WAMOS 2018:

Thorsten Knoll
thorsten.knoll@hs-rm.de

Adapting Kerckhoffs principle:

A path from cryptography to open source hardware

The path:

- The idea
- Kerckhoffs principle (KP)
- Attacks and KP
- Mitigations and KP
- The need for Open Source Hardware (OSH)
- Conclusion: Adapting KP

The idea

New:	Sidechannel attacks on mainstream CPUs (~2000)
Brandnew:	Meltdown, Spectre, BranchScope, TLBleed, Spectre-NG, … (2018)
Very old:	Kerckhoffs principle (1883)

New: Sidechannel attacks on mainstream CPUs (~2000)

Brandnew: Meltdown, Spectre, BranchScope, TLBleed, Spectre-NG, … (2018)

Very old: Kerckhoffs principle (1883)

Paper:
Adapting Kerckhoffs principle

Kerckhoffs principle (KP)

1883 - Auguste Kerckhoffs in “La cryptographie militaire”:

Design principles for (military) cipher systems:

1. The system must be practically, if not mathematically, indecipherable.
2. It should not require secrecy, and it should not be a problem if it falls into enemy hands.
3. It must be possible to communicate and remember the key without using written notes, and correspondents must be able to change or modify it at will.
4. It must be applicable to telegraph communications.
5. It must be portable, and should not require several persons to handle or operate.
6. Lastly, given the circumstances in which it is to be used, the system must be easy to use and should not be stressful to use or require its users to know and comply with a long list of rules.

Translation from french to english: https://en.wikipedia.org/wiki/Kerckhoffs%27s_principle

1883 - Auguste Kerckhoffs in “La cryptographie militaire”:

Design principles for (military) cipher systems:

1. The system must be practically, if not mathematically, indecipherable.
2. It should not require secrecy, and it should not be a problem if it falls into enemy hands.
3. It must be possible to communicate and remember the key without using written notes, and correspondents must be able to change or modify it at will.
4. It must be applicable to telegraph communications.
5. It must be portable, and should not require several persons to handle or operate.
6. Lastly, given the circumstances in which it is to be used, the system must be easy to use and should not be stressful to use or require its users to know and comply with a long list of rules.

Translation from french to english: https://en.wikipedia.org/wiki/Kerckhoffs%27s_principle

1883 - Auguste Kerckhoffs in “La cryptographie militaire”:

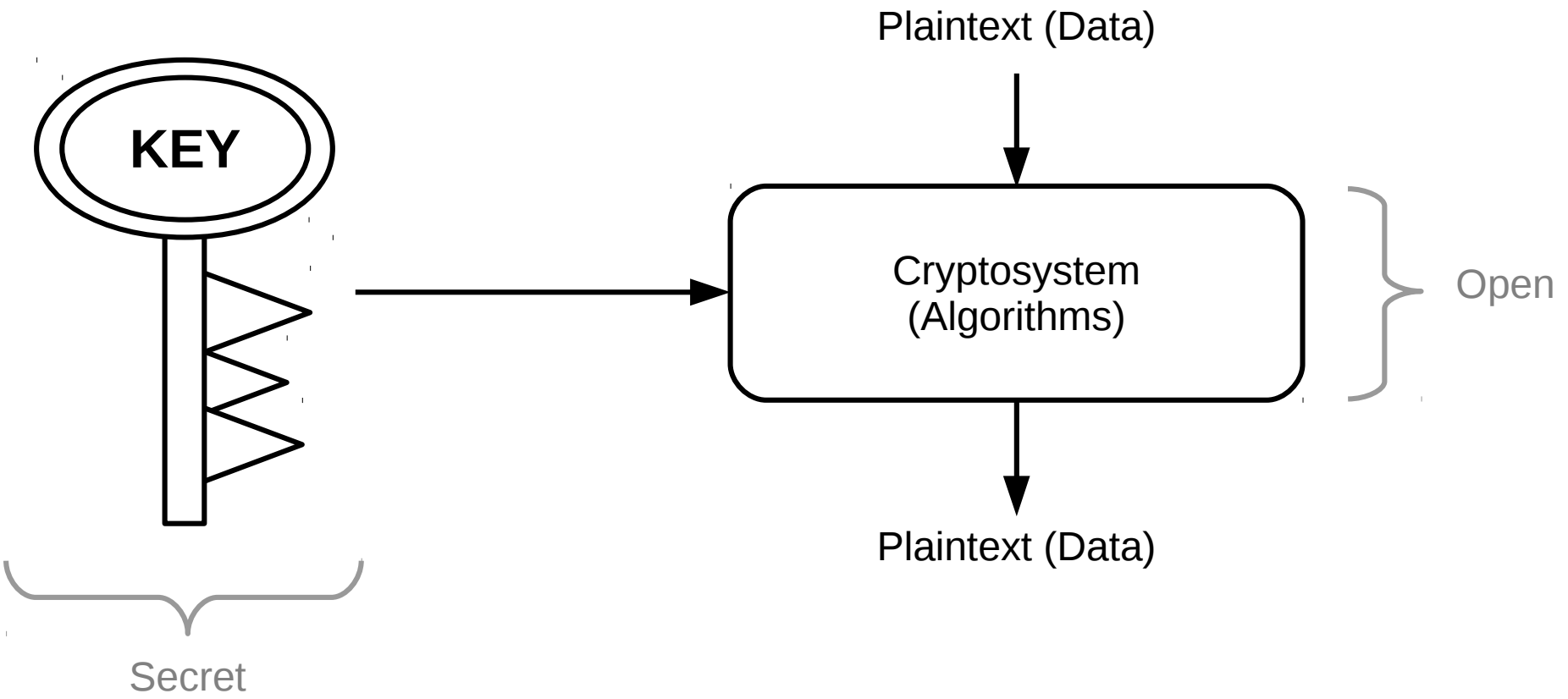
Design principles for (military) cipher systems:

2. It should not require secrecy, and it should not be a problem if it falls into enemy hands.

Broader interpretation:

A cryptosystem’s security must solely depend on keeping the keys secret, not the algorithms.

Now known as **“Kerckhoffs principle” (KP)**


Plaintext (Data)
KEY
Cryptosystem
(Algorithms)
Open
Plaintext (Data)
Secret

1949 - Claude E. Shannon:

Reformulation of KP:

The enemy knows the system.

Design principle:

A cryptosystem should be designed as if the enemy would already know the system.

Now known as **“Shannon’s maxim” (SM)**

2002 - Bruce Schneier:

- Generalization for all secure systems.

- More secrets to be kept → More fragile.
- Less secrets to be kept → More robust.

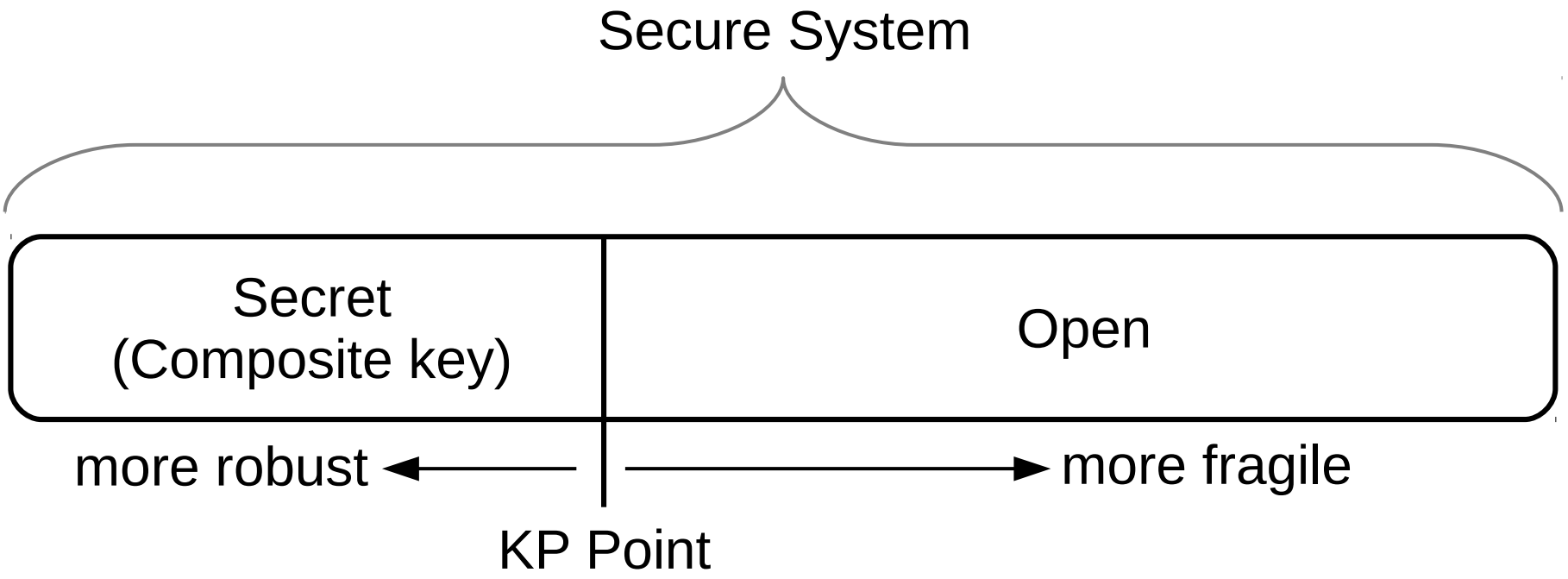
- If algorithms, protocols, implementations have to be kept secret
  → They’re part of the (composite) key.

- How easily and costly is the replacing of compromised keyparts?

- Openness enables reviews and evaluation, implied a community.

- Design principle:
  Minimize the number of secrets in your security system.

2018 – Thorsten Knoll:

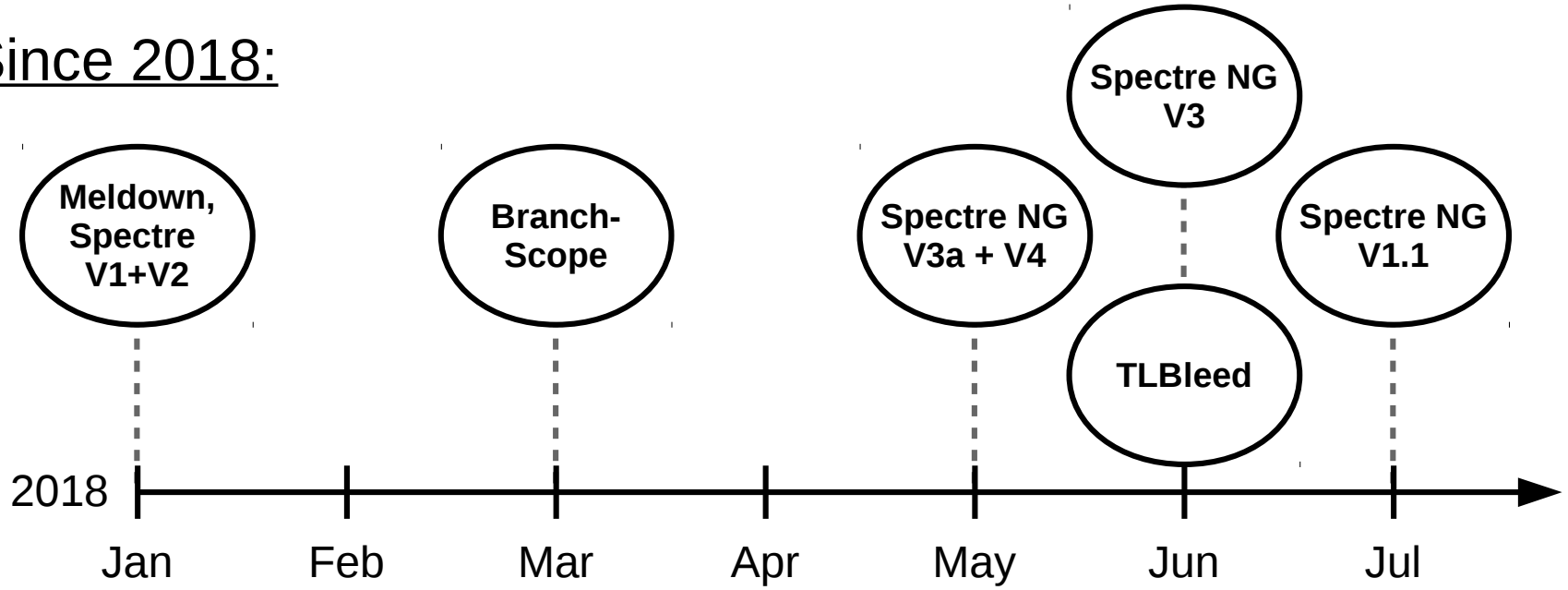


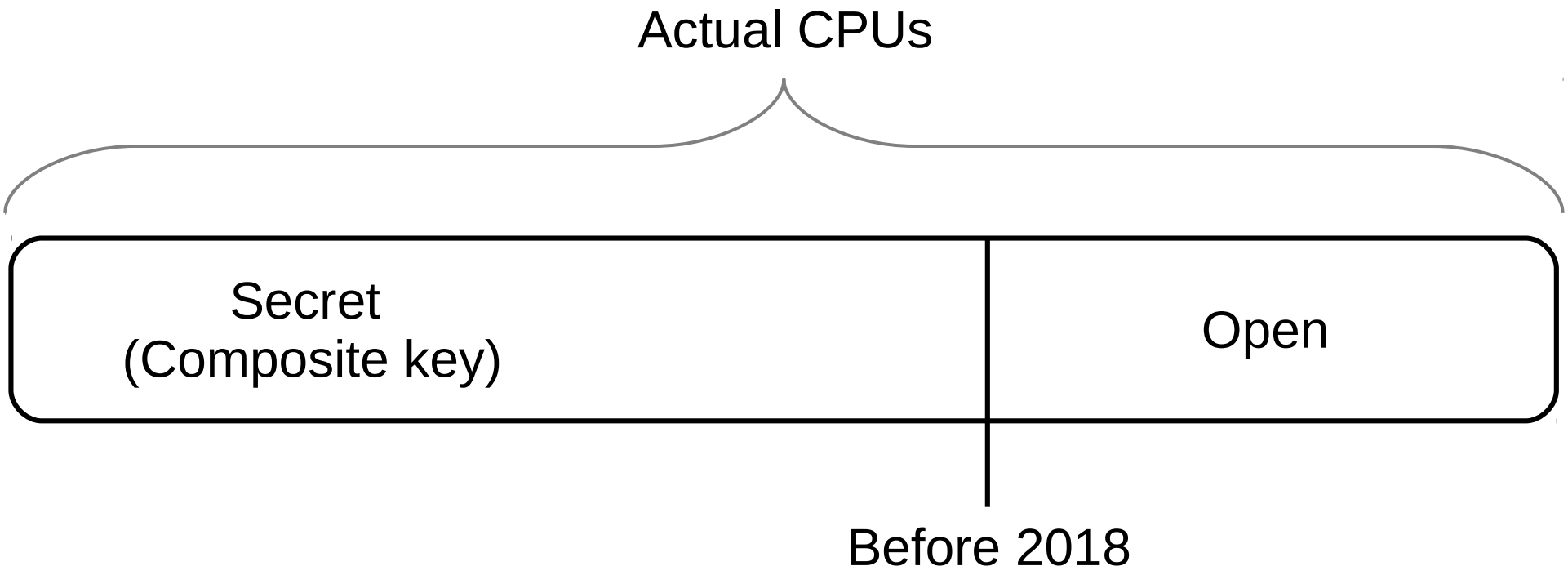
Attacks and KP

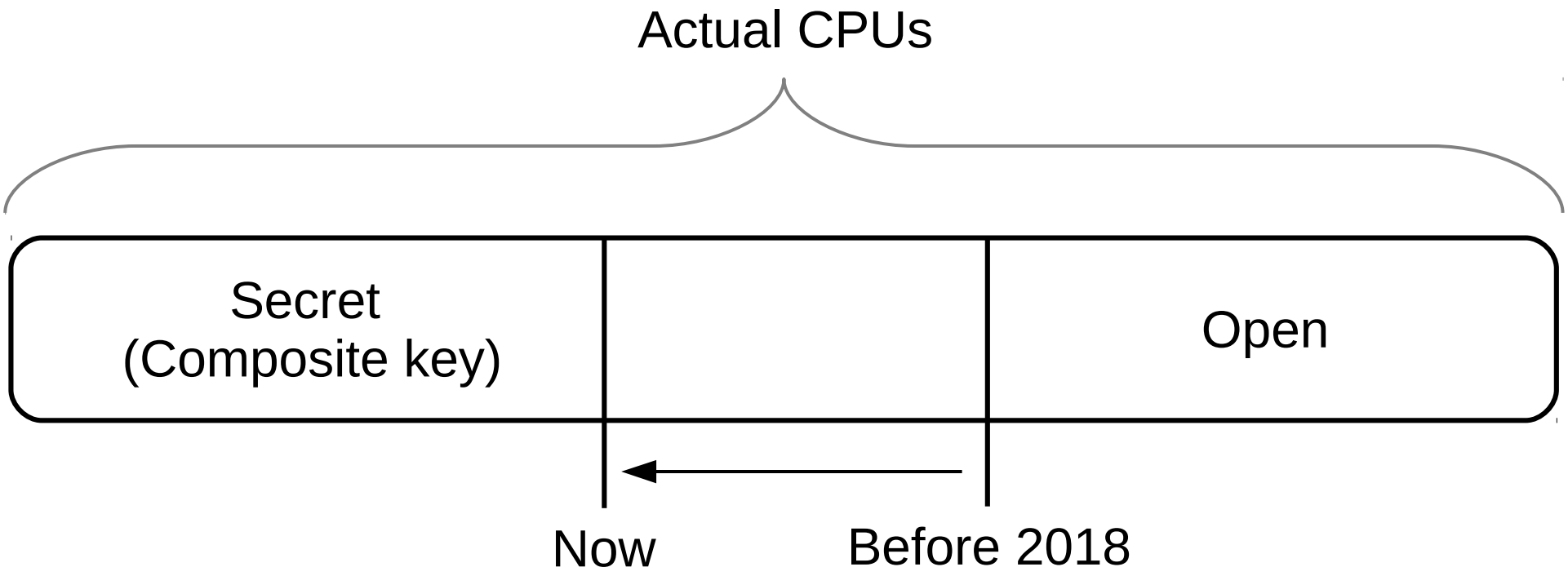
Since 2003:

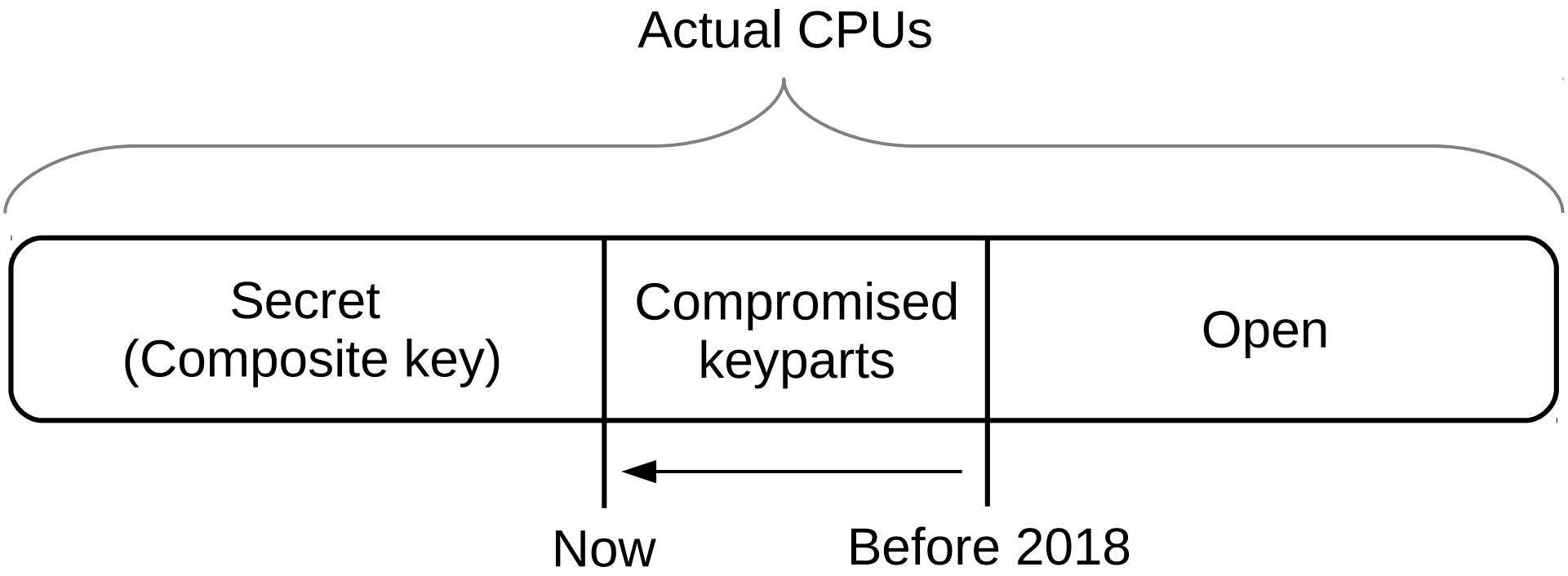
- Remote attacks are practical
- Prime and Probe
- Predicting secret keys via branch prediction
- Cache games
- Flush and reload
- Evict and reload

Since 2018:




Actual CPUs
Secret
(Composite key)
Open
Before 2018

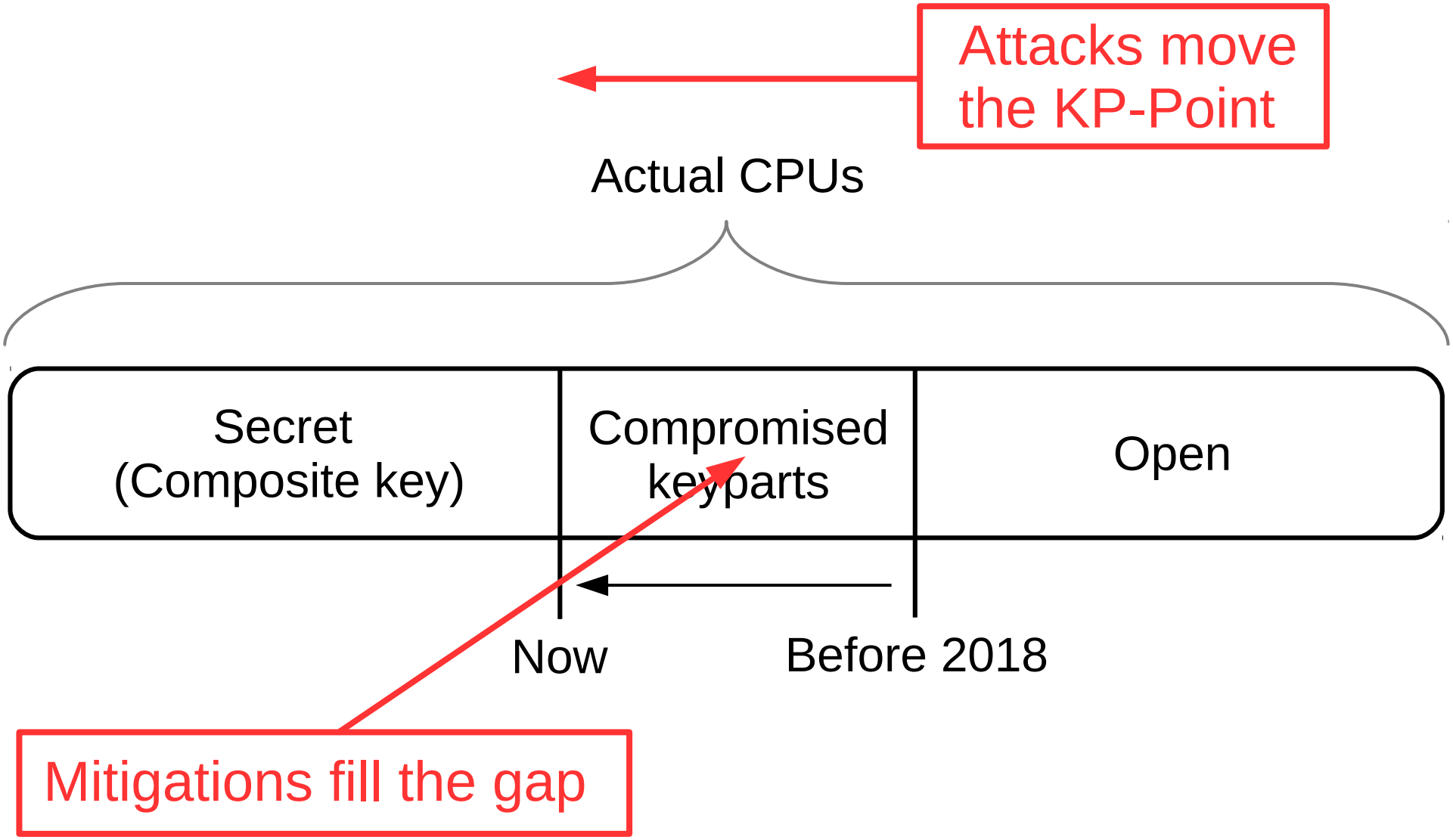

Actual CPUs
Secret
(Composite key)
Open
Now
Before 2018


Actual CPUs
Secret
(Composite key)
Compromised
keyparts
Open
Now
Before 2018

Mitigations and KP

Mitigation strategies:

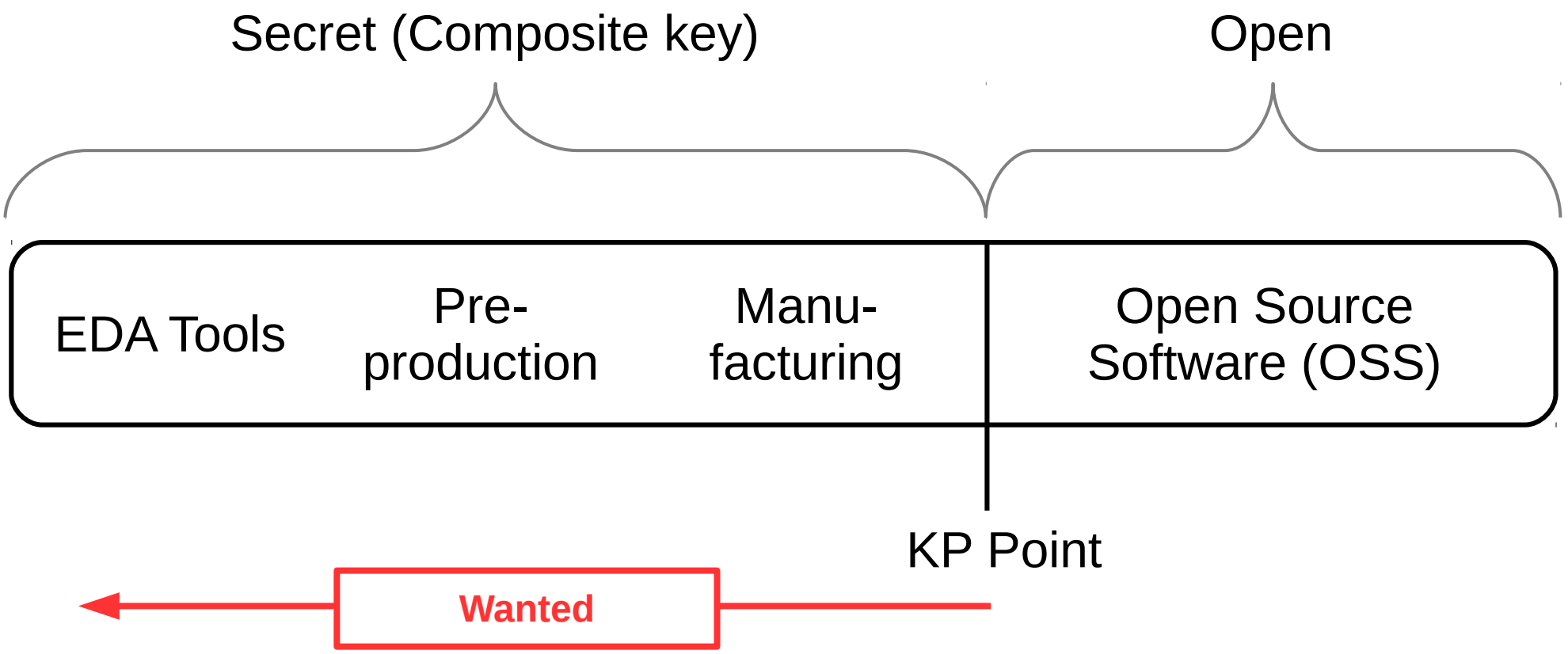
- In Software → Mostly open.
- Seems like a hot-fix for a bigger problem (in hardware).
- A lot of work in a small timeframe.
- Billions of affected devices sold.
- Not every attack vector is mitigatable in software.
- Call for openness arises: New ISA’s including timings.


Attacks move the KP-Point
Actual CPUs
Secret (Composite key)
Compromised keyparts
Open
Now
Before 2018
Mitigations fill the gap

The need for Open Source Hardware (OSH)

2018 – Sovereignty in IT:

- 60 pages, describing the actual state.
- Tons of examples: Hardware-Trojans, -Backdoors, -Killswitches.
- Defines 13 actionpoints to start with.
- Security can’t be added to hardware by software.
- If software is secure, the attacks target a level deeper (hardware).
- Design and production of silicon is closed by now and should be opened to gain back sovereignty.


Secret (Composite key)
Open
EDA Tools
Pre-
production
Manu-
facturing
Open Source
Software (OSS)
KP Point
Wanted

Conclusion: Adapting Kerckhoffs principle

Conclusions:

- Attacks move the KP-Point by research, not by design.
- Mitigations fill the gap to secure systems again.
- Mitigations are necessary. Billions of devices sold.
- Kerckhoffs principle is adaptable for measuring the situation.
- More Openness doesn’t solve the actual situation, but in long term it could relax the short timeframe situation.
- A strong community is needed, for review and evaluation.
- Open Source Hardware is on the rise (RISC-V).

Conclusion: Adapting Kerckhoffs principle

Think ahead:

- NIST has gone the way of finding new cryptography algorithm standards through open, public competitions.
- Maybe we'll see such a competition for PBUs, TLBs and PHTs in near future?

Think ahead:

- NIST has gone the way of finding new cryptography algorithm standards through open, public competitions.
- Maybe we'll see such a competition for PBUs, TLBs and PHTs in near future?
- Why would anyone build an i7 in Open Source? It might not even be possible.
- Instead putting the focus on massive parallelisation. Think about some thousand RISC-V on a single die. That would even be possible with FPGA prototyping in academia.

The path:

- All of this is a path to more robust, secure and open systems.
- Going the presented path might take decades.

The path:

- All of this is a path to more robust, secure and open systems.
- Going the presented path might take decades.

But we'll never arrive there,
when we don't start moving!

The path:

- All of this is a path to more robust, secure and open systems.
- Going the presented path might take decades.

But we'll never arrive there,
when we don't start moving!

Thank you.