Removable USB media security

Feedback from implementing a USB decontamination station

THCon 2024

April 4th & 5th, 2024

TOULOUSE HACKING CONVENTION

Innover. Simplifier. Partager.

Who are we?

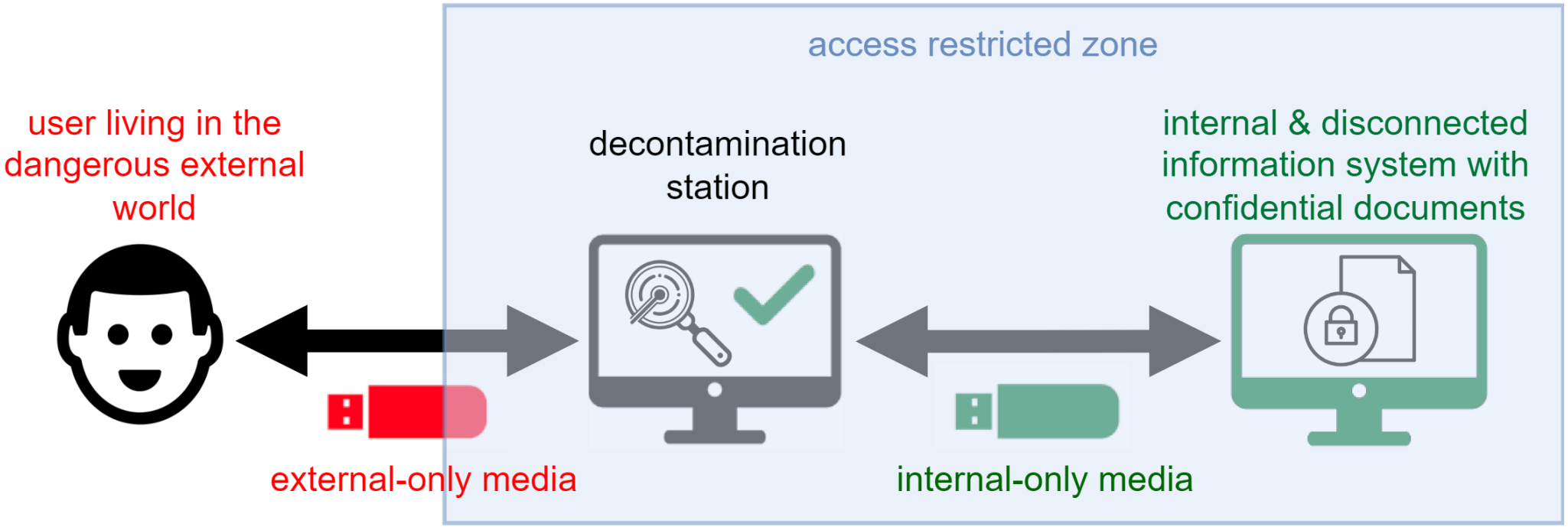
GUILLON Benoît
Cybersecurity engineer
Viveris Technologies

BELMON Valentin
Cybersecurity engineer
Viveris Technologies

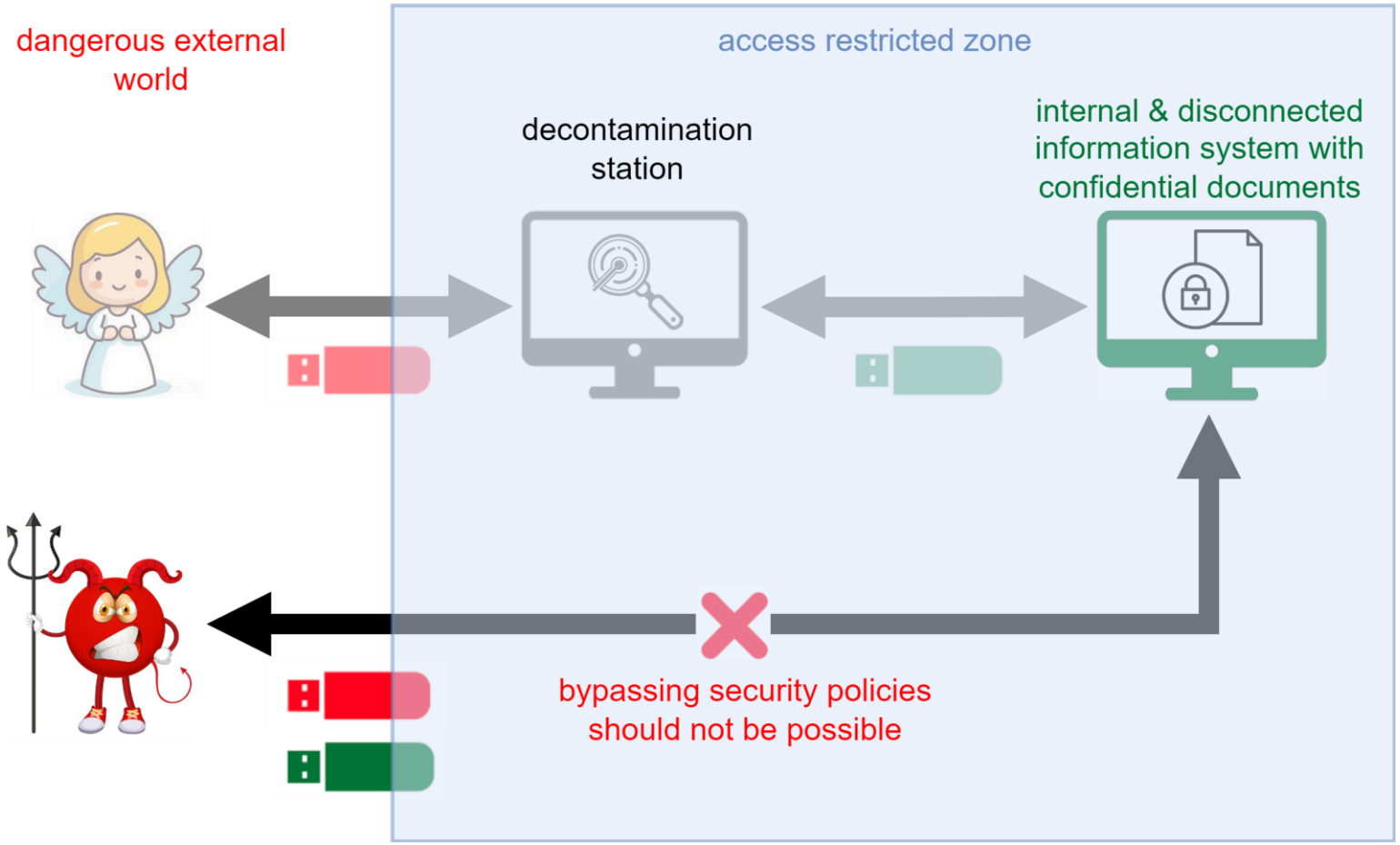
A little bit of context…

Viveris implements a USB decontamination station

- Secure exchanges between an internal isolated information system and the rest of the world
- To ensure data security, users have to scan USB removable media when importing/exporting documents



Identified risk: data infiltration/exfiltration (station bypass)



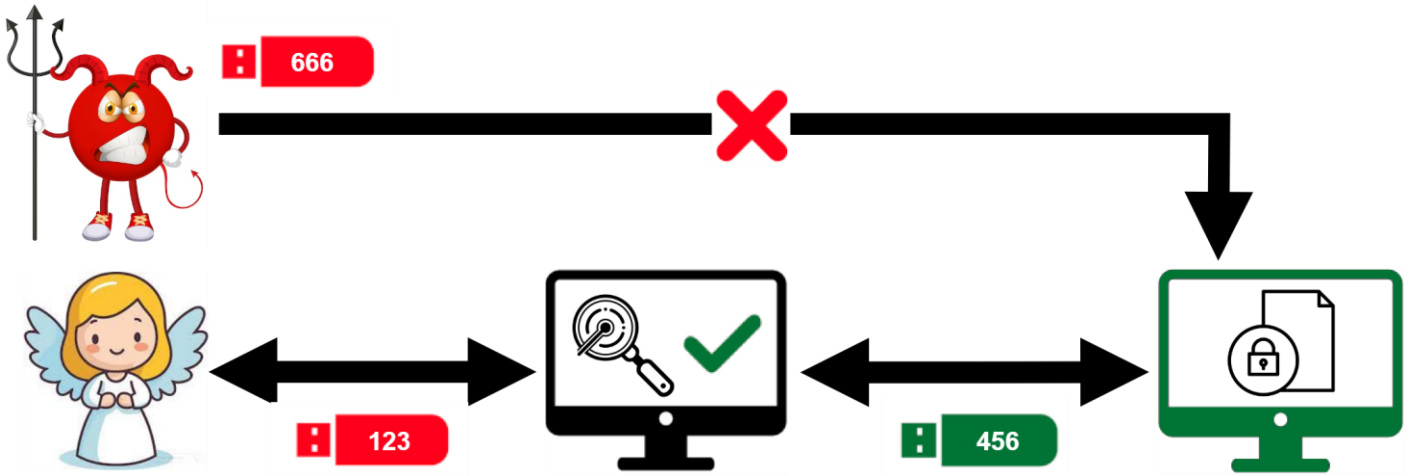
- How to forbid people to bypass the station and infiltrate/exfiltrate confidential documents?
- Existing mitigations
  - Enclosure walls
  - Door access control
  - Security training aka « don’t do it! »
- Won’t stop a team member or a motivated attacker

Evaluation	Infiltration	**0/5**
	Exfiltration	**0/5**

VIVERIS

Solution #1: access control with USB serial ID

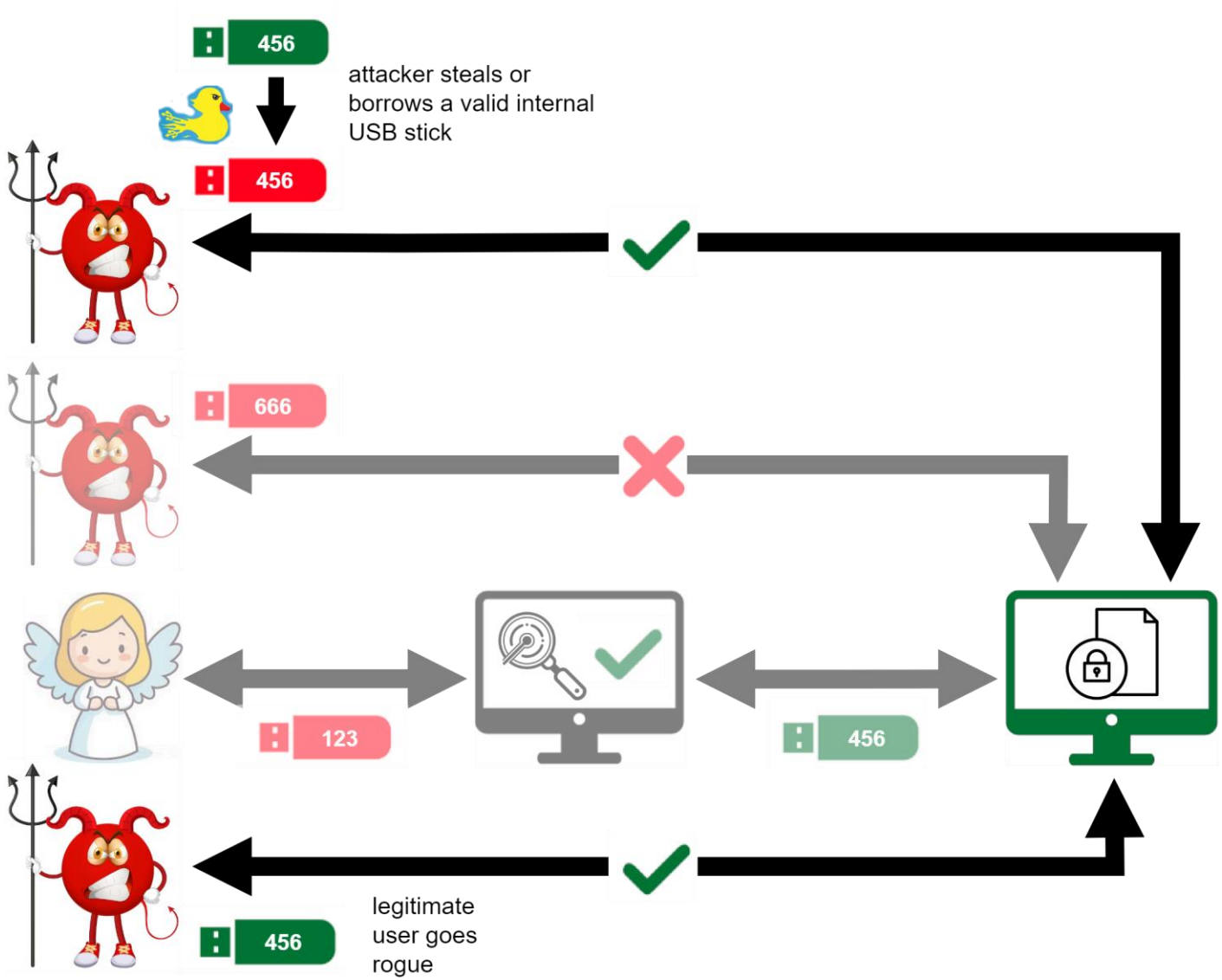
- Allowlist of internal USB serial Ids
  - Easy to implement
  - Stop direct usage of any USB stick by attackers



Evaluation	Infiltration	?
	Exfiltration	?

VIVERIS

Solution #1: access control with USB serial ID

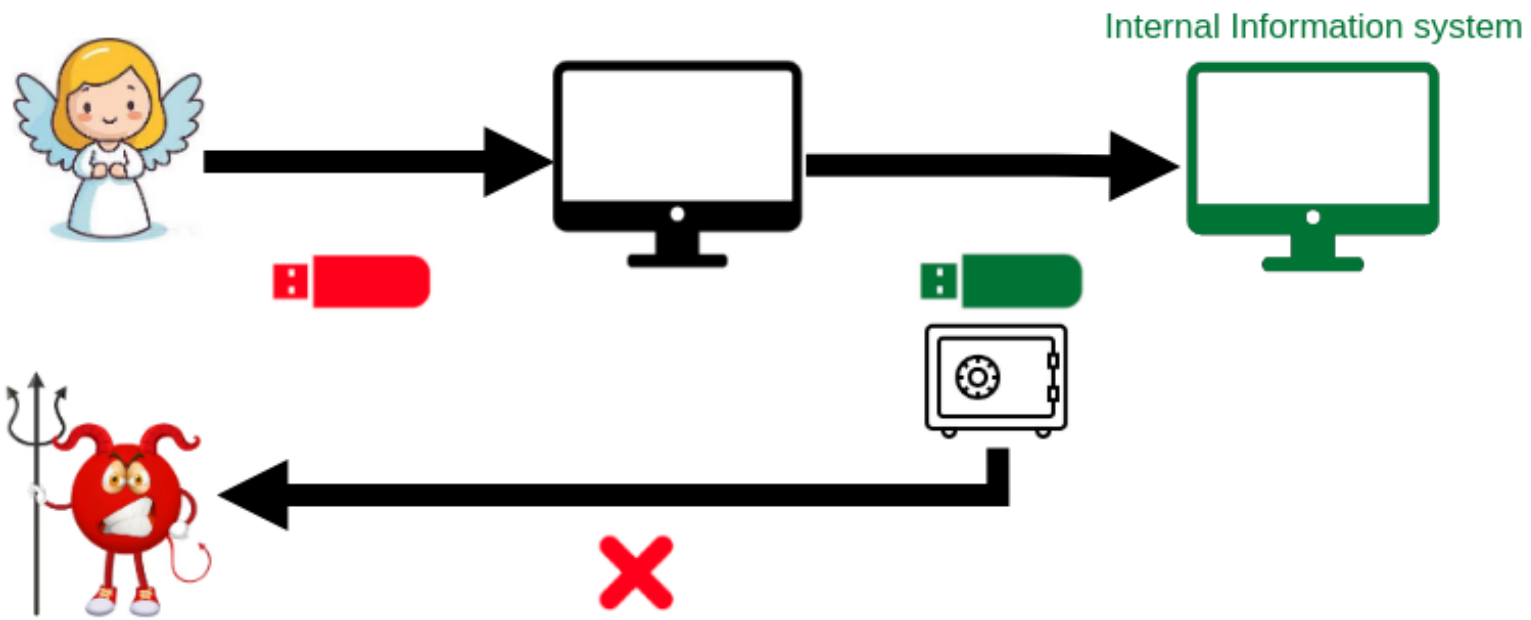


- Allowlist of internal USB serial Ids
  - Easy to implement
  - Stop direct usage of any USB stick by attackers
- Caveats
  - Borrowing internal USB stick and copying USB serial ID is easy
  - Stealing internal USB stick too
  - Impossible to forbid internal USB stick everywhere in the world

Evaluation	Infiltration	1/5
	Exfiltration	1/5

VIVERIS

Solution #2: physical protection of allowed USB sticks

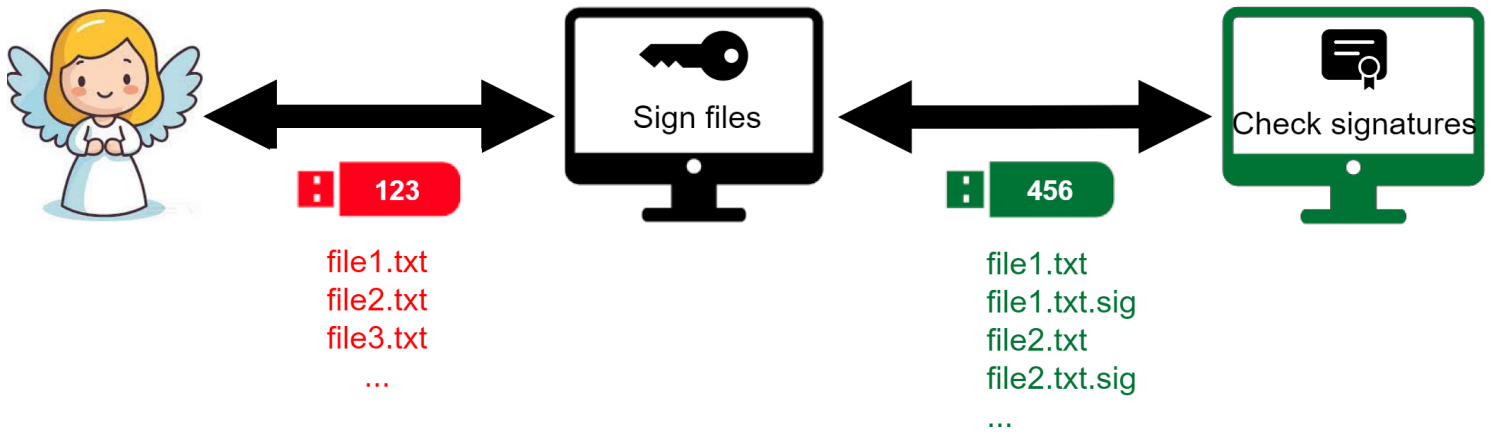


- Keep USB sticks locked in a safe
  - Increase difficulty to steal/borrow
- Caveats
  - Human and not technical solution
  - Make life of legitimate users harder

Evaluation	Infiltration	0/5
	Exfiltration	0/5

VIVERIS

Solution #3: cryptographic proof of decontamination



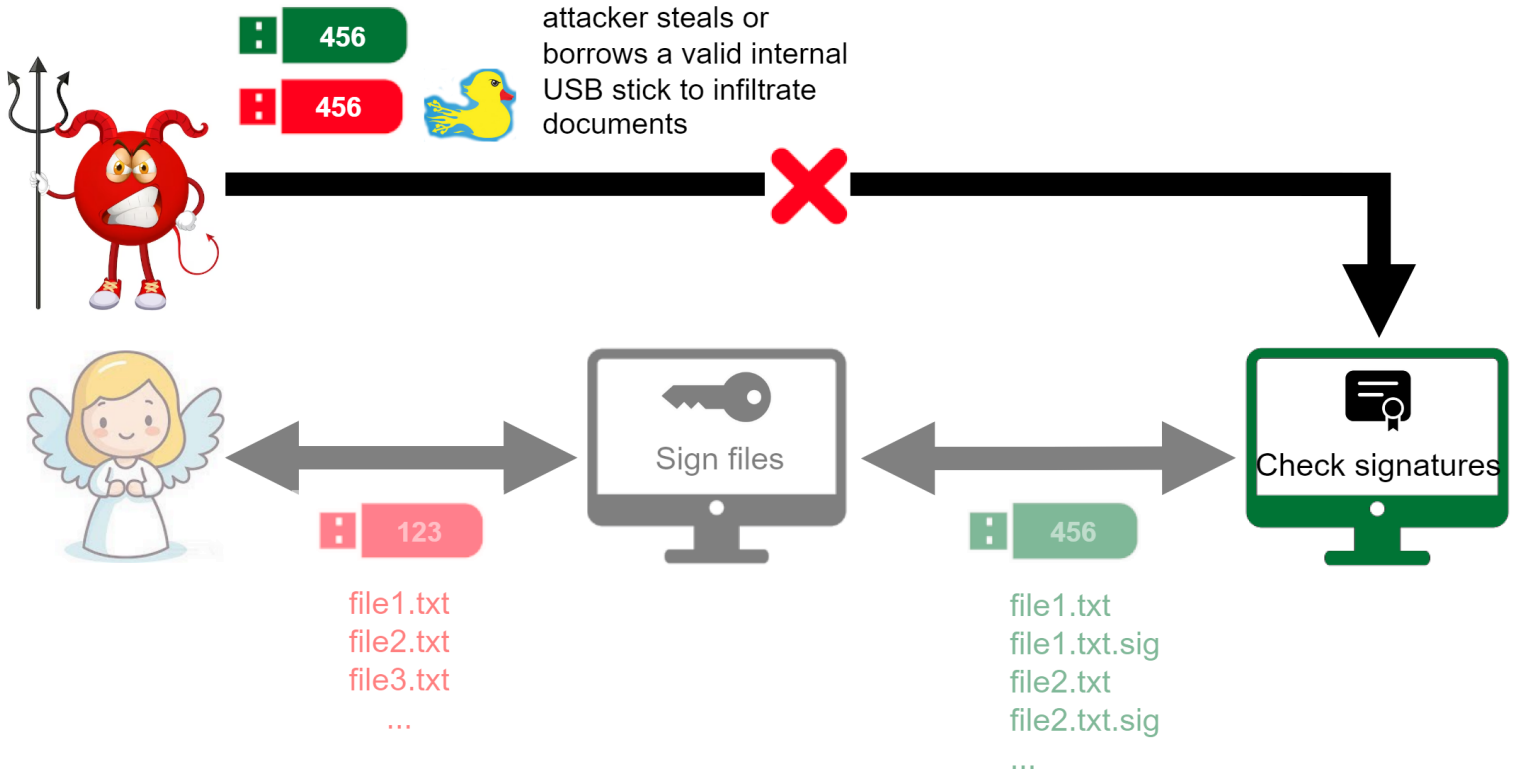
- Cryptographic proof of decontamination
  - Decontamination station signs healthy files
  - Workstations block access to unsigned or badly-signed files

Evaluation		
	Infiltration	?
	Exfiltration	?

The Windows agent is work in progress by the ANSSI for Keysas https://keysas.fr/

VIVERIS

Solution #3: cryptographic proof of decontamination

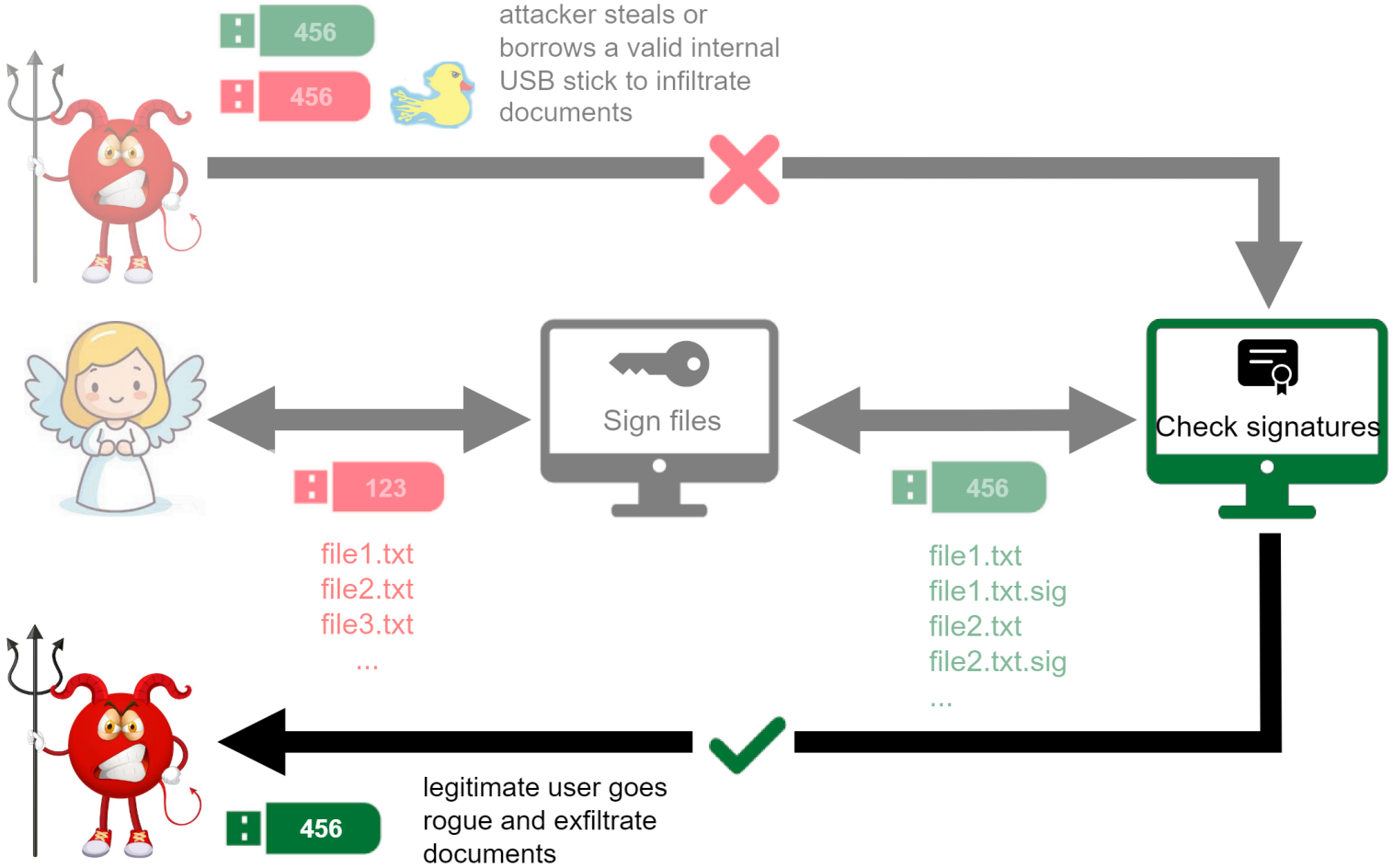


- Cryptographic proof of decontamination
  - Decontamination station signs healthy files
  - Workstations block access to unsigned or badly-signed files

Evaluation		
	Infiltration	5/5
	Exfiltration	?

The Windows agent is work in progress by the ANSSI for Keysas https://keysas.fr/

Solution #3: cryptographic proof of decontamination



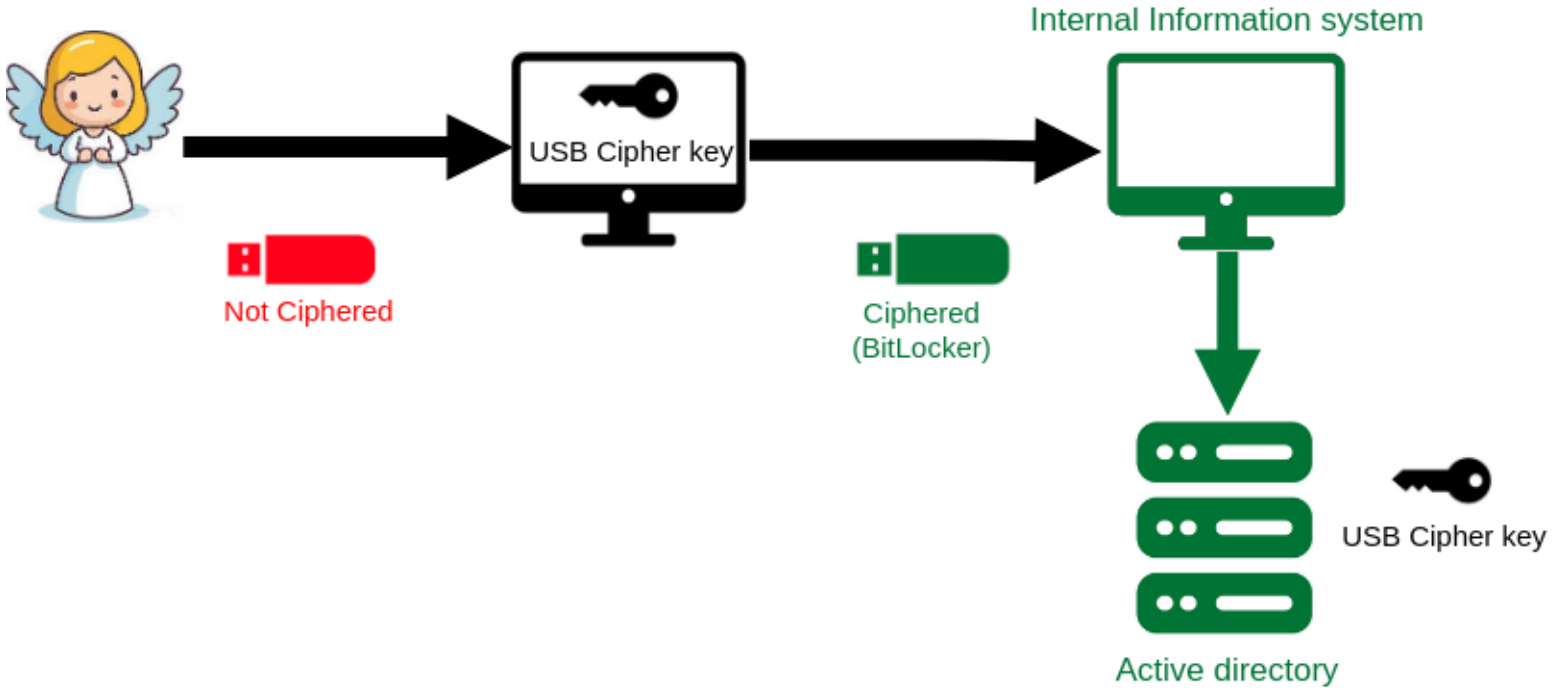
- Cryptographic proof of decontamination
  - Decontamination station signs healthy files
  - Workstations block access to unsigned or badly-signed files
- Caveats
  - Not natively supported by Windows (ad-hoc development required)
  - Exfiltration still possible

Evaluation	Infiltration	5/5
	Exfiltration	1/5

The Windows agent is work in progress by the ANSSI for Keysas https://keysas.fr/

VIVERIS

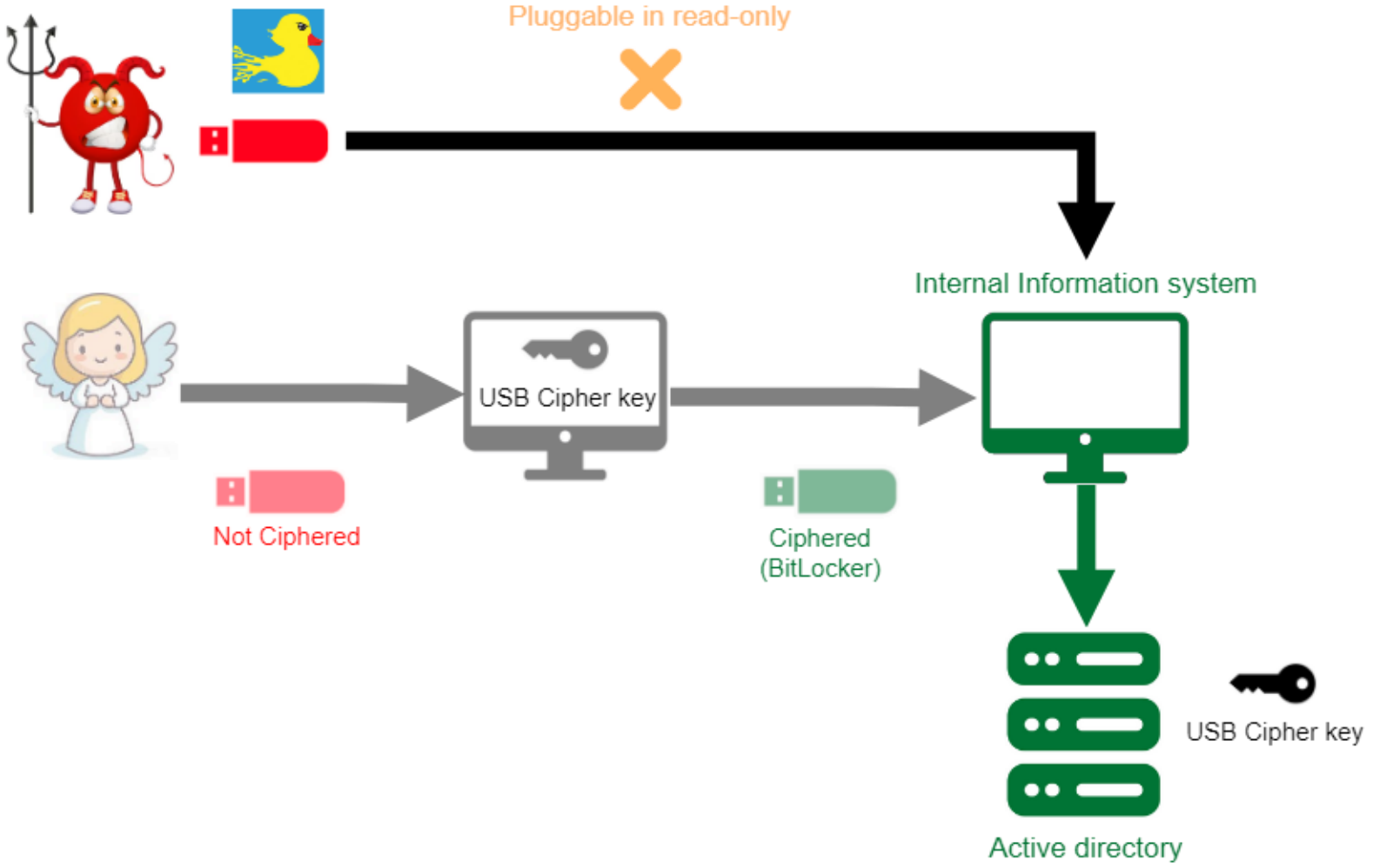
Solution #4: encrypted USB sticks



- Protect content by encryption
  - Bitlocker encrypted partition
  - Decontamination station & workstations know encryption key
  - Workstations reject clear-text and unknown-key encrypted USB sticks
- Caveats
  - Requires AD to secure key storage

Evaluation	Infiltration	?
	Exfiltration	?

Solution #4: encrypted USB sticks

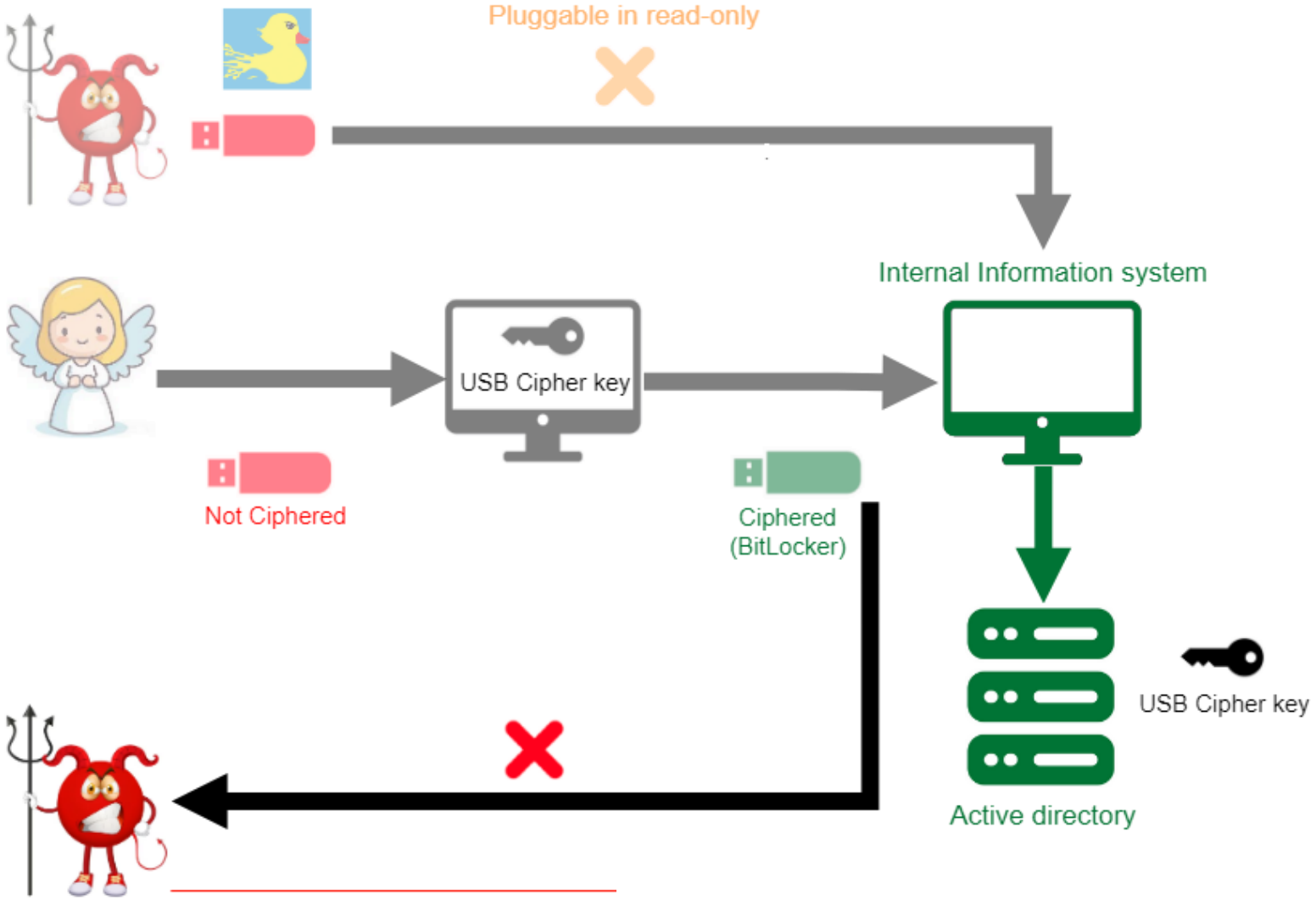


- Protect content by encryption
  - Bitlocker encrypted partition
  - Decontamination station & workstations know encryption key
  - Workstations reject clear-text and unknown-key encrypted USB sticks
- Caveats
  - Requires AD to secure key storage
  - Workstations still accepts unknown-key USB sticks, but in read-only

Evaluation	Infiltration	1/5
	Exfiltration	?

VIVERIS

Solution #4: encrypted USB sticks

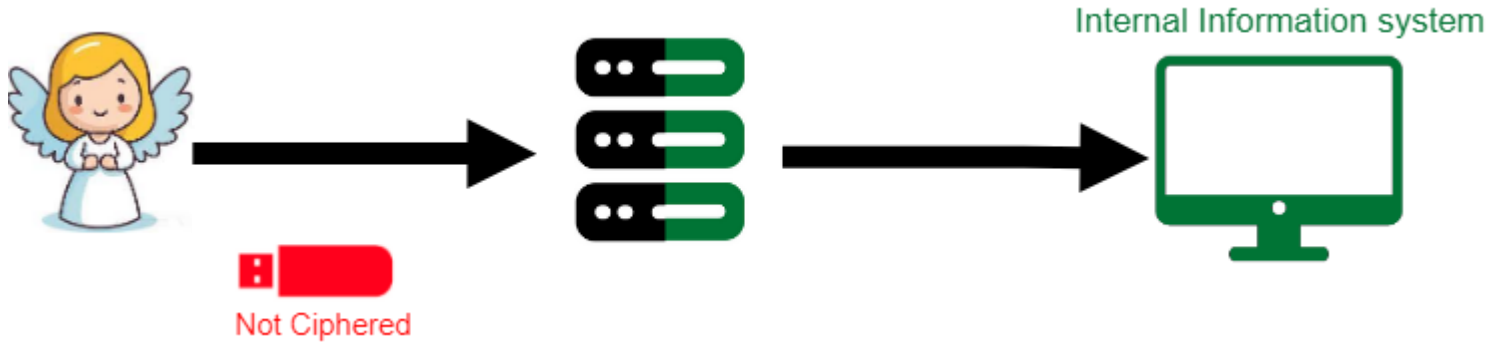


- Protect content by encryption
  - Bitlocker encrypted partition
  - Decontamination station & workstations know encryption key
  - Workstations reject clear-text and unknown-key encrypted USB sticks
- Caveats
  - Requires AD to secure key storage
  - Workstations still accepts unknown-key USB sticks, but in read-only

Evaluation	Infiltration	**1/5**
	Exfiltration	**4/5**

VIVERIS

Solution #5: replace internal USB sticks by network

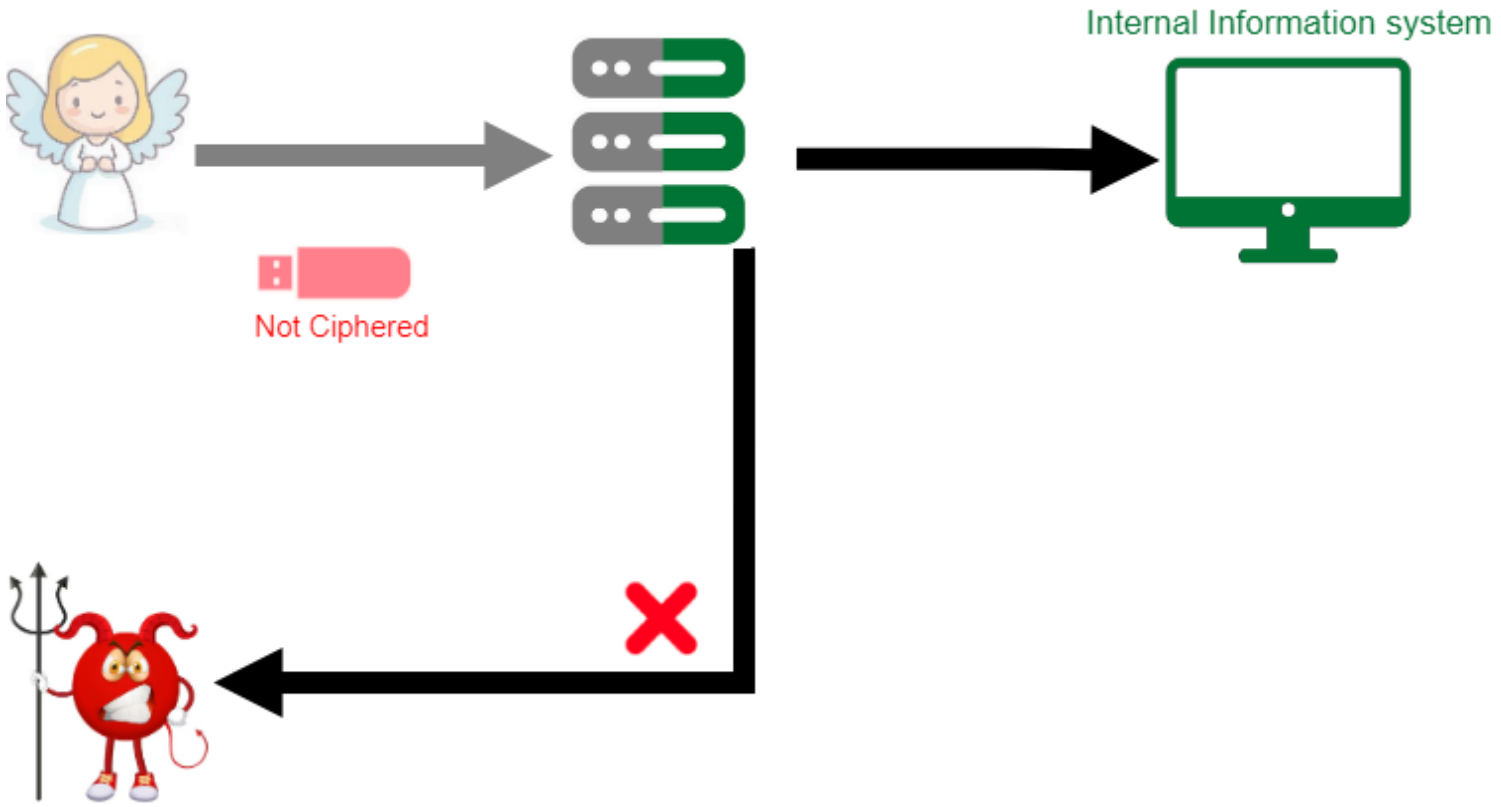


- Avoid internal USB sticks entirely
  - Use internal network instead
  - Fully disable USB storage on internal workstations
  - Make life of legitimate users easier
- Caveats
  - Large architectural change
  - Need segregation on internal network

Evaluation	Infiltration	?
	Exfiltration	?

VIVERIS

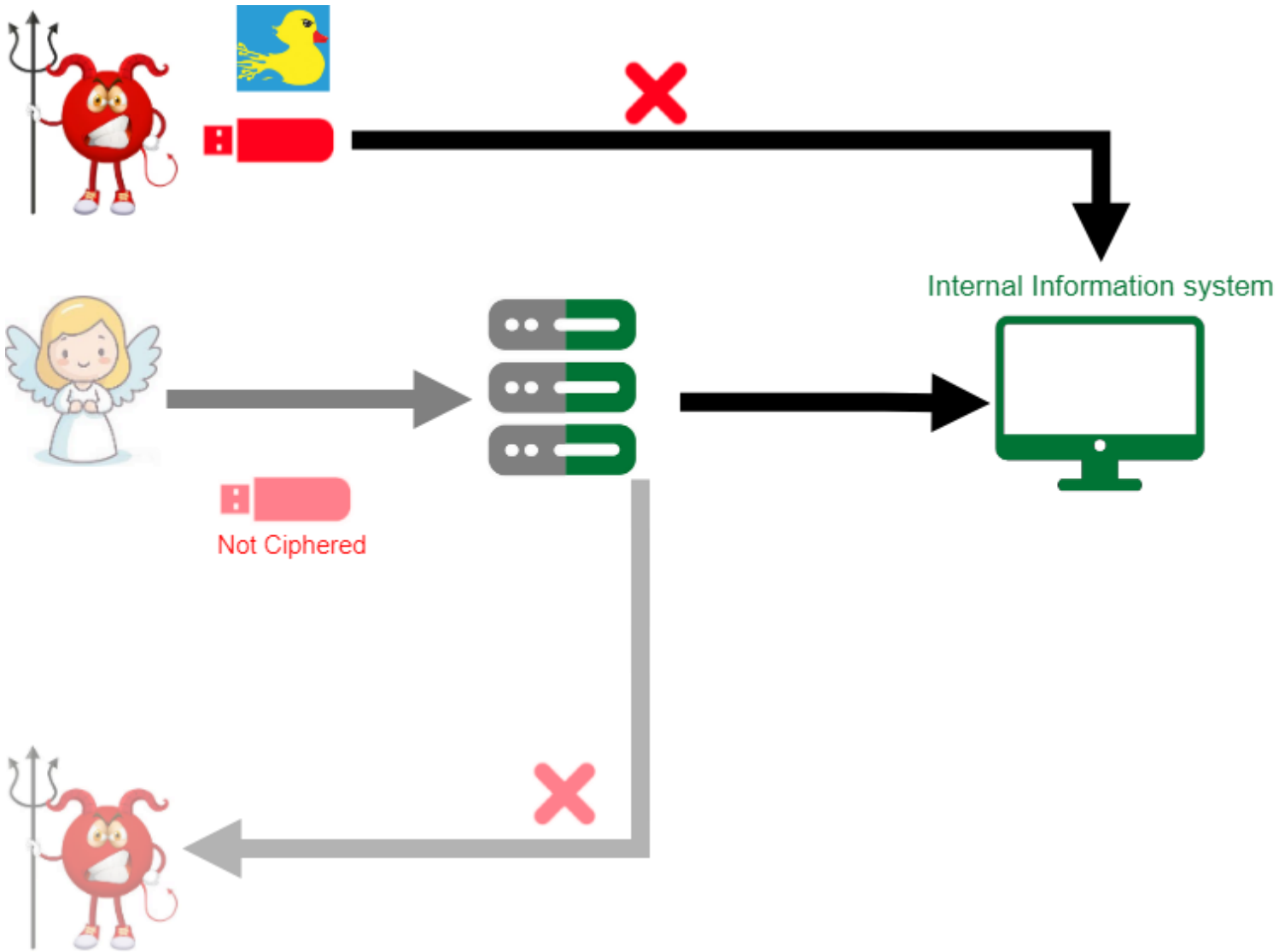
Solution #5: replace internal USB sticks by network



- Avoid internal USB sticks entirely
  - Use internal network instead
  - Fully disable USB storage on internal workstations
  - Make life of legitimate users easier
- Caveats
  - Large architectural change
  - Need segregation on internal network

Evaluation	Infiltration	?
	Exfiltration	5/5

Solution #5: replace internal USB sticks by network



- Avoid internal USB sticks entirely
  - Use internal network instead
  - Fully disable USB storage on internal workstations
  - Make life of legitimate users easier
- Caveats
  - Large architectural change
  - Need segregation on internal network

Evaluation	Infiltration	5/5
	Exfiltration	5/5

VIVERIS

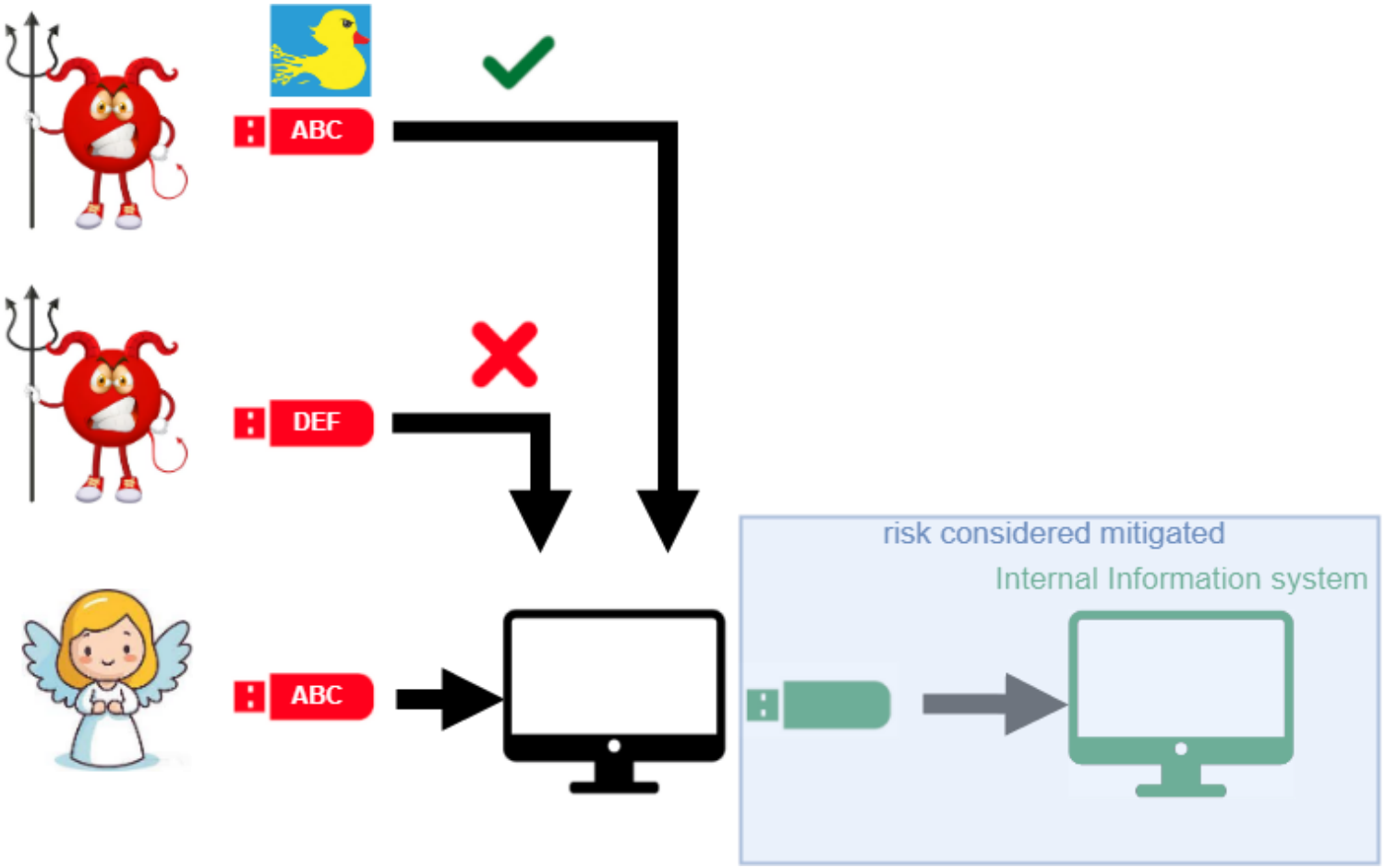
Conclusion (part one)

- Multiple solutions identified
  - Pros / cons discussed
  - Can be cumulated together to cover all cases

Evaluation	#1 (serial ID)	#2 (safe)	#3 (signatures)	#4 (encryption)	#5 (network)
Infiltration	1/5	0/5	5/5	1/5	5/5
Exfiltration	1/5	0/5	1/5	4/5	5/5

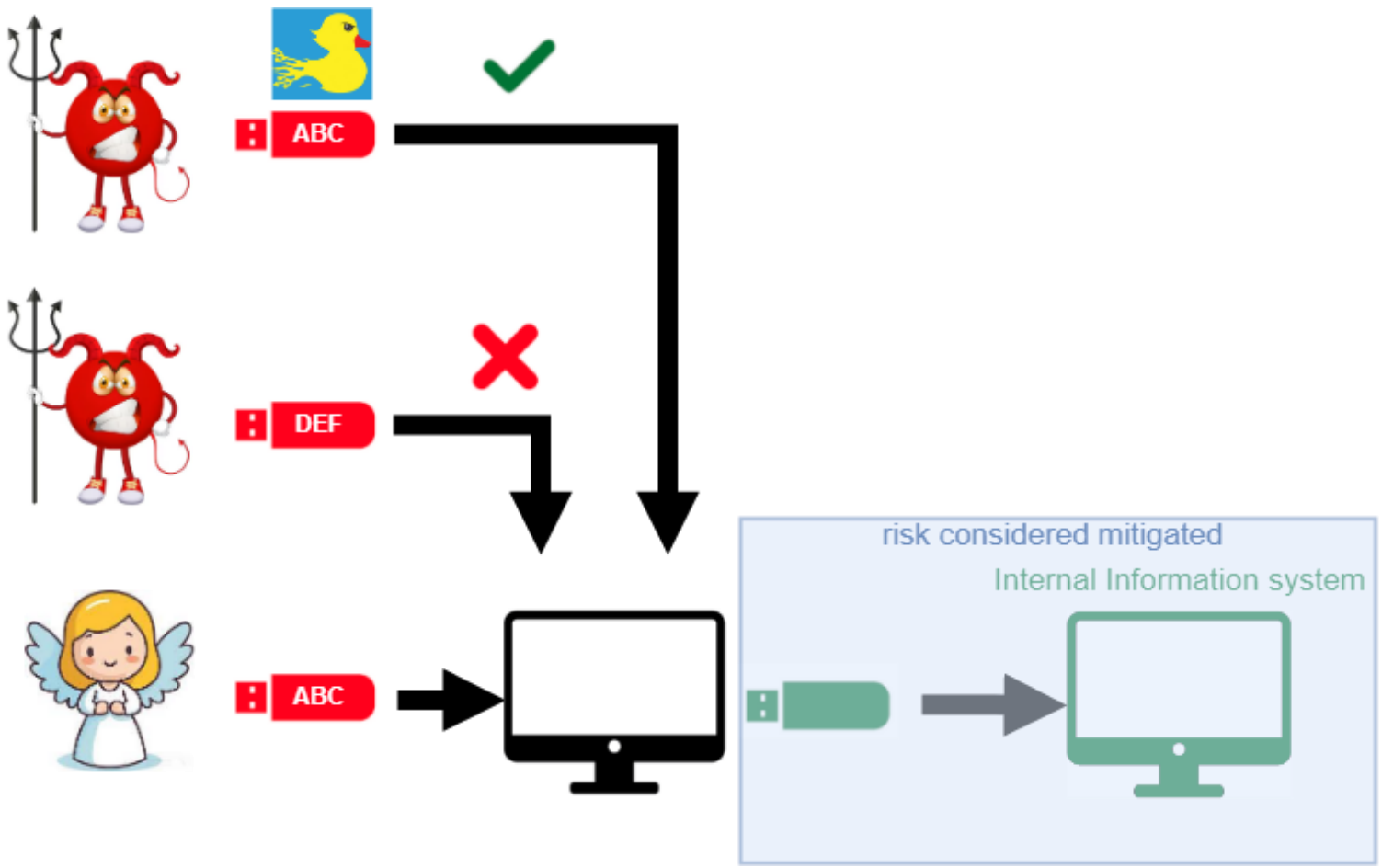
- As of now, Viveris retained 2 solutions
  - Solution #1: USB serial ID access control
  - Solution #4: encrypted sticks were implemented
- Solution #3 could be added when the Keysas implementation is ready
- … but solution #5 (network) is the preferred way to go!

Identified risk: attack on station with filesystem vulnerabilities



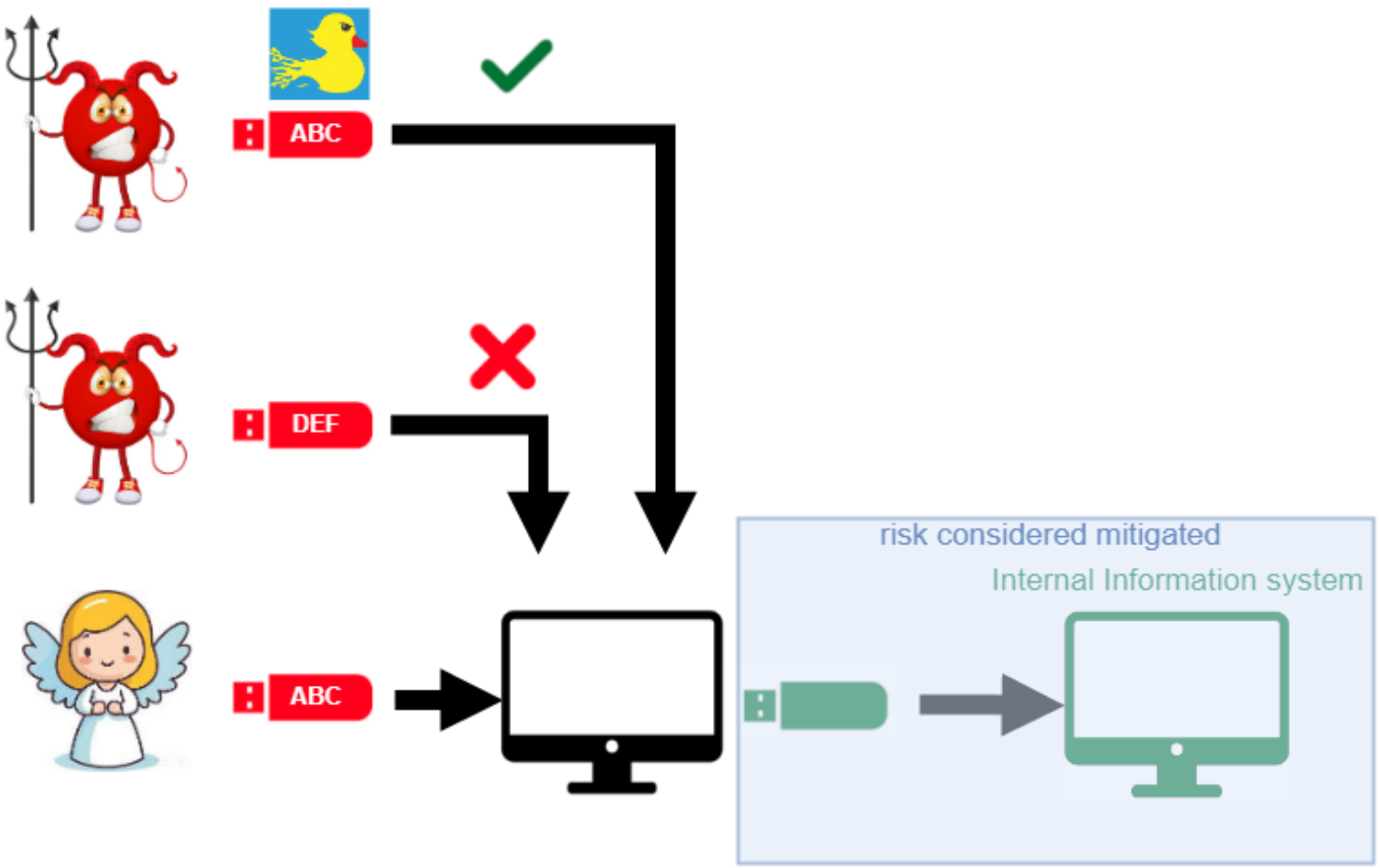
- Risk of document infiltration / exfiltration by bypassing station is now considered mitigated

Identified risk: attack on station with filesystem vulnerabilities



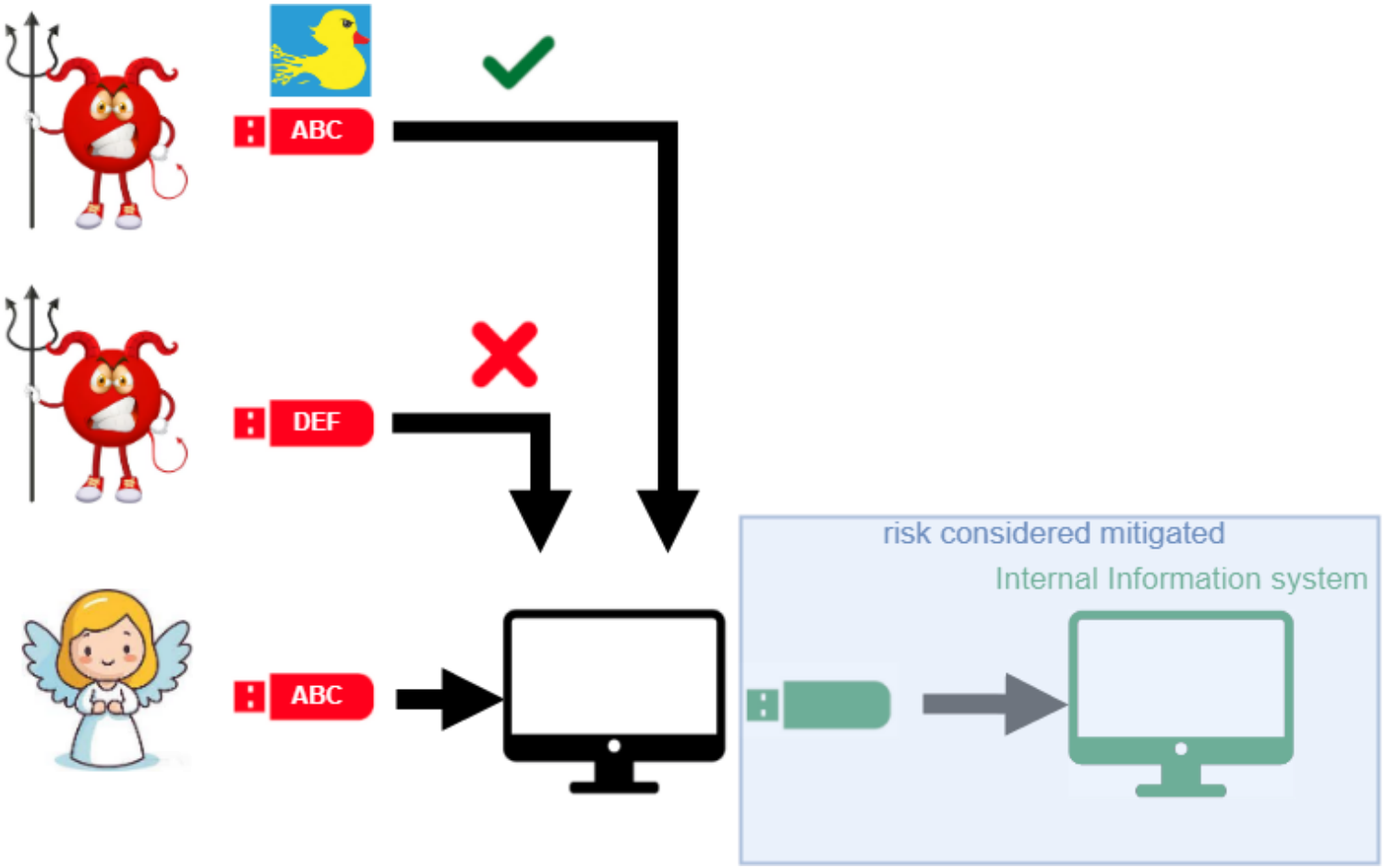
- Risk of document infiltration / exfiltration by bypassing station is now considered mitigated
- Risks related to the station itself shall now be considered

Identified risk: attack on station with filesystem vulnerabilities



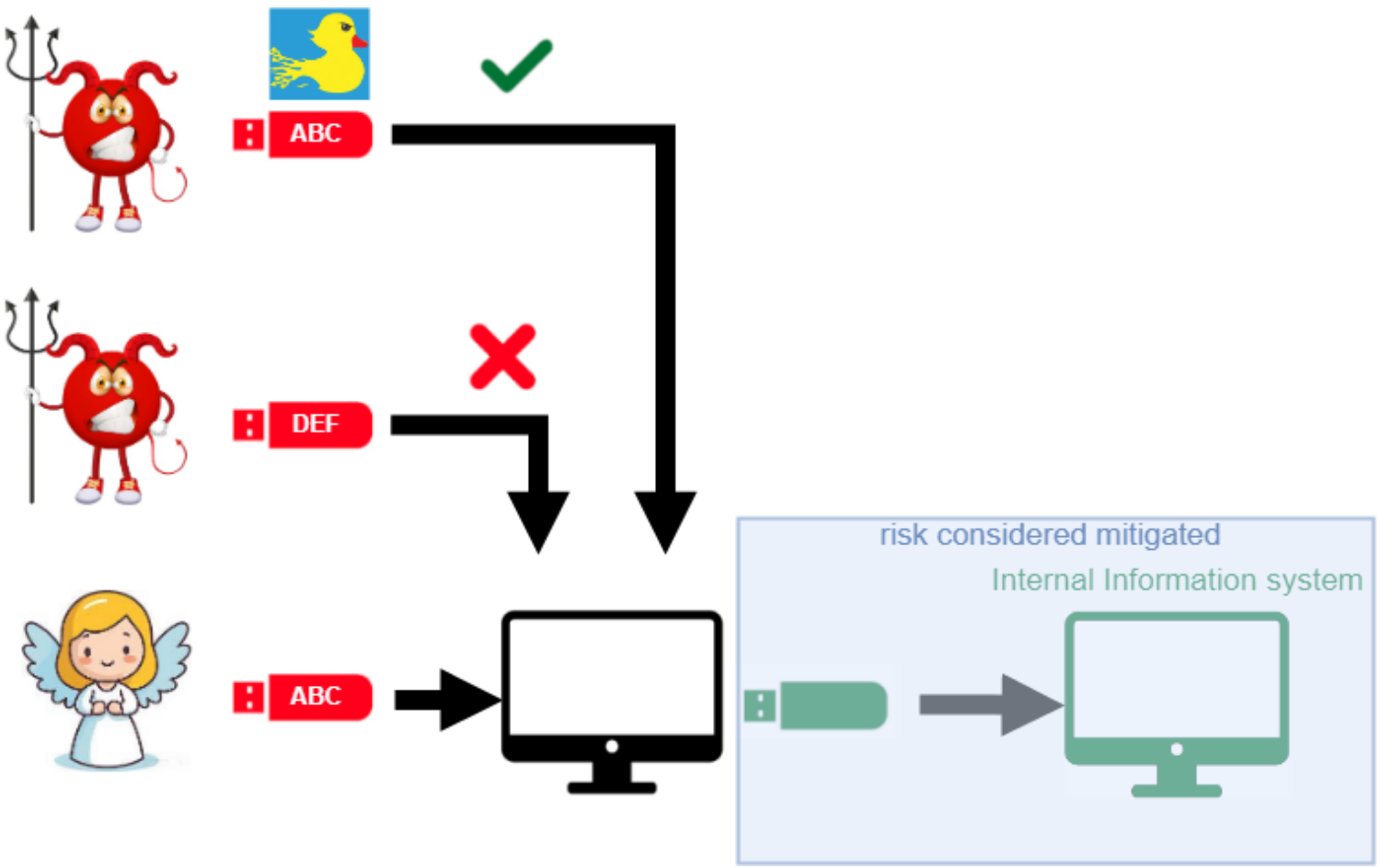
- Risk of document infiltration / exfiltration by bypassing station is now considered mitigated
- Risks related to the station itself shall now be considered
  - Malicious files on external USB stick

Identified risk: attack on station with filesystem vulnerabilities



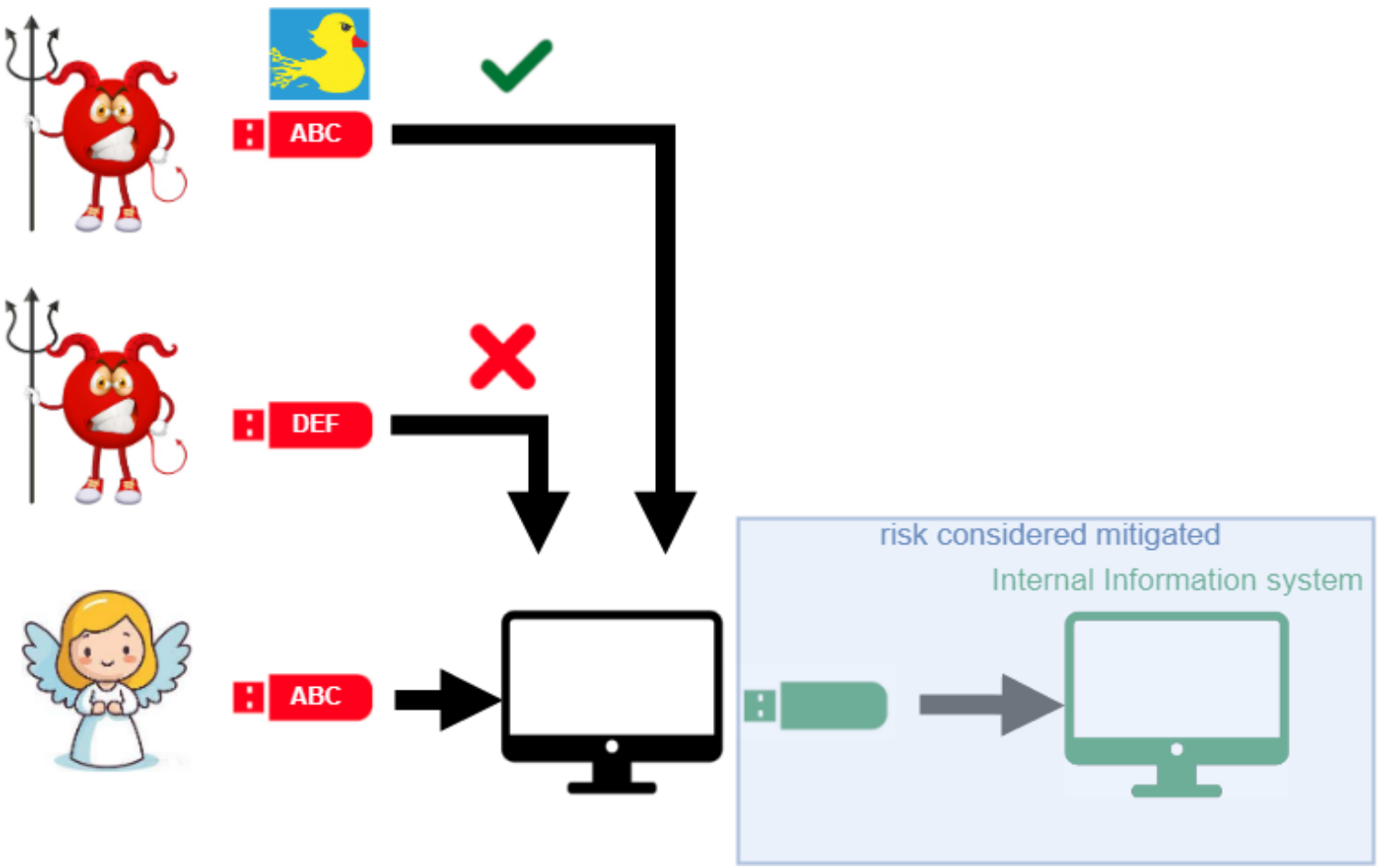
- Risk of document infiltration / exfiltration by bypassing station is now considered mitigated
- Risks related to the station itself shall now be considered
  - Malicious files on external USB stick
    - Mitigated by design

Identified risk: attack on station with filesystem vulnerabilities



- Risk of document infiltration / exfiltration by bypassing station is now considered mitigated
- Risks related to the station itself shall now be considered
  - Malicious files on external USB stick
    - Mitigated by design
  - Malicious on-disk filesystem formats (FAT32, NTFS, ExFAT, Ext4)

Identified risk: attack on station with filesystem vulnerabilities



- Risk of document infiltration / exfiltration by bypassing station is now considered mitigated
- Risks related to the station itself shall now be considered
  - Malicious files on external USB stick
    - Mitigated by design
  - Malicious on-disk filesystem formats (FAT32, NTFS, ExFAT, Ext4)
    - Not mitigated yet

VIVERIS

Solution #1: trust Linux kernel security

- Vulnerabilities of filesystem implementation in Linux kernel do exist
  - Example of a recent exploitable ExFAT vulnerability

Vulnerability Details : CVE-2023-4273

A flaw was found in the exFAT driver of the Linux kernel. The vulnerability exists in the implementation of the file name reconstruction function, which is responsible for reading file name entries from a directory index and merging file name parts belonging to one file into a single long file name. Since the file name characters are copied into a stack variable, a local privileged attacker could use this flaw to overflow the kernel stack.

CVSS scores for CVE-2023-4273

Base Score	Base Severity	CVSS Vector	Exploitability Score	Impact Score	Source
6.7	MEDIUM	CVSS:3.1/AV:L/AC:L/PR:H/UI:N/S:U/C:H/I:H/A:H	0.8	5.9	nvd@nist.gov
6.0	MEDIUM	CVSS:3.1/AV:L/AC:L/PR:H/UI:N/S:U/C:H/I:H/A:N	0.8	5.2	secalert@redhat.com

Solution #1: trust Linux kernel security

- Linux filesystem implementation will probably never be 100% secure

Evaluation	
Support of all FS	?
Security	?
Maintainability	?

Solution #1: trust Linux kernel security

- Linux filesystem implementation will probably never be 100% secure

Evaluation	
Support of all FS	?
Security	?
Maintainability	?

> *“ Note that of the mainstream file systems, ext4 and xfs don't guarantee that it's safe to blindly take maliciously provided file systems, such as those provided by a untrusted container, and mount it on a file system without problems. ”*
>
> Theodore Y. Ts’o – A maintainer of the Linux ext4 filesystem

Solution #1: trust Linux kernel security

- Linux filesystem implementation will probably never be 100% secure
  - Defining an allowlist of filesystems is a possible solution but that is not a comprehensive solution

Evaluation	
Support of all FS	?
Security	?
Maintainability	?

> *“ Note that of the mainstream file systems, ext4 and xfs don't guarantee that it's safe to blindly take maliciously provided file systems, such as those provided by a untrusted container, and mount it on a file system without problems. ”*
>
> Theodore Y. Ts’o – A maintainer of the Linux ext4 filesystem

Solution #1: trust Linux kernel security

- Linux filesystem implementation will probably never be 100% secure
  - Defining an allowlist of filesystems is a possible solution but that is not a comprehensive solution

Evaluation	
Support of all FS	+
Security	-
Maintainability	+

> *“ Note that of the mainstream file systems, ext4 and xfs don't guarantee that it's safe to blindly take maliciously provided file systems, such as those provided by a untrusted container, and mount it on a file system without problems. ”*
>
> Theodore Y. Ts’o – A maintainer of the Linux ext4 filesystem

Solution #1: trust Linux kernel security

- Linux filesystem implementation will probably never be 100% secure
  - Defining an allowlist of filesystems is a possible solution but that is not a comprehensive solution

Evaluation	
Support of all FS	+
Security	-
Maintainability	+

> *“ Note that of the mainstream file systems, ext4 and xfs don't guarantee that it's safe to blindly take maliciously provided file systems, such as those provided by a untrusted container, and mount it on a file system without problems. ”*
>
> Theodore Y. Ts’o – A maintainer of the Linux ext4 filesystem

Solution chosen by ANSSI:

Keysas is a modern decontamination station prototype, fast, which aims to be secure (https://keysas.fr)

Solution #2: userspace implementations of filesystems

- Userspace filesystem implementations that can be isolated with standard Linux security mechanisms

Evaluation	
Support of all FS	?
Security	?
Maintainability	?

Solution #2: userspace implementations of filesystems

- Userspace filesystem implementations that can be isolated with standard Linux security mechanisms
  - FAT32: grub-mount

Evaluation	
Support of all FS	?
Security	?
Maintainability	?

Solution #2: userspace implementations of filesystems

- Userspace filesystem implementations that can be isolated with standard Linux security mechanisms
  - FAT32: grub-mount
  - NTFS: FUSE NTFS-3G
  - ExFAT: FUSE ExFAT
  - EXT4: fuse2fs

Evaluation	
Support of all FS	?
Security	?
Maintainability	?

Solution #2: userspace implementations of filesystems

- Userspace filesystem implementations that can be isolated with standard Linux security mechanisms
  - FAT32: grub-mount
  - NTFS: FUSE NTFS-3G
  - ExFAT: FUSE ExFAT
  - EXT4: fuse2fs

Evaluation	
Support of all FS	-
Security	+
Maintainability	-

Solution #2: userspace implementations of filesystems

- Userspace filesystem implementations that can be isolated with standard Linux security mechanisms
  - FAT32: grub-mount
  - NTFS: FUSE NTFS-3G
  - ExFAT: FUSE ExFAT
  - EXT4: fuse2fs

Evaluation	
Support of all FS	-
Security	+
Maintainability	-

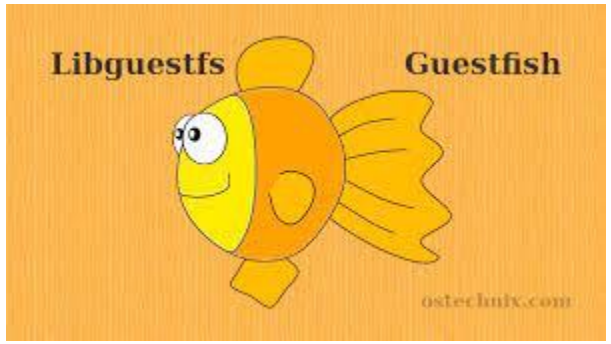
Solution chosen by the CEA:

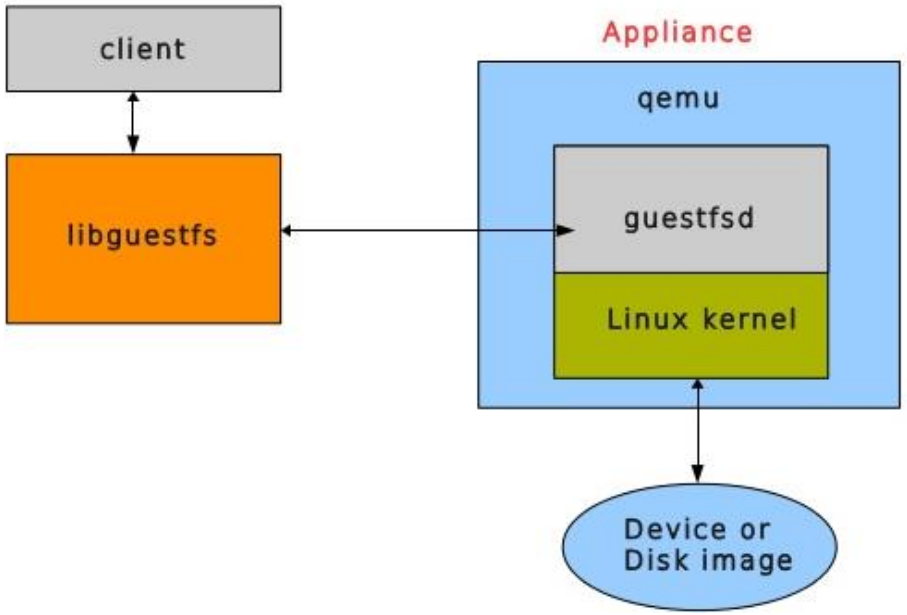
Tool and framework for securely reading untrusted USB mass storage devices (https://github.com/cea-sec/usbsas)

Solution #3: isolation with virtualization

- libguestfs (https://www.libguestfs.org/)
  - Virtualization to isolate filesystems from host Linux kernel
  - Use both well-known Linux kernel and FUSE implementations inside a Virtual Machine

Evaluation	
Support of all FS	?
Security	?
Maintainability	?



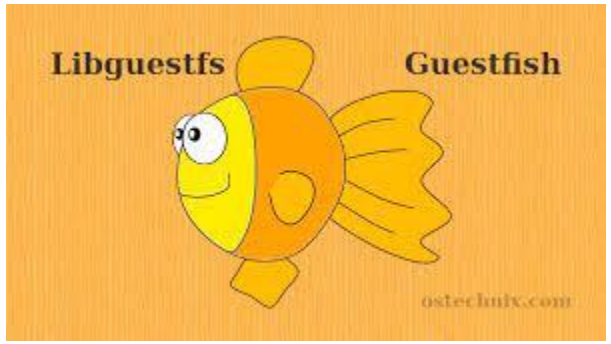


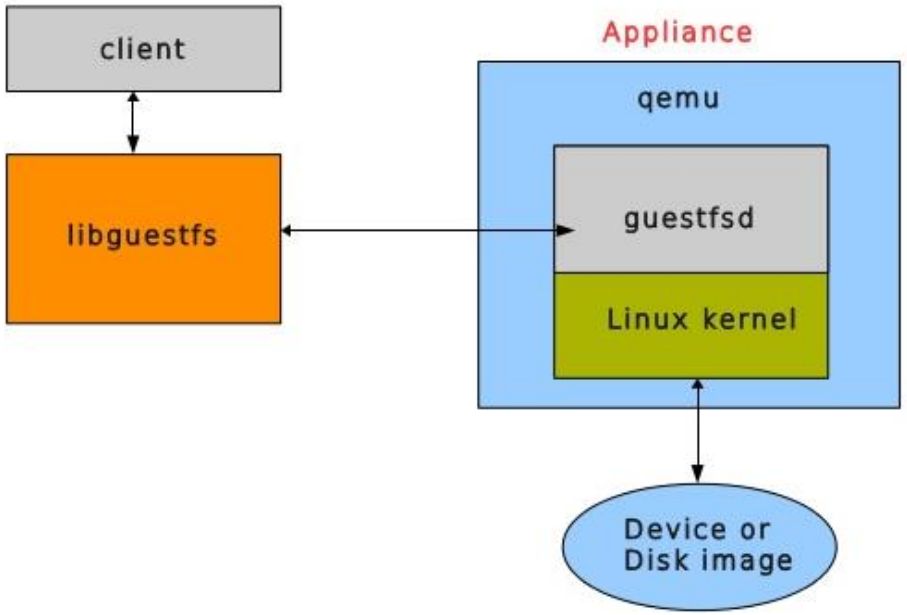
VIVERIS

Solution #3: isolation with virtualization

- libguestfs (https://www.libguestfs.org/)
  - Virtualization to isolate filesystems from host Linux kernel
  - Use both well-known Linux kernel and FUSE implementations inside a Virtual Machine

Evaluation	
Support of all FS	+
Security	+
Maintainability	+



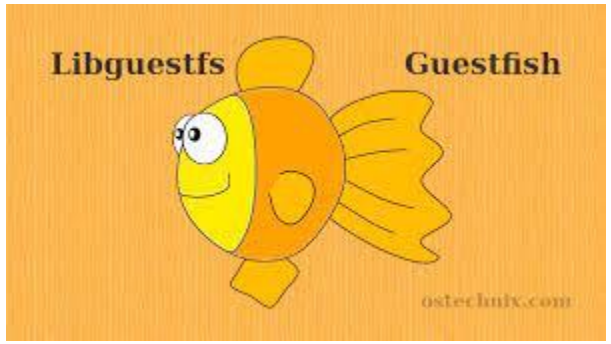


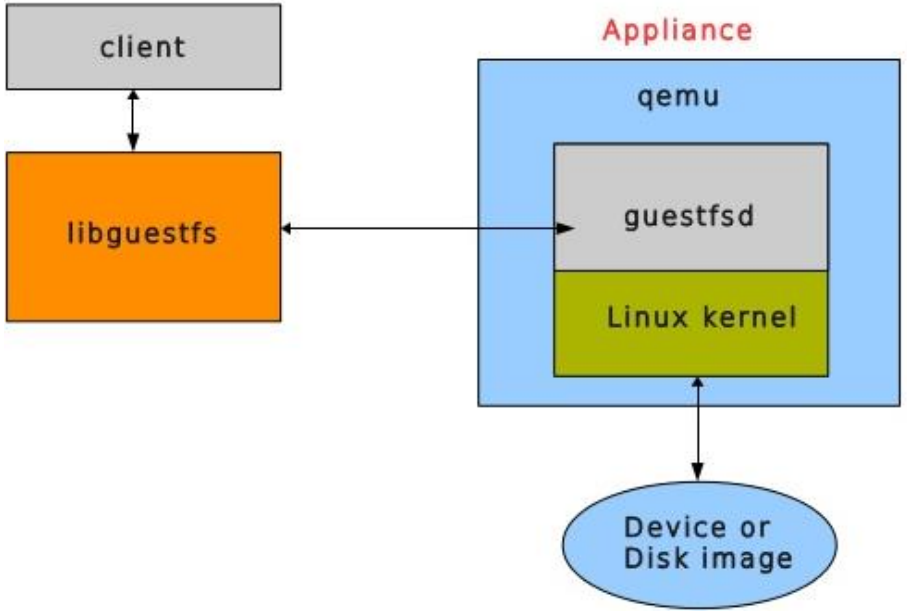
VIVERIS

Solution #3: isolation with virtualization

- libguestfs (https://www.libguestfs.org/)
  - Virtualization to isolate filesystems from host Linux kernel
  - Use both well-known Linux kernel and FUSE implementations inside a Virtual Machine
  - Python binding available

Evaluation	
Support of all FS	+
Security	+
Maintainability	+



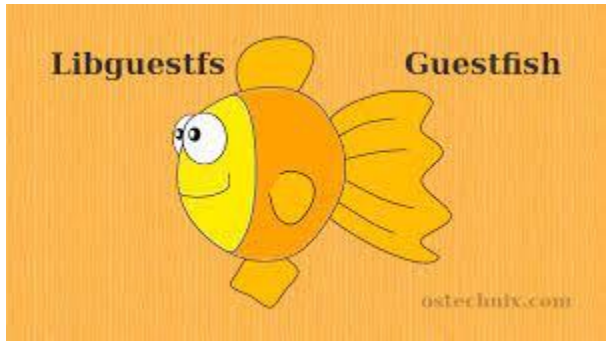


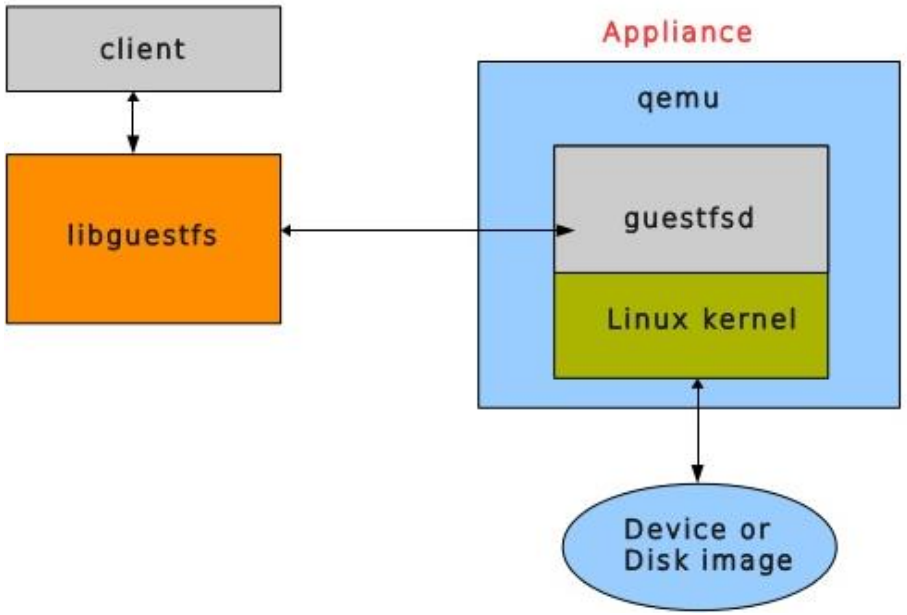
VIVERIS

Solution #3: isolation with virtualization

- libguestfs (https://www.libguestfs.org/)
  - Virtualization to isolate filesystems from host Linux kernel
  - Use both well-known Linux kernel and FUSE implementations inside a Virtual Machine
  - Python binding available

Evaluation	
Support of all FS	+
Security	+
Maintainability	+

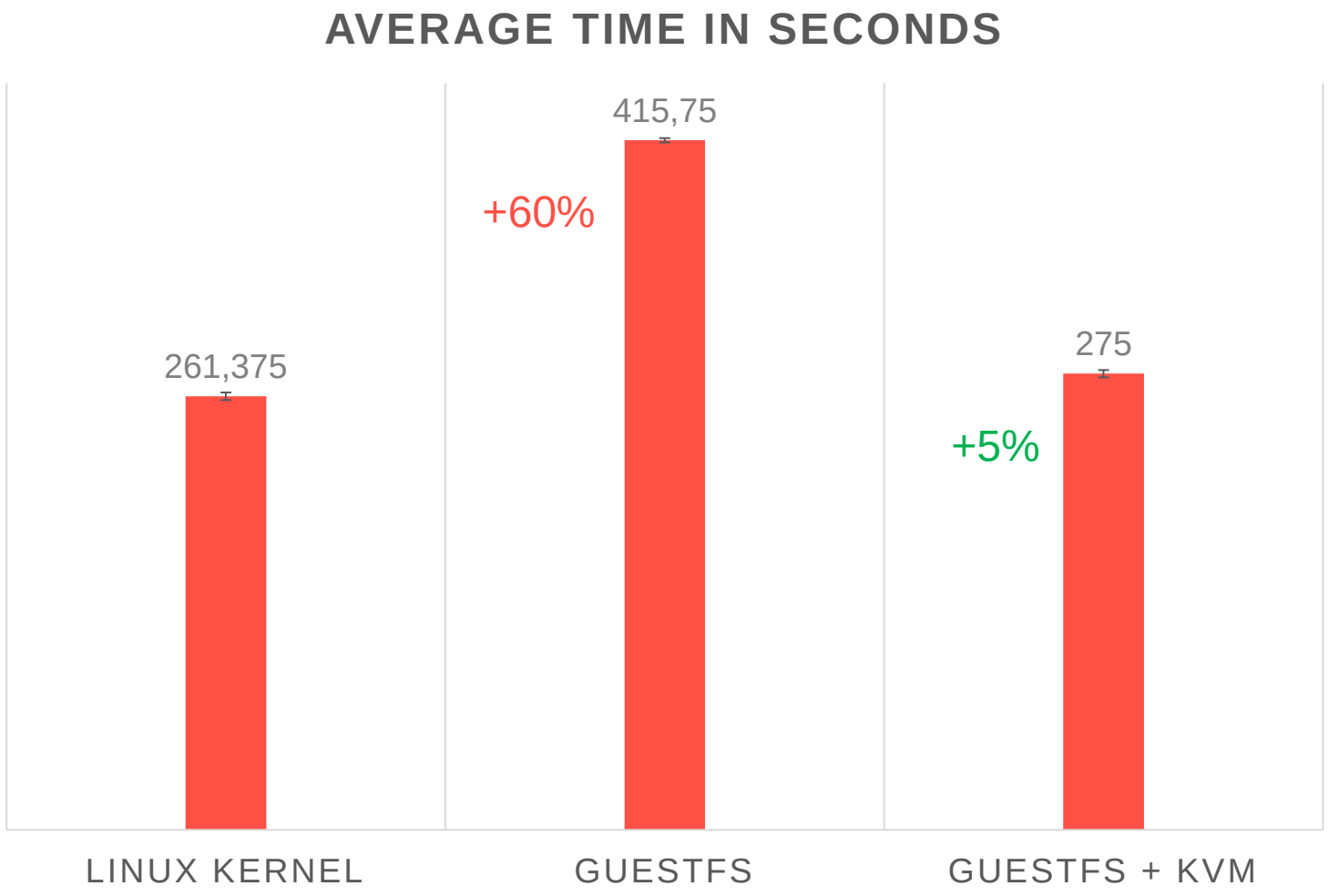




Solution matches all evaluation criteria, but what about the performance impact of virtualization?

Solution #3: tool to mount media on virtual machine

- Test protocol
  - Measure wall-clock time for
    - Mount input USB disk
    - Copy its content in tmpfs (4x 1GB)
    - Mount output USB disk
    - Copy content to output USB disk
    - Force disk sync
  - Run 10 times
    - Drop shortest and longest time
    - Measure standard deviation
  - Use same ExFAT SanDisk USB-3 disk
- Run test protocol for
  - Linux kernel (for reference)
  - Libguestfs without KVM support
  - Libguestfs with KVM support



Conclusion

- Mitigation of data infiltration/exfiltration risks
  - USB serial ID access control
  - Encrypted sticks were implemented

- Mitigation of filesystems vulnerabilities

Evaluation	#1 (Linux kernel)	#2 (FUSE)	#3 (Libguestfs)
Support of all FS	+	-	+
Security	-	+	+
Maintainability	+	-	+

  - Libguestfs chosen

Sources

- https://www.cvedetails.com/cve/CVE-2023-4273/
- https://dfir.ru/2023/08/23/cve-2023-4273-a-vulnerability-in-the-linux-exfat-driver/
- https://lwn.net/ml/linux-kernel/20190818155812.GB13230@infradead.org/
- https://www.gnutoolbox.com/libguestfs-manage-virtual-machine-disk-images/
- All icons are from Noun Project (CC BY 3.0)

VIVERIS

Innover. Simplifier. Partager.