2021 embedded vision summit® VIRTUAL | MAY 25-28

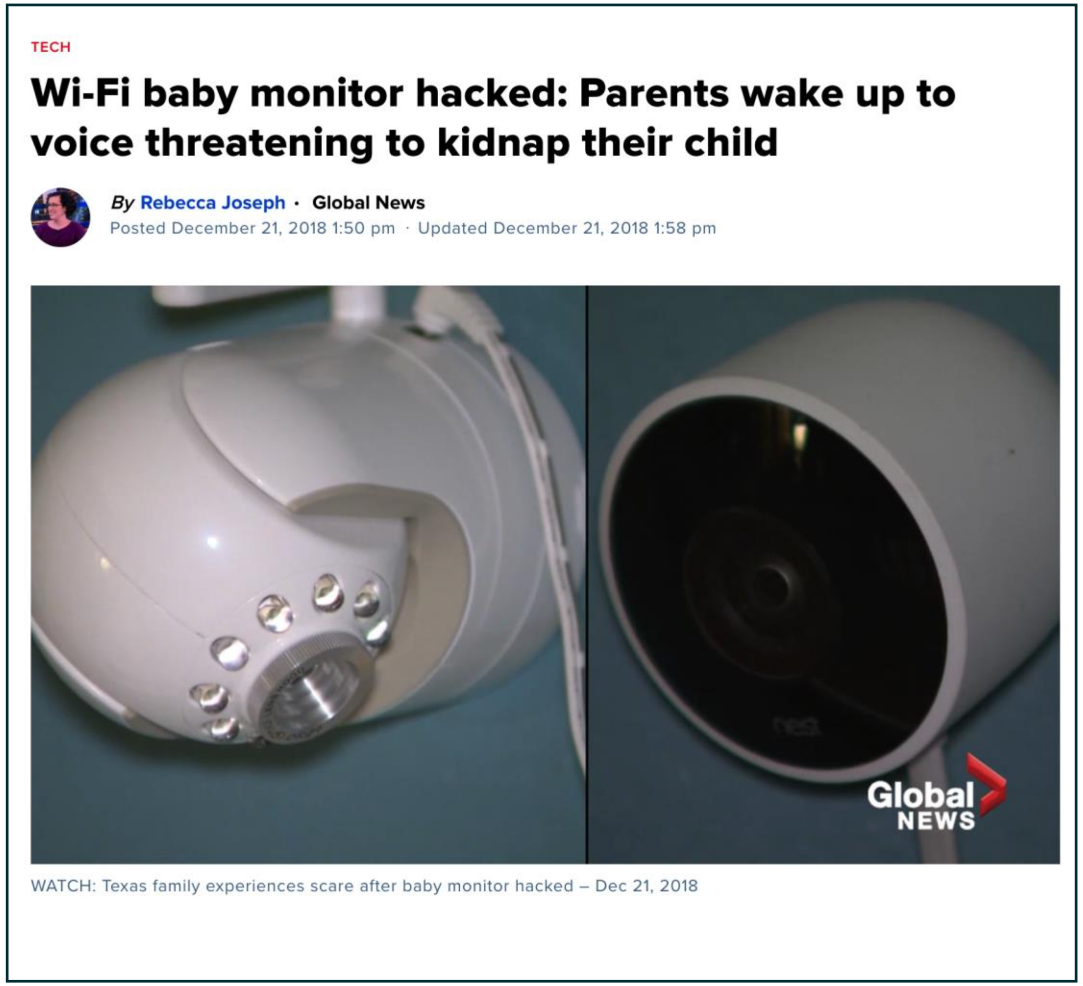
Secure Hardware Architecture for Embedded Vision

Jonathan Cefalu
CEO, NeuroBinder, Inc.

NeuroBinder

We Need to Take Advanced Hackers More Seriously

- Hackers are much more skilled than most product engineers give them credit for.
- This leads to camera-based products repeatedly getting hacked, and a growing societal distrust of cameras in general.
- To protect our customers, we need to change our mindset and find an embedded vision hardware architecture that makes it *electrically impossible* for a hacker to get access to live video.



TECH

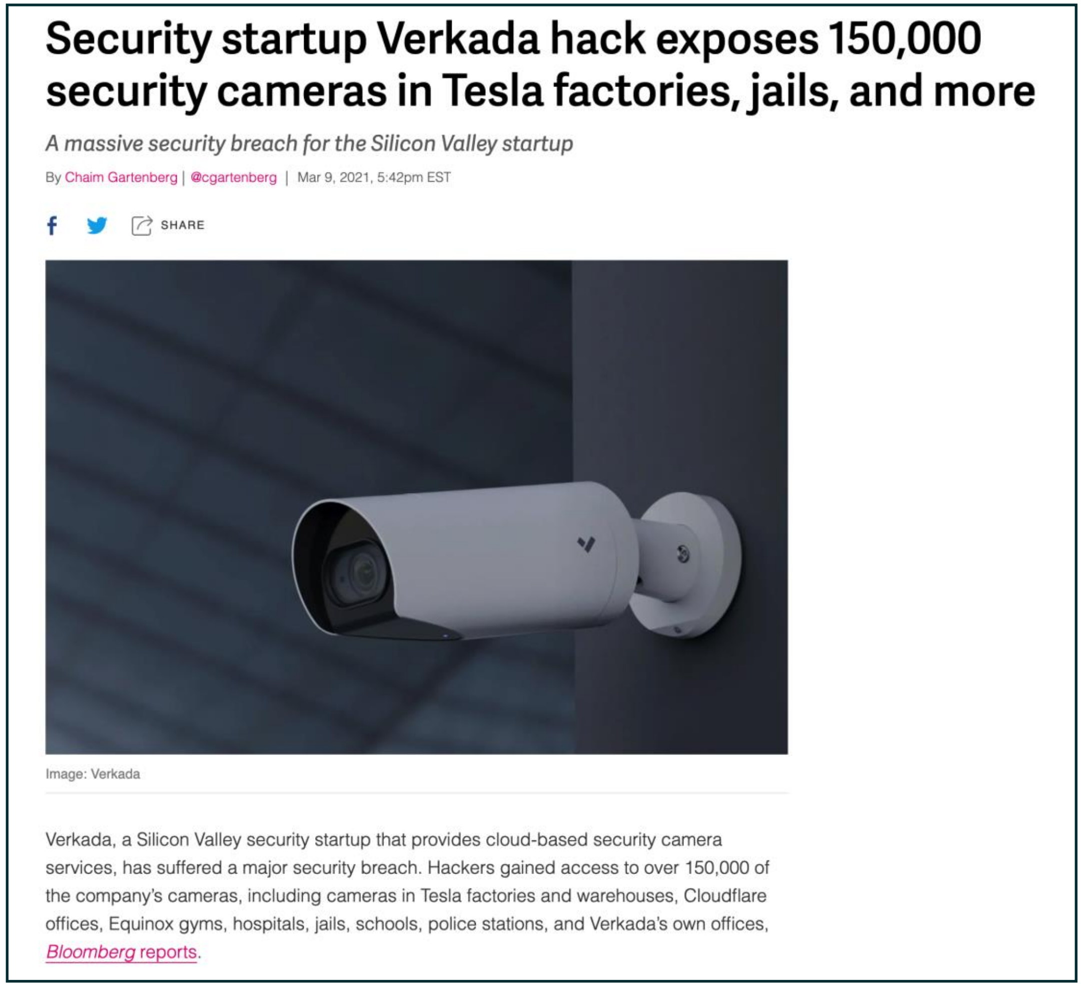
Wi-Fi baby monitor hacked: Parents wake up to voice threatening to kidnap their child

By Rebecca Joseph · Global News
Posted December 21, 2018 1:50 pm · Updated December 21, 2018 1:58 pm

WATCH: Texas family experiences scare after baby monitor hacked – Dec 21, 2018

We Need to Take Advanced Hackers More Seriously

- Hackers are much more skilled than most product engineers give them credit for.
- This leads to camera-based products repeatedly getting hacked, and a growing societal distrust of cameras in general.
- To protect our customers, we need to change our mindset and find an embedded vision hardware architecture that makes it *electrically impossible* for a hacker to get access to live video.



Security startup Verkada hack exposes 150,000 security cameras in Tesla factories, jails, and more

A massive security breach for the Silicon Valley startup

By Chaim Gartenberg | @cgartenberg | Mar 9, 2021, 5:42pm EST

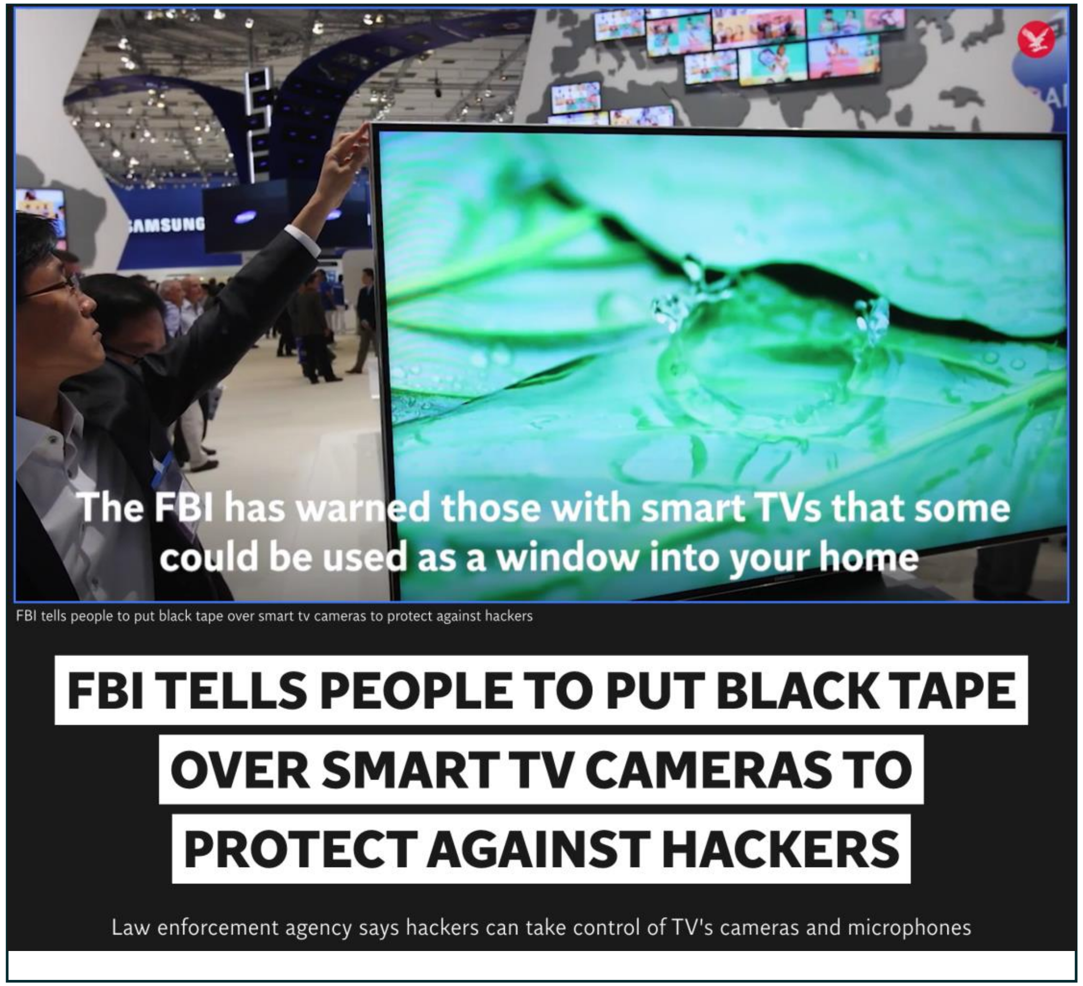
SHARE

Image: Verkada

Verkada, a Silicon Valley security startup that provides cloud-based security camera services, has suffered a major security breach. Hackers gained access to over 150,000 of the company's cameras, including cameras in Tesla factories and warehouses, Cloudflare offices, Equinox gyms, hospitals, jails, schools, police stations, and Verkada's own offices, *Bloomberg reports*.

We Need to Take Advanced Hackers More Seriously

- Hackers are much more skilled than most product engineers give them credit for.
- This leads to camera-based products repeatedly getting hacked, and a growing societal distrust of cameras in general.
- To protect our customers, we need to change our mindset and find an embedded vision hardware architecture that makes it *electrically impossible* for a hacker to get access to live video.


The FBI has warned those with smart TVs that some could be used as a window into your home

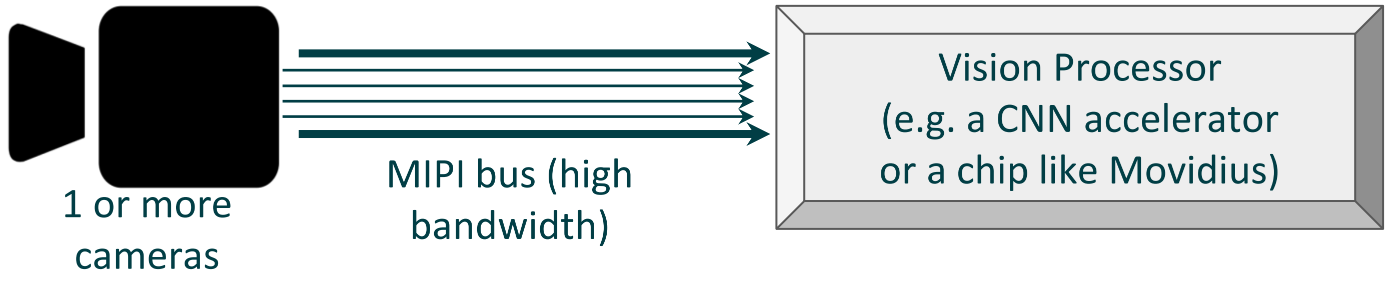
FBI tells people to put black tape over smart tv cameras to protect against hackers

FBI TELLS PEOPLE TO PUT BLACK TAPE OVER SMART TV CAMERAS TO PROTECT AGAINST HACKERS

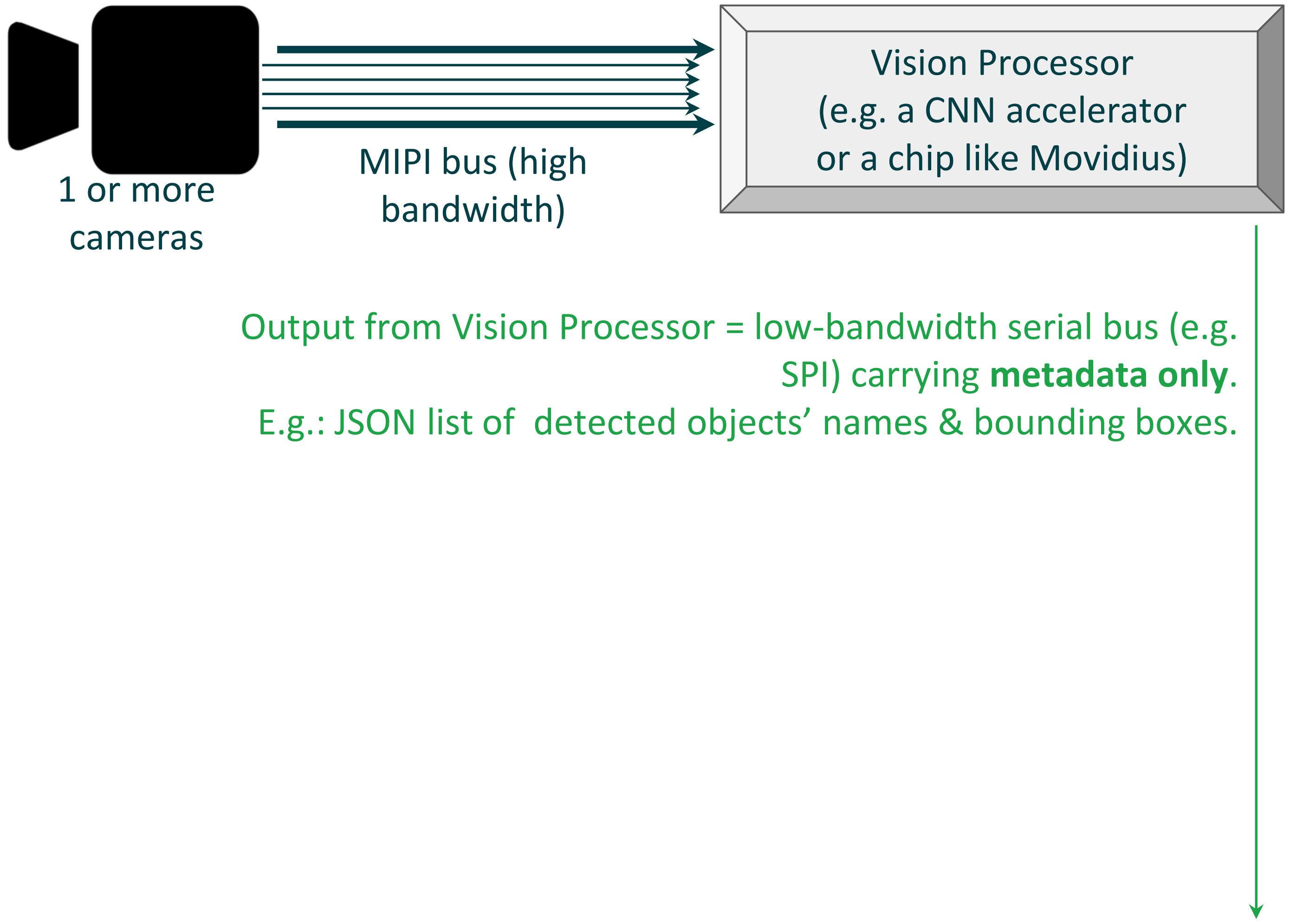
Law enforcement agency says hackers can take control of TV's cameras and microphones

Key Concepts

- Camera imagery flows from the camera(s) to a dedicated vision processor (e.g. a CNN accelerator or other embedded vision chip).
- The vision processor converts the camera imagery into **metadata** such as e.g. the names of objects in the scene, and the bounding box of each object.

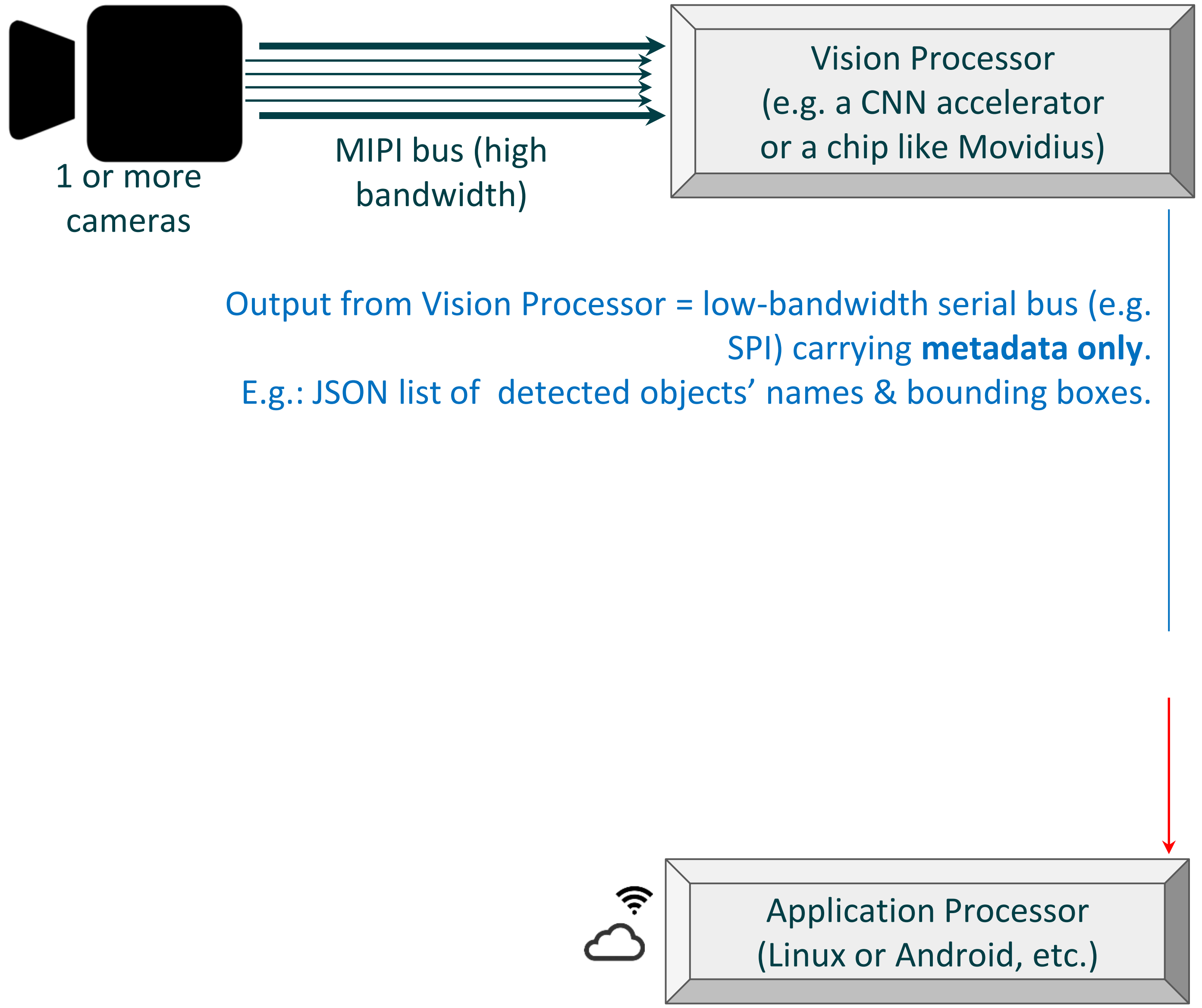

1 or more cameras
MIPI bus (high bandwidth)
Vision Processor (e.g. a CNN accelerator or a chip like Movidius)

Key Concept: Restrict Bandwidth of Output Bus


1 or more cameras
MIPI bus (high bandwidth)
Vision Processor (e.g. a CNN accelerator or a chip like Movidius)
Output from Vision Processor = low-bandwidth serial bus (e.g. SPI) carrying **metadata only**.
E.g.: JSON list of detected objects' names & bounding boxes.

- The vision processor is wired so as to have its only output interfaces be very low bandwidth (e.g. only SPI) which are intentionally incapable of streaming camera frames.
- The metadata that the vision processor generates is outputted via JSON or a similar representation over the low-bandwidth output ports.
- Since **no high-bandwidth output ports exist**, even if the vision processor were hacked, it cannot leak live video to the outside world.

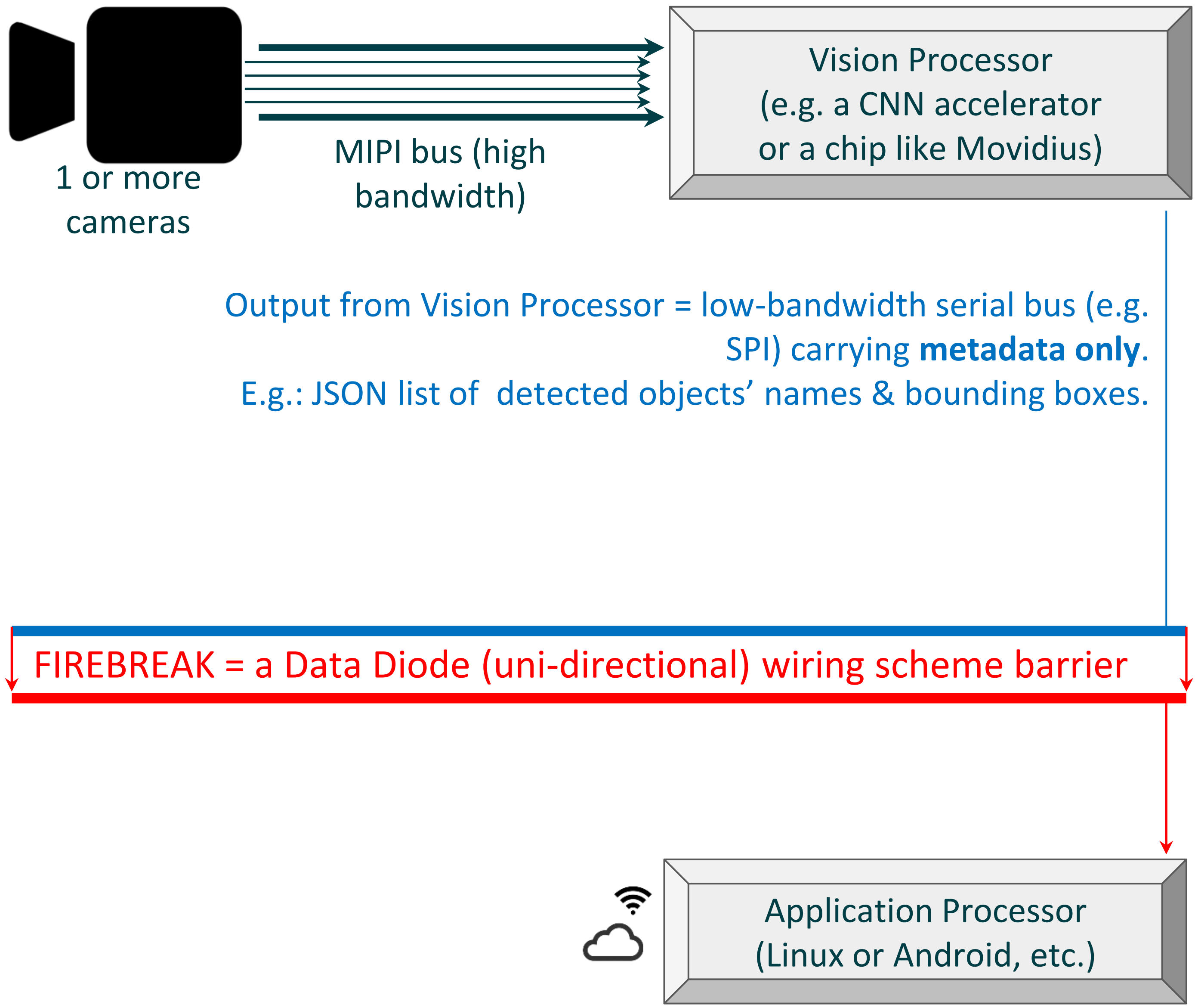
Key Concept: Firebreak = 1-way Data Diode



1 or more cameras
MIPI bus (high bandwidth)
Vision Processor (e.g. a CNN accelerator or a chip like Movidius)
Application Processor (Linux or Android, etc.)

Output from Vision Processor = low-bandwidth serial bus (e.g. SPI) carrying **metadata only**.
E.g.: JSON list of detected objects’ names & bounding boxes.

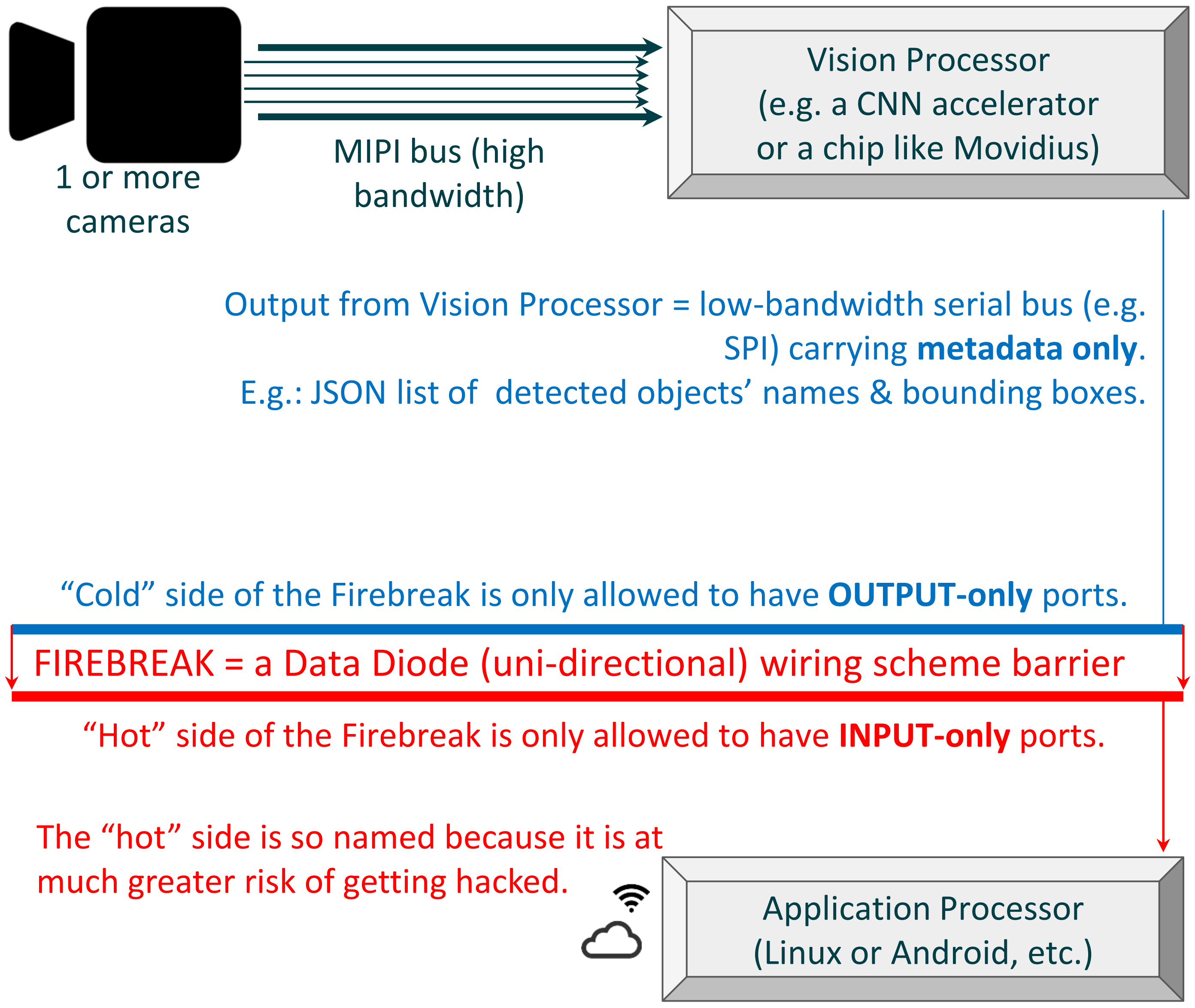
Key Concept: Firebreak = 1-way Data Diode


1 or more cameras
MIPI bus (high bandwidth)
Vision Processor (e.g. a CNN accelerator or a chip like Movidius)
Application Processor (Linux or Android, etc.)

Output from Vision Processor = low-bandwidth serial bus (e.g. SPI) carrying **metadata only**.
E.g.: JSON list of detected objects' names & bounding boxes.

FIREBREAK = a Data Diode (uni-directional) wiring scheme barrier

Key Concept: Firebreak = 1-way Data Diode


1 or more cameras
MIPI bus (high bandwidth)
Vision Processor (e.g. a CNN accelerator or a chip like Movidius)
Application Processor (Linux or Android, etc.)

Output from Vision Processor = low-bandwidth serial bus (e.g. SPI) carrying **metadata only**.
E.g.: JSON list of detected objects' names & bounding boxes.

"Cold" side of the Firebreak is only allowed to have **OUTPUT-only** ports.

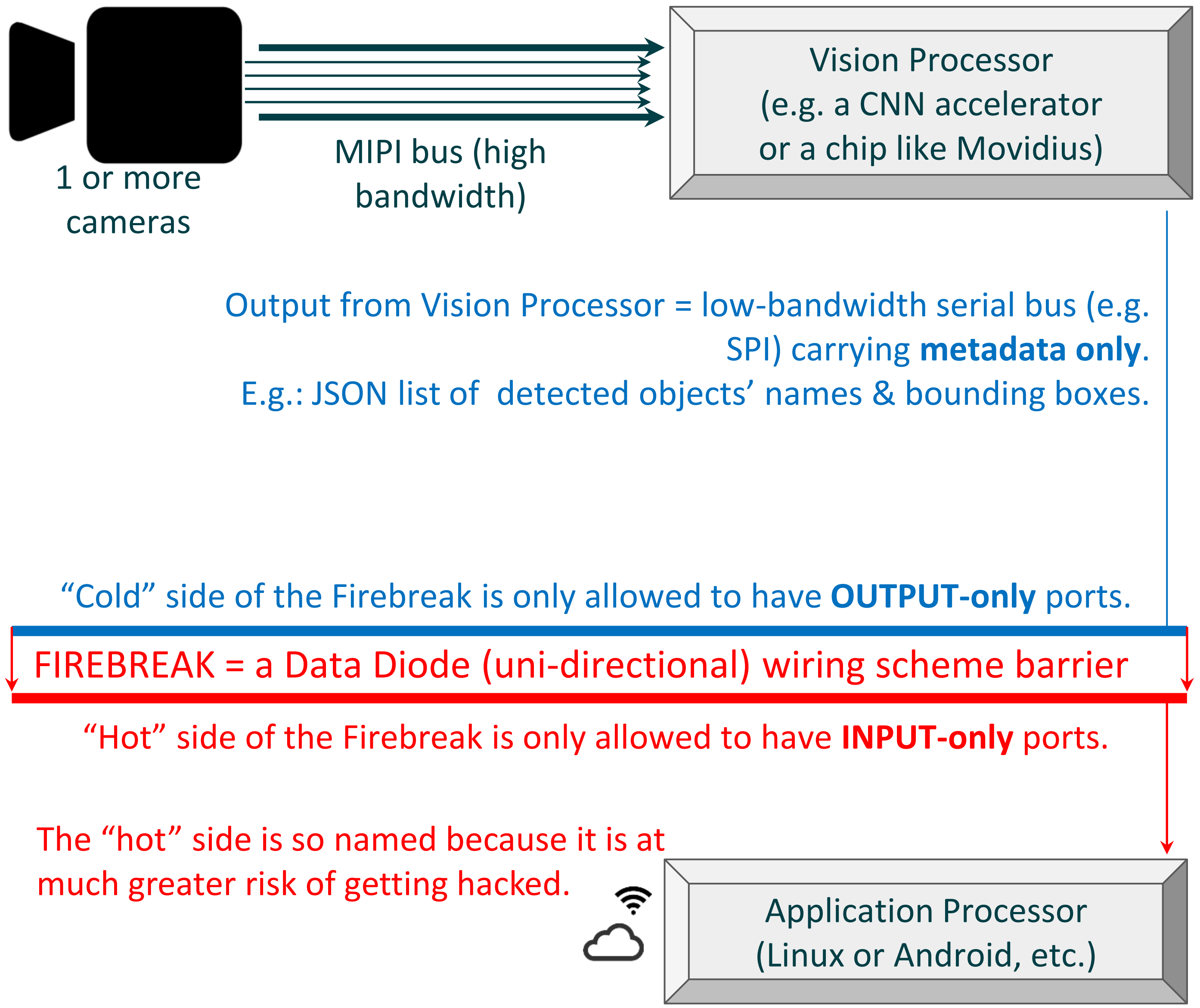
FIREBREAK = a Data Diode (uni-directional) wiring scheme barrier

"Hot" side of the Firebreak is only allowed to have **INPUT-only** ports.

The "hot" side is so named because it is at much greater risk of getting hacked.

Key Concept: Firebreak = 1-way Data Diode

- The system is wired so as to establish a **Firebreak** — a dividing line that you build into your system between two **strictly separated** sides called the "cold" side and the "hot" side.
- The Firebreak seeks to prevent a hacker who has taken complete control of all firmware on the "hot" side from being able to find a way into the "cold" side.
- This is one application of a common concept called a Data Diode (see Resources slide).
- **Every** connection across the Firebreak must be wired uni-directionally:
  1 **output-only** electrical pin on the "cold" side
  + 1 **input-only** electrical pin on the "hot" side"


1 or more cameras
MIPI bus (high bandwidth)
Vision Processor (e.g. a CNN accelerator or a chip like Movidius)
Application Processor (Linux or Android, etc.)

Output from Vision Processor = low-bandwidth serial bus (e.g. SPI) carrying **metadata only**.
E.g.: JSON list of detected objects' names & bounding boxes.

"Cold" side of the Firebreak is only allowed to have **OUTPUT-only** ports.

FIREBREAK = a Data Diode (uni-directional) wiring scheme barrier

"Hot" side of the Firebreak is only allowed to have **INPUT-only** ports.

The "hot" side is so named because it is at much greater risk of getting hacked.

Key Concept: Firebreak = 1-way Data Diode

- *IF THIS RULE IS NOT ENFORCED*
  100% STRICTLY, THE FIREBREAK CAN BE BREACHED.
- **NO EXCEPTIONS!**
- *THIS MEANS*
  NO EXCEPTIONS
  NOT FOR JTAG, I2C, PCIe, NOTHING.

- **Every** connection across the Firebreak must be wired uni-directionally:
  1 **output-only** electrical pin on the "cold" side
  + 1 **input-only** electrical pin on the "hot" side"

Challenges and their Mitigations

Q: **If the “cold” section is output-only, how can its firmware be updated?**

A: The cold section’s fw should be updatable only by direct physical access by the user taking a rare, intentional action, such as plugging a cable + pressing a button.

- E.g., suppose that the product uses USB for charging. One option would be to connect the USB data lines to the cold section and to configure the cold section’s chips to accept a fw update via USB **only if the product’s reset button is pressed while USB is connected**.
  - Requiring the cold section fw update process to be initiated via a button press mitigates the possibility that an attacker could surreptitiously install firmware when the device is charging, by hacking the USB host PC.
- Firmware for the hot section can be installed using over-the-air updates over Wi-Fi / BT / etc.
- Both the cold and hot sections should use code signing with strong encryption.

Challenges and their Mitigations

Q: Don't some applications fundamentally require outputting video? Is there a way to make those more secure?

A: Some applications certainly do fundamentally require outputting video, but I invite you to consider that many applications do not really require outputting video if you think creatively.

- For example, a system designed to count the number of shoppers in a store should count the people using on-device processing, and only transmit the real-time number of people in the frame; this system would not have any good reason for transmitting video!
- In some applications, the ability to view live video is only relevant for the development stage or for debugging issues in the field. The best way to handle these cases is to make it so that debugging can be performed by plugging a cable (e.g. USB or Ethernet) into a debug port built into the "cold" side of the system. This cold-side debug port would only be used for debugging, and would not be used by a customer. This port does not violate the Firebreak because it **(1) does not connect cold to hot** and **(2) does not connect cold to the Internet or a radio**.
- Many applications do not need to transmit live video, but could make do with simply transmitting live metadata + recording archival video to disk, with that disk drive being a part of the cold side. This disk storage would later be accessed by a USB port built into the cold side if the video is ever needed (e.g. for investigating a burglary).

Challenges and their Mitigations

Q: **What about cost? Won't including a dedicated vision chip be more expensive?**

A: There are other, lower-cost ways to implement the Firebreak concept other than the one presented in the previous slides. Here are two lower-cost possibilities which might work for your use case:

- Option 1: If the computer vision workload of your device is low, and if your device's camera requirements (FoV, low light sensitivity, etc.) are suitable, you might be able to use a camera module such as the Qualcomm Glance, which has a small embedded vision chip built right into the camera. The Qualcomm Glance still needs a host to initialize the module at boot, but this host could be any low-cost (~$0.20) microcontroller. Then you can establish a Firebreak between that microcontroller and the main Application Processor.
- Option 2: If your device only needs to send data to the Internet but never needs to receive anything, you can put the Firebreak in between the Application Processor and the radio. This "transmit only" condition applies to certain types of IoT sensor devices. For this approach, you need a radio/modem module which is capable of operating in a standalone manner running all of its own required firmware (e.g. the Espressif ESP32), and you can establish a Firebreak in between the Application Processor and the radio module.

Conclusions

- We need a new way of thinking about security for embedded vision devices.
- Attackers are much more capable than we’ve assumed.
- If something can be hacked, it will be hacked.
- The responsible solution is to design our hardware so that the hardware itself cannot be hacked, by wiring in certain security guarantees at the schematic level.

- The Firebreak concept is one application of the Data Diode technique (see Resources slide).
- To implement a Firebreak, you need to split your system into a “cold” (most secure) section and a “hot” (application code) section, with **strict separation** between the two sections.
- Every connection across the Firebreak must be wired uni-directionally:
  1 **output-only** electrical pin on the “cold” side
  + 1 **input-only** electrical pin on the “hot” side”

Resources

Further Reading

Wikipedia: Unidirectional Network
https://en.wikipedia.org/wiki/Unidirectional_network

Fiber System: Data Diodes for Isolated and Classified Networks
https://www.fibersystem.com/data-diodes/

Owl Cyber Defense: Learn About Data Diodes
https://owlcyberdefense.com/learn-about-data-diodes/

Forcepoint: Segment and Defend Networks with Automated Uni-Directional Transfer
https://www.forcepoint.com/sites/default/files/resources/datasheets/datasheet_data_diode_en.pdf

Email for Questions:
jon@neurobinder.com