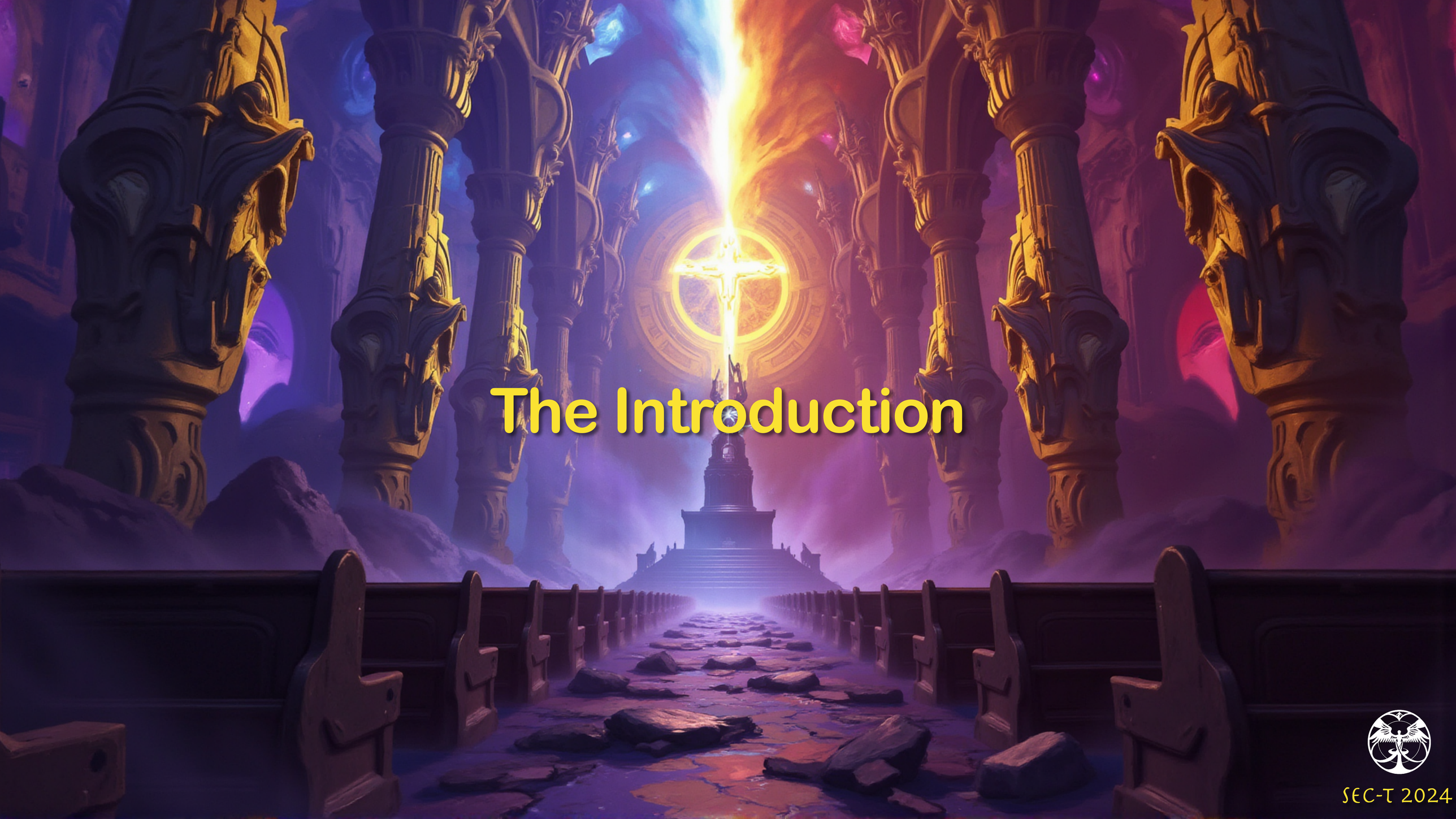

Blessed Sanctum
Save Us
SEC-T 2024


The Introduction

The Introduction

- Introduction
- The Eternal Battle
- Sanctorum Sanctum
- Holy Communion
- Worship Of The Packet
- Encrypted Vessel
- Protected Vessel
- ~~The Heretic~~

SEC-T 2024

The Introduction

- Sanctum, a VPN daemon with a few novel approaches.
  - ISC licensed, fully free and open.
  - Key exchanges are ”stateless” (can flow one-way).
  - Symmetrical keying only.
  - Fully privilege separated at every level.
  - Sandboxed with modern techniques.
- Written in good old C99.

The Introduction

- Small, reviewable and performant.
  - > 20/gbps on good hardware.
- Different modes of operation:
  - Normal tunnels between peer A and B.
  - One-way tunnels (if a data diode is in the way).
  - Authenticated relay with automatic key distribution.
- OpenBSD, MacOS, Linux.
  - x86_64, risc-v, aarch64 and sparc64.

The Introduction

- Anti-replay.
- Traffic transported with ESP in tunnel mode (UDP).
  - IETF RFC 4303.
  - NAT traversal.
- Traffic Flow Confidentiality
  - Avoid signalling what you send (padding of packets).
  - Traffic analysis prevention (outer encapsulation).

The Introduction

Easily configured:

On peer A:

```
# hymn add 01-02 tunnel 10.10.0.1/30 mtu 1422 \
  peer 1.2.3.4:1337 secret /etc/sanctum/secret
```

On peer B:

```
# hymn add 02-01 tunnel 10.10.0.2/30 mtu 1422 \
  peer 0.0.0.0:0 secret /etc/sanctum/secret
```

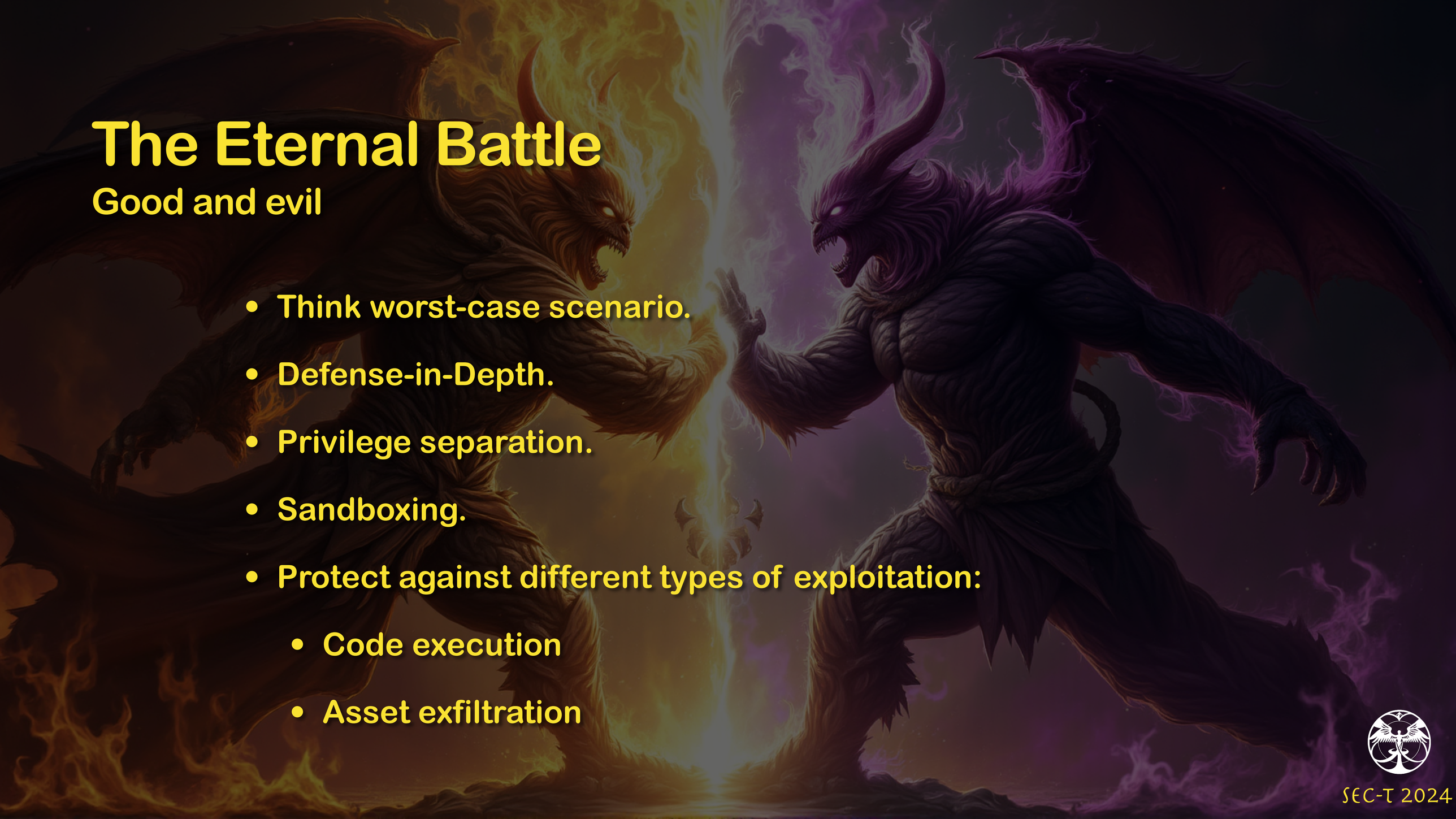
The Introduction

- Because I can.
- I didn’t like any of the solutions out there.
- WireGuard (tm) is trademarked.
  - Doesn’t feel very open and free to me.
- So, I set out to build something to replace it on my machines.
  - Something I can trust and is fully privilege separated.
- Not my first rodeo building this type of software.

The Introduction

https://sanctorum.se/

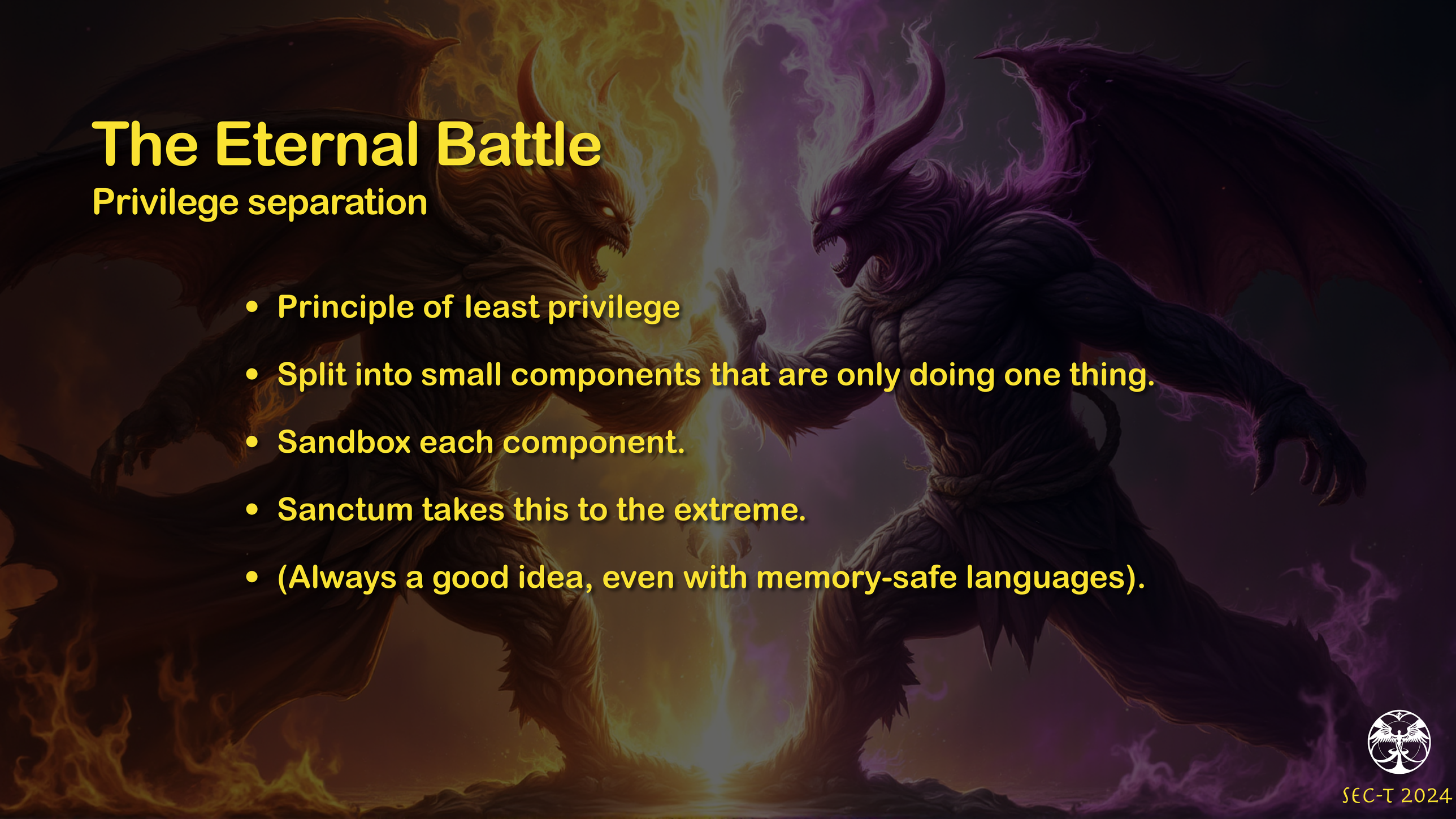

The Eternal Battle



The Eternal Battle

Good and evil

- Think worst-case scenario.
- Defense-in-Depth.
- Privilege separation.
- Sandboxing.
- Protect against different types of exploitation:
  - Code execution
  - Asset exfiltration



The Eternal Battle

Privilege separation

- Principle of least privilege
- Split into small components that are only doing one thing.
- Sandbox each component.
- Sanctum takes this to the extreme.
- (Always a good idea, even with memory-safe languages).

The Eternal Battle

Privilege separation

- Run components across different OS processes.
- Allows each process to be sandboxed appropriately.
- Each OS process gets its own address space.
  - Essentially running inside of its own ”domain”.
  - Nothing shared.
  - Sensitive assets can be protected better.
- Given that you have HW that can be trusted.

SEC-T 2024

Sanctorum Sanctum

The Components

- Identify the components
  - Sending
  - Receiving
  - Encrypting
  - Decrypting
  - Key Exchanges

Sanctorum Sanctum

The Components

- Identify the components
  - Sending → Both on clear and crypto side
  - Receiving → Both on clear and crypto side
  - Encrypting
  - Decrypting
  - Key Exchanges

Sanctorum Sanctum

Processes

purgatory-tx	Sends packets on the crypto ifc.
purgatory-rx	Receives packets from crypto ifc.
heaven-tx	Sends packets on clear ifc.
heaven-rx	Receives packets from clear ifc.
bless	Encrypts packets
confess	Decrypts packets
chapel	Performs key exchanges

Sanctorum Sanctum

Asset separation

	purgatory-tx	Sends packets on the crypto ifc.
	purgatory-rx	Receives packets from crypto ifc.
	heaven-tx	Sends packets on clear ifc.
	heaven-rx	Receives packets from clear ifc.
Encryption key	bless	Encrypts packets
Decryption key	confess	Decrypts packets
	chapel	Performs key exchanges

Sanctorum Sanctum

Asset separation

	purgatory-tx	Sends packets on the crypto ifc.
	purgatory-rx	Receives packets from crypto ifc.
	heaven-tx	Sends packets on clear ifc.
	heaven-rx	Receives packets from clear ifc.
Encryption key	bless	Encrypts packets
Decryption key	confess	Decrypts packets
Both temporarily	chapel	Performs key exchanges

Shared secret

SEC-T 2024

Sanctorum Sanctum

Putting it all together

Sanctorum Sanctum

Apply the sandbox

Sanctorum Sanctum

Apply the sandbox

Sanctorum Sanctum

Sandboxing techniques

- OpenBSD, pledge()
- It just works.
- A bit wide sometimes.
- Can only specify ”facilities”
  - Not system call specific.

Sanctorum Sanctum

Sandboxing techniques

- Linux, seccomp
  - libc has opinions, depending on which libc.
  - open() != open()
    - open() = openat() (sometimes)
  - Horrible, error-prone and absolutely terrible.
  - But it works and is the only alternative to be able to filter system calls.
- Move non-network processes into different net namespace.

Sanctorum Sanctum

Sandboxing techniques

- MacOS, hidden C API that uses LISP to instruct the sandbox.
  - Not documented so had to dig how to enable this.
  - sandbox_init_with_parameters(profile, params, …)
  - The profile is LISP, yes really.
    - Conditionals, variables, all available.

Sanctorum Sanctum

Sandboxing techniques

```
joris@gotyon sanctum % cat share/sb/purgatory-rx.sb
;; Sandbox rules for the purgatory-rx process.

(version 3)
(deny default)

(import "/usr/local/share/sanctum/sb/common.sb")

(allow syscall-unix
       (syscall-number SYS_poll)
       (syscall-number SYS_recvfrom))

(allow network-inbound (local udp4))
joris@gotyon sanctum %
```

Holy Communion

SEC-T 2024

Holy Communion

Something is shared

- IPC between components must happen.
- Sockets could be used, but not desired for performance.
- Shared memory, different flavours:
  - POSIX
    - Operates on file descriptors.
  - SysV
    - Memory is mapping into process.

Holy Communion

SysV is the way forward

- Easy to use interface.
- shmget()
  - Get new shared memory segment
- shmat()
  - Map shared memory segment into process.
- Unix-like permissions.

Holy Communion

SysV behaviour

- Inherited on fork().
- Allocate once in the parent process.
- Immediately mark for deletion.
- Won’t be deleted until last process attached to it exits.
- Unmap unneeded segments from components that don’t need it.
  - Eg: confess does not have access to purgatory-tx queue.

Holy Communion

Packet buffers

- Packet buffers are an example of data that is shared.
- They are moved from one process to another:
  - purgatory-rx → confess
  - confess → heaven-tx
  - heaven-rx → bless
  - bless → purgatory-tx

Holy Communion

The Ring And The Atomic Destiny

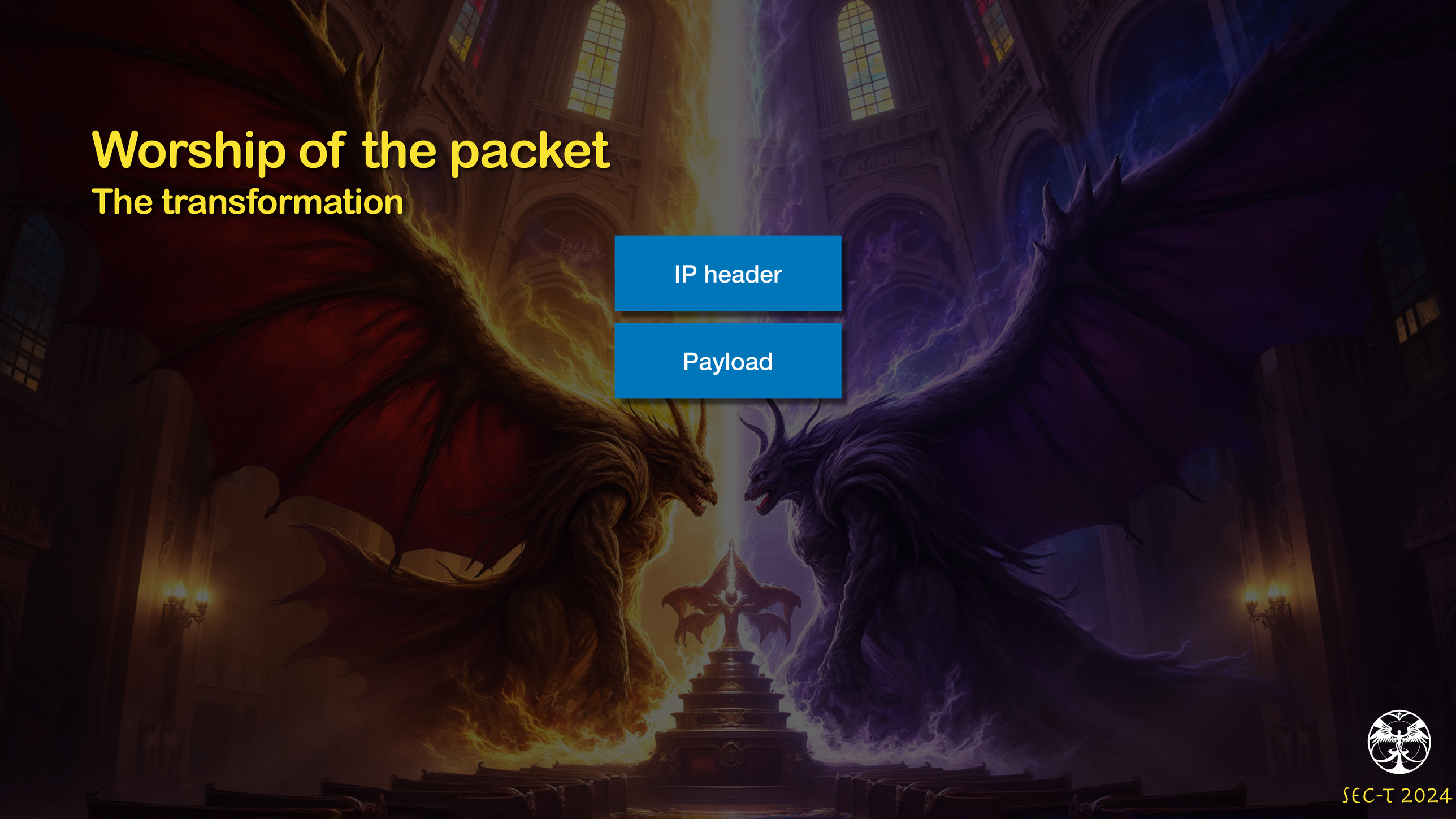
- Backed by a lock-free ring queue that is:
  - Multi-producer
  - Multi-consumer
- Implemented using atomic compiler intrinsics:
  - __atomic_store_n(..)
  - __atomic_load_n(...)
  - __atomic_compare_exchange(...)

Holy Communion


Worship of the Packet

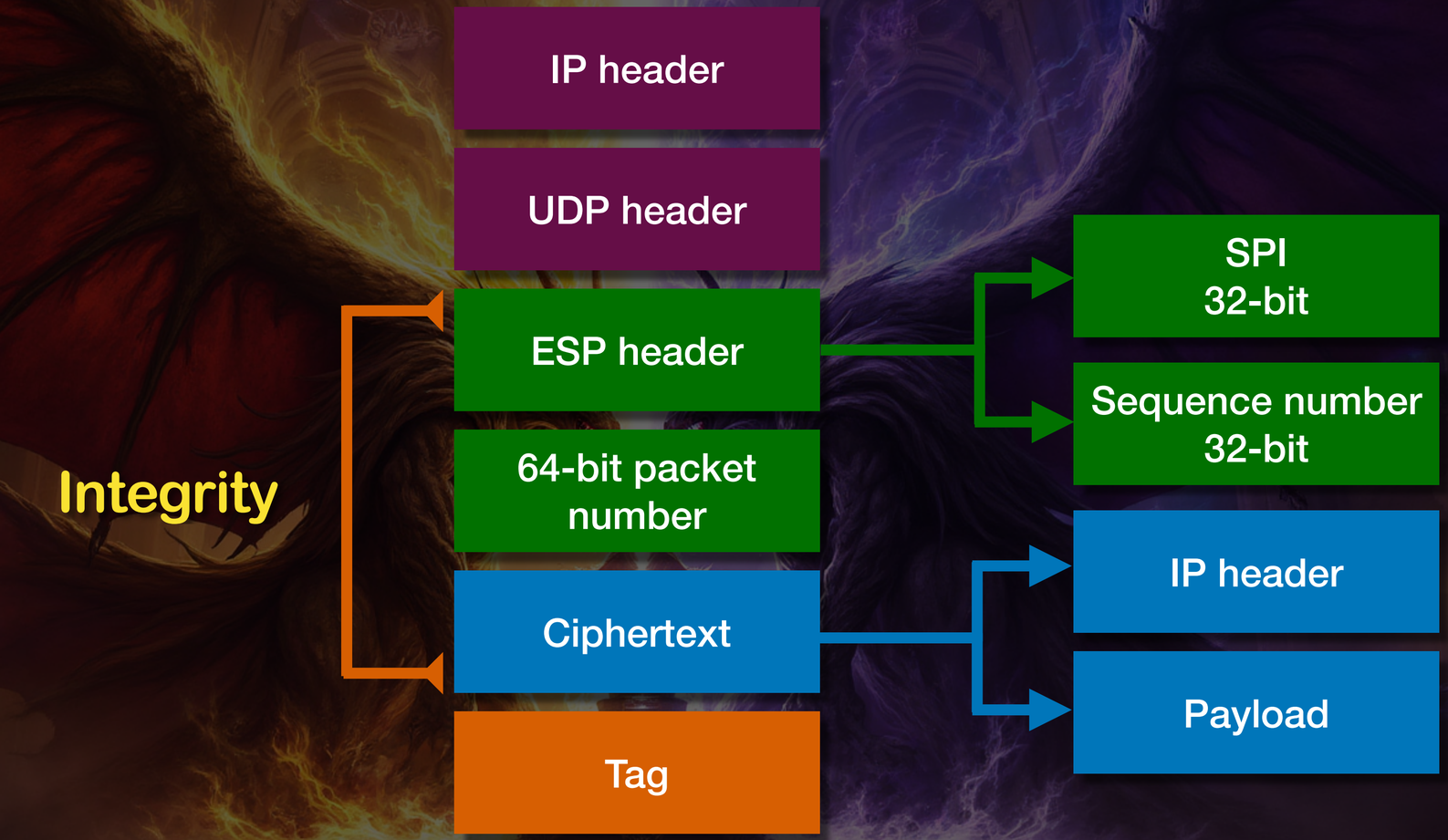
Worship of the packet

The transformation



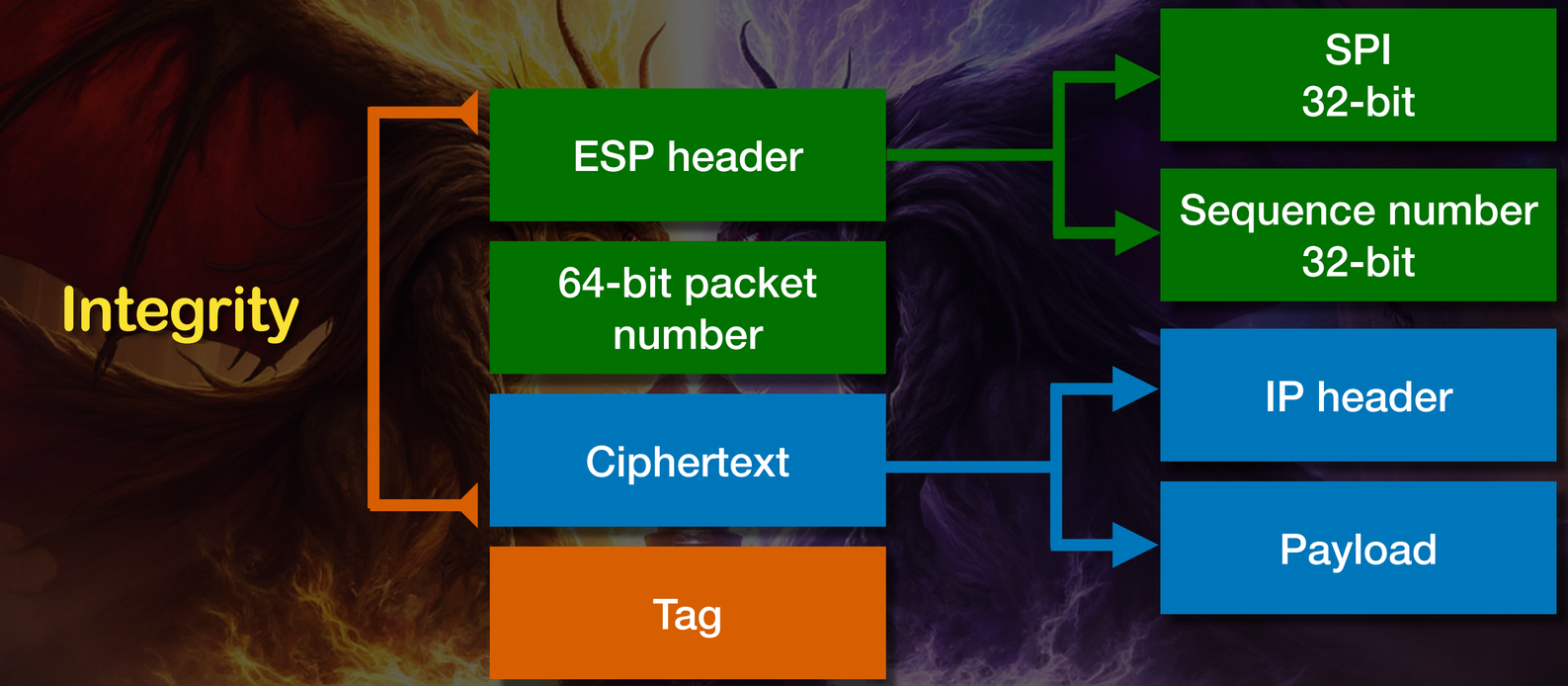
Worship of the packet

The transformation



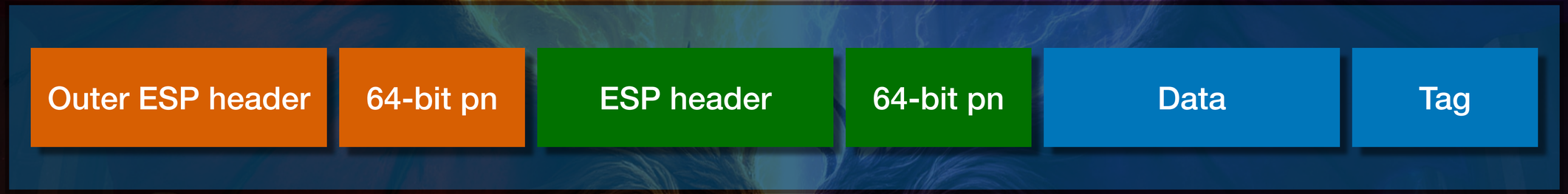
Worship of the packet

The transformation



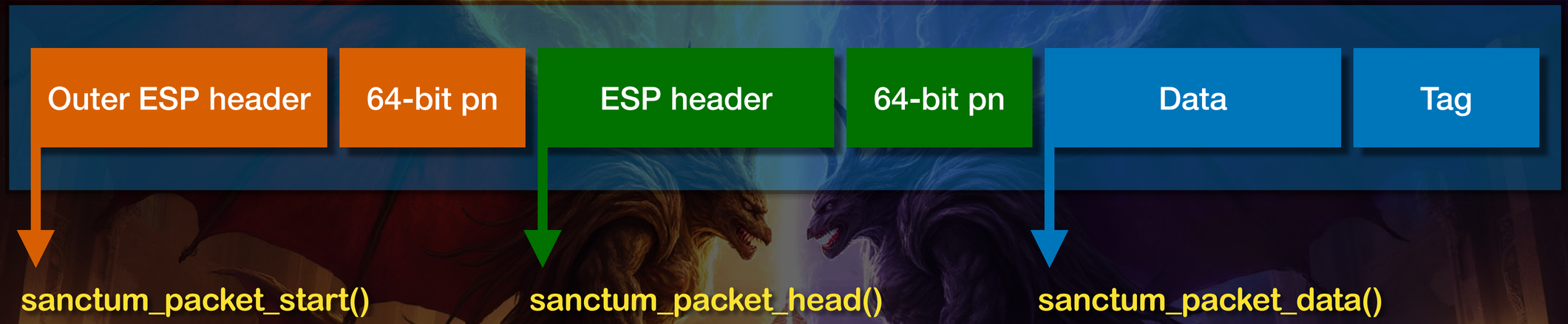
Worship of the packet

Non-Impetus



Worship of the packet

Non-Impetus



Worship of the packet

Non-Impetus

Worship of the packet

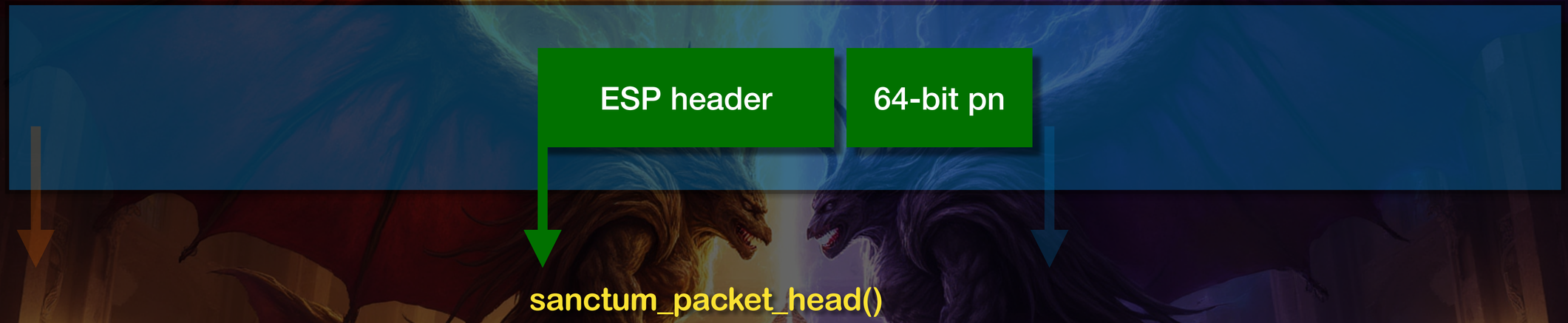
Non-Impetus

heaven-rx reads a packet

Worship of the packet

Non-Impetus

Bless fills in the ESP header and 64-bit packet number



Worship of the packet

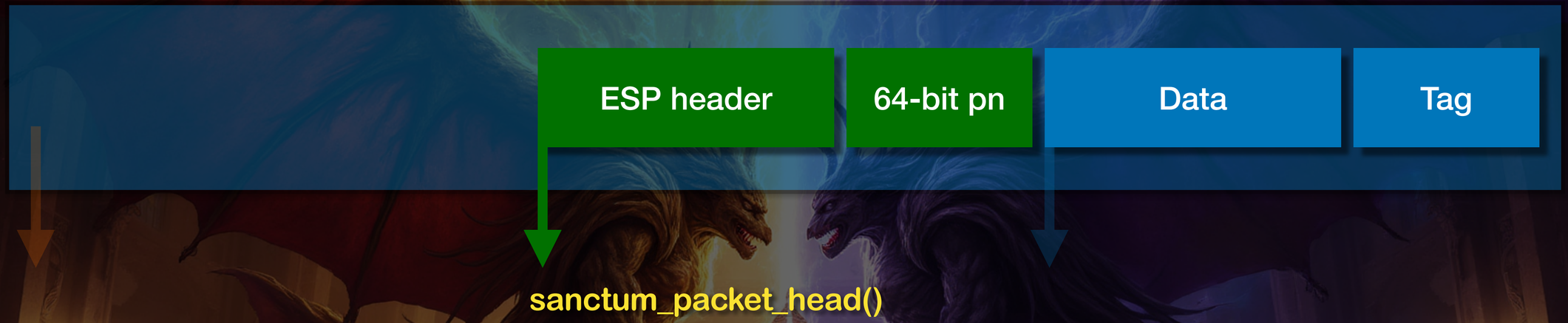
Non-Impetus

Bless encrypts and adds the tag.

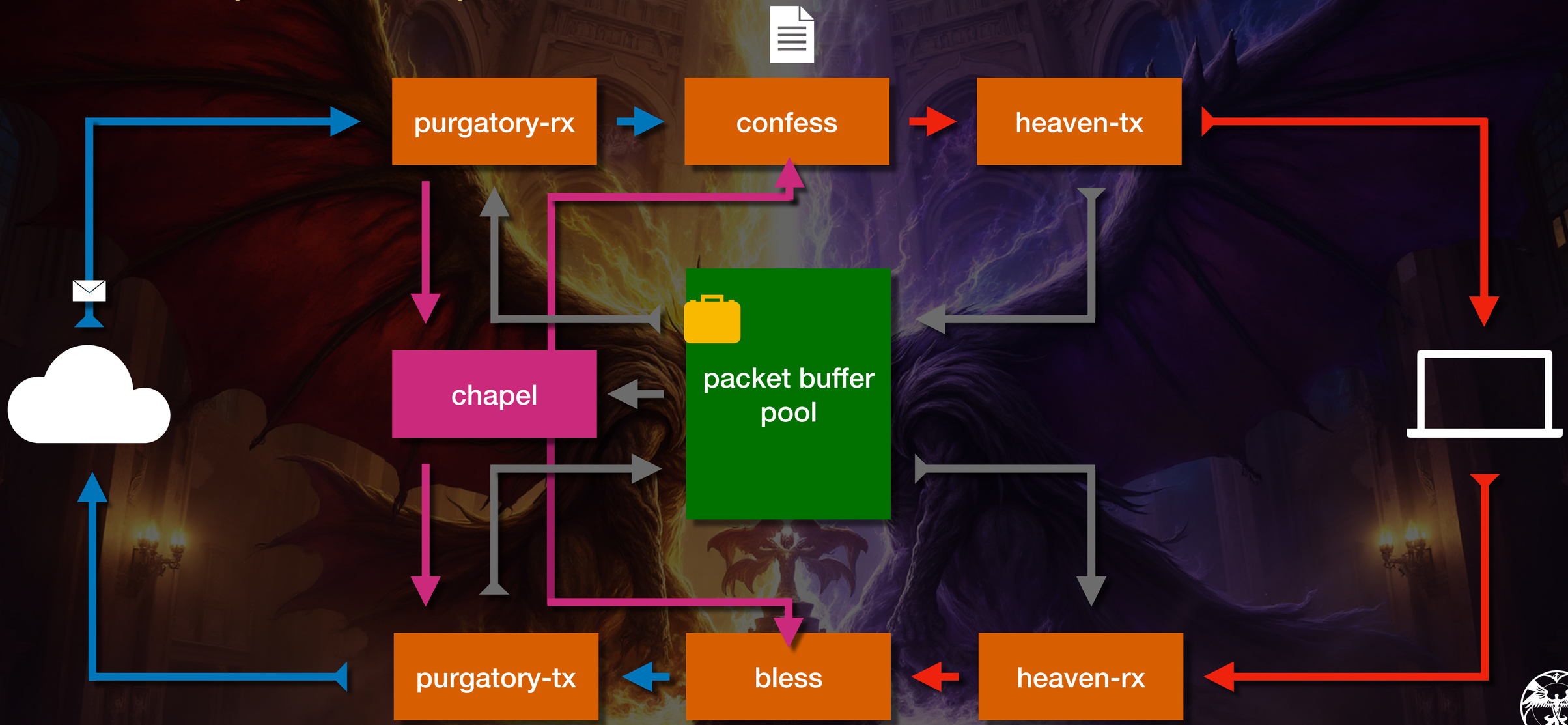
Worship of the packet

Non-Impetus

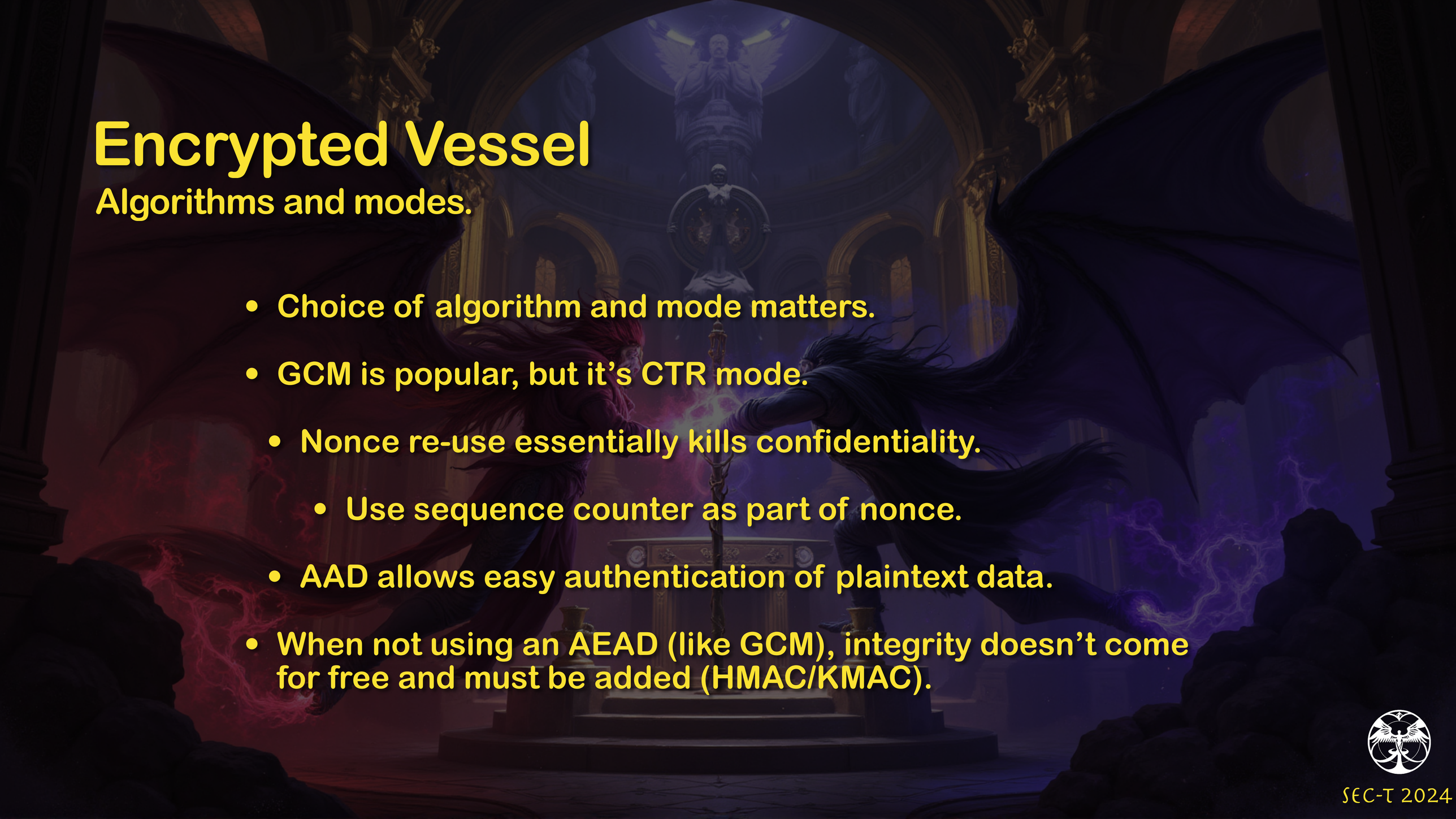
purgatory-tx sends



Worship of the packet



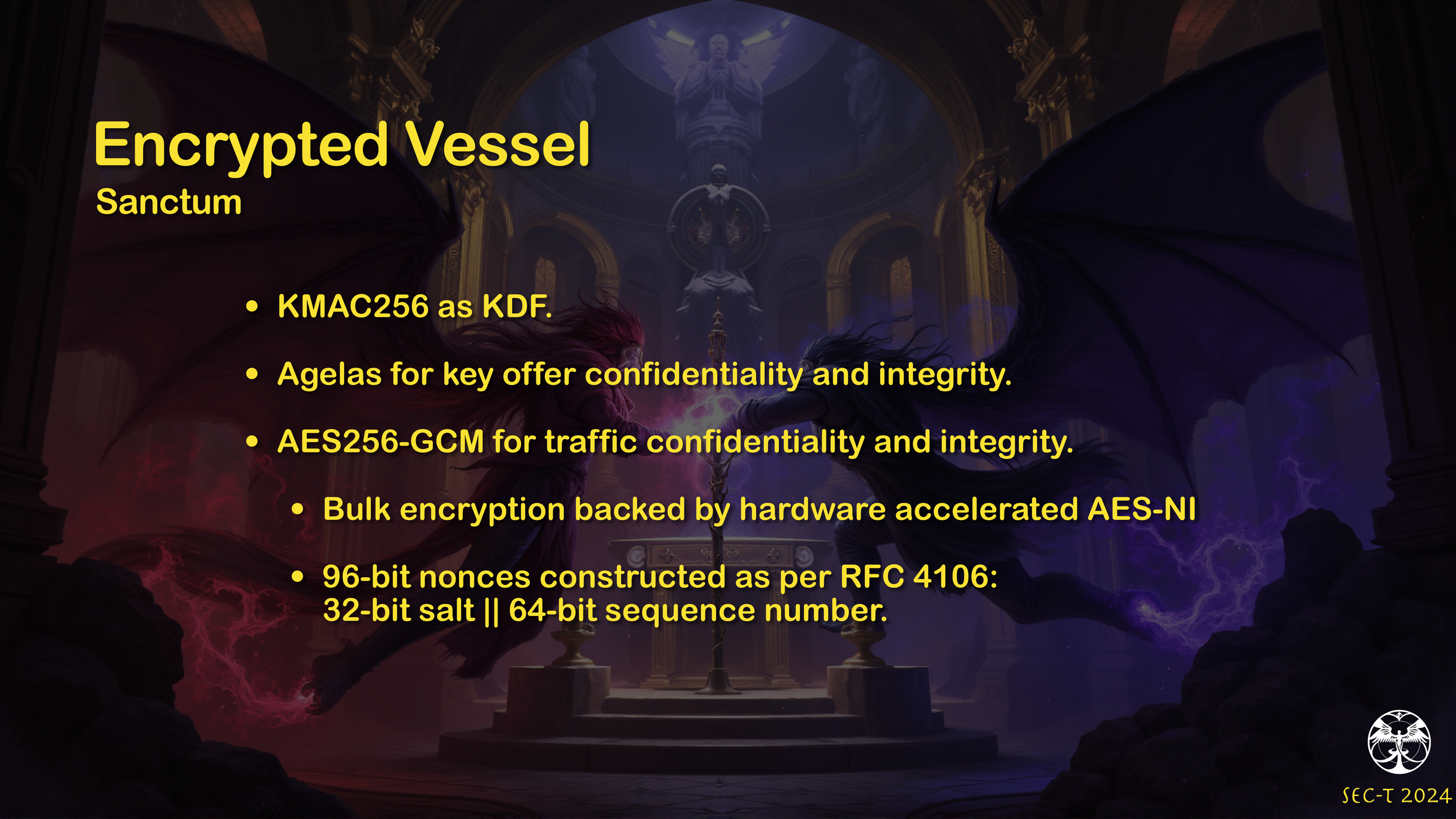
SEC-T 2024



Encrypted Vessel

Algorithms and modes.

- Choice of algorithm and mode matters.
- GCM is popular, but it’s CTR mode.
  - Nonce re-use essentially kills confidentiality.
    - Use sequence counter as part of nonce.
  - AAD allows easy authentication of plaintext data.
- When not using an AEAD (like GCM), integrity doesn’t come for free and must be added (HMAC/KMAC).



Encrypted Vessel

Sanctum

- KMAC256 as KDF.
- Agelas for key offer confidentiality and integrity.
- AES256-GCM for traffic confidentiality and integrity.
  - Bulk encryption backed by hardware accelerated AES-NI
  - 96-bit nonces constructed as per RFC 4106:
    32-bit salt || 64-bit sequence number.

Encrypted Vessel

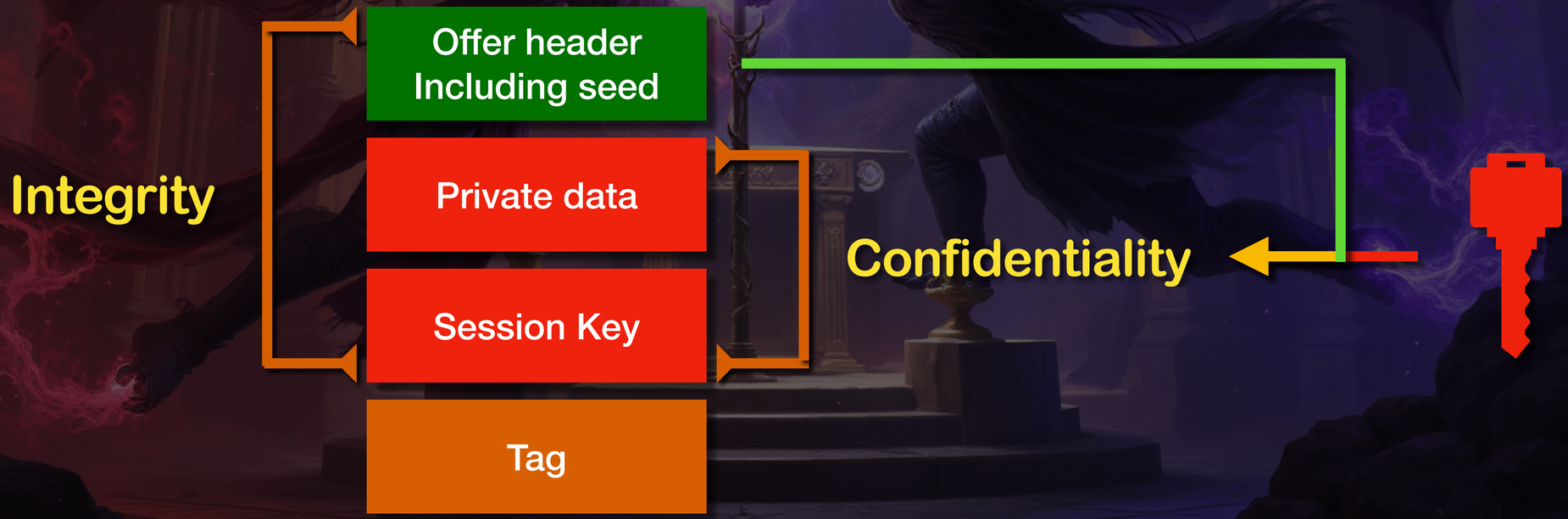
Key offering

- Security Associations (SA)
  - Maps SPI to a key.
- Two RX SA slots, one active, one pending.
- Seamless rollover from pending -> active.

Encrypted Vessel

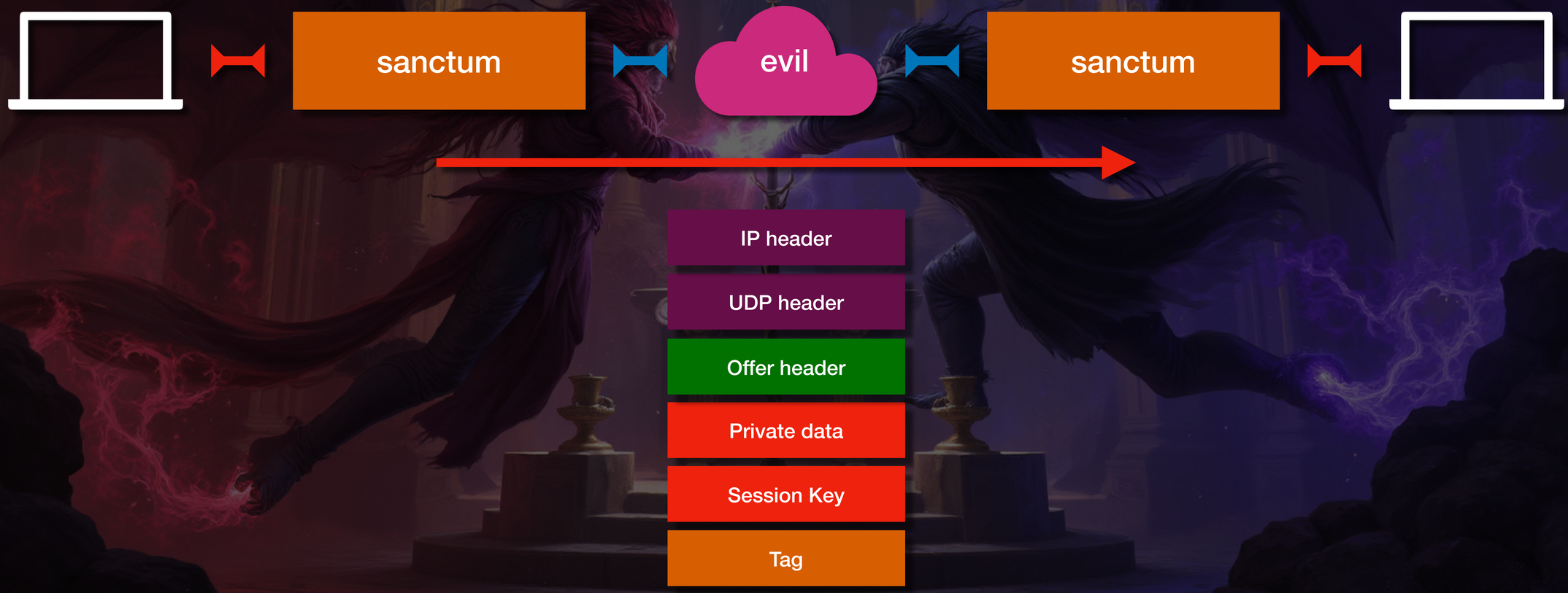
Key offering

```
derive_key(seed):
    ss = shared secret, 256-bit
    wk = KMAC256(ss, "SANCTUM.SACRAMENT.KDF", len(seed) || seed), 512-bit
    return wk
```



Encrypted Vessel

Key offering



Encrypted Vessel

Secure channel

- Anti-replay prevents the intercepting and resending of packets.
- This protects the integrity of the secure channel.
  - Each packet only arrives on the clear side once.
- A packet that is encrypted and has an integrity tag will always be valid under the correct key / nonce combination.
- Sequence numbers!
- Maintain a sliding window of received packets.

Protected Vessel

SEC-T 2024

Protected Vessel

Traffic Analysis

- Sanctum uses magic values in headers for:
  - Key exchanges.
  - Cathedral messages.
- Makes it easy to identify packets for Sanctum.
- Makes it easy to see you are using Sanctum.
- What if you didn’t want people to know?
  - Because opsec.

Protected Vessel

Traffic Analysis

- Traffic Analysis Protection fixes this.
- Adds an outer ESP encapsulation layer.
- Hides all the magic values and makes the entire ESP stream indistinguishable from any other ESP stream.
  - Fakes SA rollover.
  - Correct sequence counting.

Prayer Of Closing

SEC-T 2024

Prayer of Closing

- Sanctum
  - Fully privilege separated.
  - Highly performant.
  - Small footprint.
- You now know a little more on how its internals work.
  - Shared memory, ring queues, use of atomics.
- You now know better how to approach privilege separation.

Prayer of Closing

- Building this type of software is hard.
- Requires many years of experience to get right.
- So keep hacking and keep writing C code.
  - You _are_ needed.
- Shout out to my Protokol0x41 hackers.

Prayer of Closing

- Didn’t talk about the cathedral works.
- Didn’t talk about how one-way tunnels work.
- Didn’t talk about how processes wake each other up with futex().
  - Or why the -rx and -tx split made things a lot easier.
- And much, much more.
- I will be here all conference.


Thanks for listening
Go forth and hack