accept.

Would this be useful for the userspace?
Crypto layer problems

- There is a lot of memcpy in the crypto layer
- IV generators copy if src and dst buffer are different
- Some algorithm implementations are not able to do SG operations
- Might be worth to do some performance optimizations in the crypto layer
- IPsec performance optimizations are 'eaten up' in the crypto layer
Crypto layer problems

- There is a lot of memcpy in the crypto layer
- IV generators copy if src and dst buffer are different
- Some algorithm implementations are not able to do SG operations
- Might be worth to do some performance optimizations in the crypto layer
- IPsec performance optimizations are ’eaten up’ in the crypto layer
Crypto layer problems

- There is a lot of memcpy in the crypto layer
- IV generators copy if src and dst buffer are different
- Some algorithm implementations are not able to do SG operations
- Might be worth to do some performance optimizations in the crypto layer
- IPsec performance optimizations are ’eaten up’ in the crypto layer
Crypto layer problems

- There is a lot of memcpy in the crypto layer
- IV generators copy if src and dst buffer are different
- Some algorithm implementations are not able to do SG operations
- Might be worth to do some performance optimizations in the crypto layer
- IPsec performance optimizations are 'eaten up' in the crypto layer
Crypto layer problems

- There is a lot of memcpy in the crypto layer
- IV generators copy if src and dst buffer are different
- Some algorithm implementations are not able to do SG operations
- Might be worth to do some performance optimizations in the crypto layer
- IPsec performance optimizations are ’eaten up’ in the crypto layer
Crypto layer problems

- There is a lot of memcpy in the crypto layer
- IV generators copy if src and dst buffer are different
- Some algorithm implementations are not able to do SG operations
- Might be worth to do some performance optimizations in the crypto layer
- IPsec performance optimizations are ’eaten up’ in the crypto layer
Hardware GRO

- Hardware GRO: Routeable version of LRO
- Middleboxes could benefit from receive side HW offload too
- Infrastructure was introduced recently
- Do the NIC vendors plan to support it???
Hardware GRO

- Hardware GRO: Routeable version of LRO
- Middleboxes could benefit from receive side HW offload too
- Infrastructure was introduced recently
- Do the NIC vendors plan to support it???
Hardware GRO

- Hardware GRO: Routeable version of LRO
- Middleboxes could benefit from receive side HW offload too
  - Infrastructure was introduced recently
  - Do the NIC vendors plan to support it???
Hardware GRO

- Hardware GRO: Routeable version of LRO
- Middleboxes could benefit from receive side HW offload too
- Infrastructure was introduced recently
- Do the NIC vendors plan to support it???
Hardware GRO

- Hardware GRO: Routeable version of LRO
- Middleboxes could benefit from receive side HW offload too
- Infrastructure was introduced recently
- Do the NIC vendors plan to support it???