Ciclo de Palestras em Computação
UFES - 2022

UFES

Ivan De Oliveira Nunes
Rochester Institute of Technology
http://ivan.csec.rit.edu

Agenda

1. Minha trajetória desde a UFES (2009)
2. Dicas e conselhos gratuitos (se fosse bom vendia)

   - Ou: *"**existe vida após a graduação na UFES?**"*

3. Oportunidades para alunos da UFES no RIT
4. Um pouco do minha pesquisa: IoT & MCU Security

De volta a 2009…

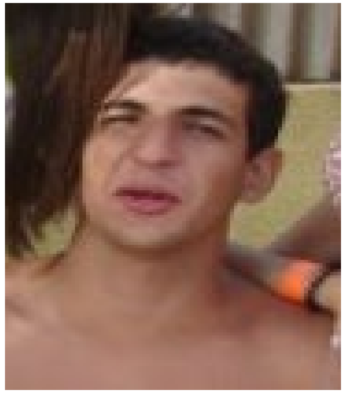
2008/2009

2008/2009

Passei no vestibular!
Agora to de boa!

2008/2009

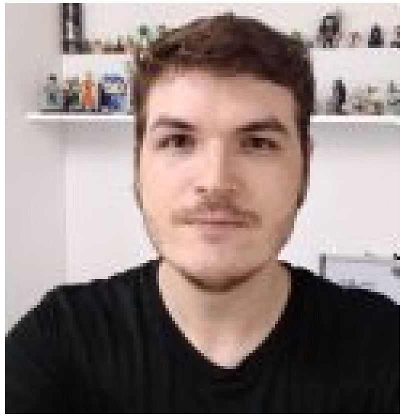
Passei no vestibular!
Agora to de boa!



6 meses depois ☹

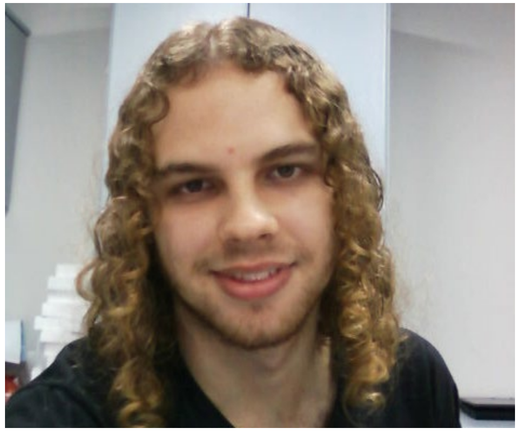
Outros “inocentes” 2009 -> 2022

Outros “inocentes” 2009 -> 2022

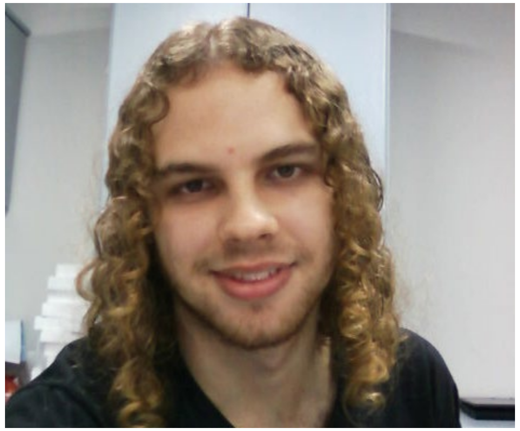


Prof. Andre Pacheco
(DI - UFES)

Outros “inocentes” 2009 -> 2022

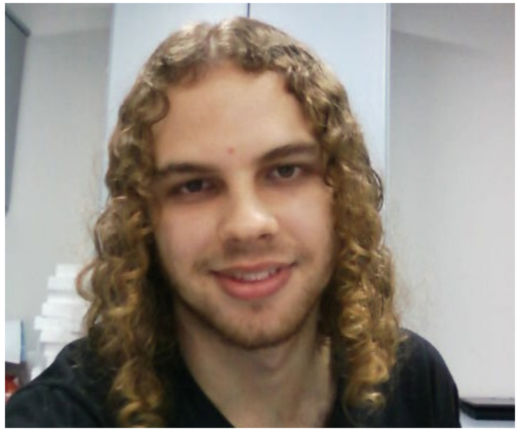


Outros “inocentes” 2009 -> 2022



Marcos Couto
(Android Developer - PicPay)

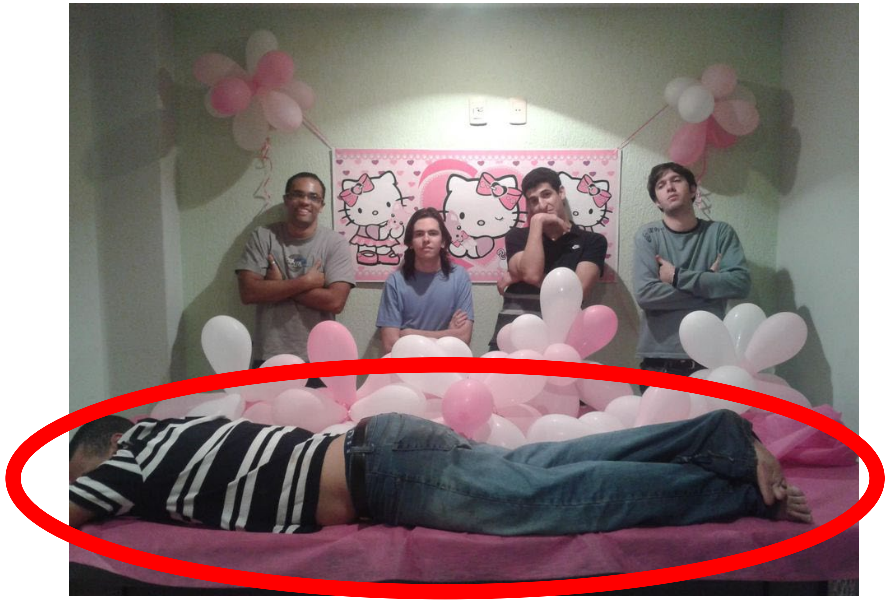
Outros “inocentes” 2009 -> 2022



Marcos Couto
(Android Developer - PicPay)

Outros “inocentes” 2009 -> 2022

Outros “inocentes” 2009 -> 2022



Outros “inocentes” 2009 -> 2022

Juan Franca
1º Tenente da Marinha do Brasil

Turma Eng. Comp. 2009

ENG COMP 09ERS

Se nós sobrevivemos, você também consegue!!!

Frases que ouço hoje dos meus colegas de turma que eu gostaria de ter ouvido quando era aluno

Conselhos gratuitos

1. “Todo mundo sobreviveu”:

Conselhos gratuitos

1. “Todo mundo sobreviveu”:
    - Obs: saúde (física e mental) em 1º lugar.
    - Todos bem (pessoalmente e profissionalmente)!
    - Não desista: forme-se! Vale a pena. ☺

Conselhos gratuitos

1. “Todo mundo sobreviveu”:
   - Obs: saúde (física e mental) em 1º lugar.
   - Todos bem (pessoalmente e profissionalmente)!
   - Não desista: forme-se! Vale a pena. ☺

2. “O tempo passa pra todo mundo”:

Conselhos gratuitos

1. “Todo mundo sobreviveu”:
   - Obs: saúde (física e mental) em 1º lugar.
   - Todos bem (pessoalmente e profissionalmente)!
   - Não desista: forme-se! Vale a pena. ☺

2. “O tempo passa pra todo mundo”:
   - Aproveite as oportunidades durante a graduação.

Exemplo: atividades durante a minha graduação (e as portas que elas abriram)

Exemplo: atividades durante a minha graduação (e as portas que elas abriram)

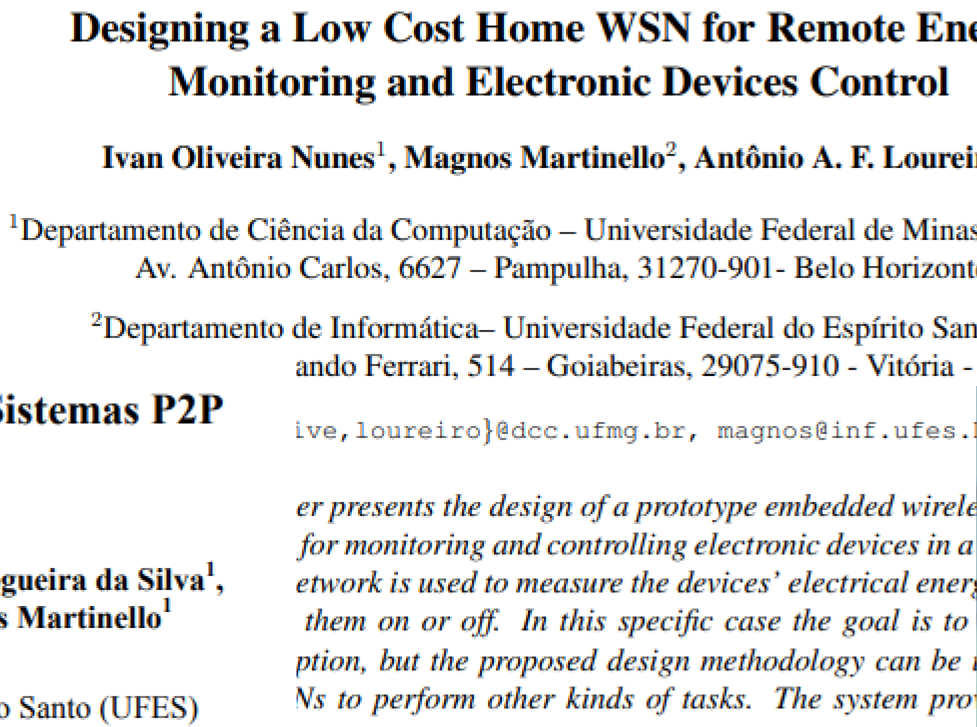
- PET Eng Comp (2009 - 2012):

Exemplo: atividades durante a minha graduação (e as portas que elas abriram)

- Torneios de Robótica (2010, 2012, 2013, 2014)
- ERUS (criada em 2012)

Exemplo: atividades durante a minha graduação (e as portas que elas abriram)

- Ensino, Pesquisa e ICs



Designing a Low Cost Home WSN for Remote Energy Monitoring and Electronic Devices Control

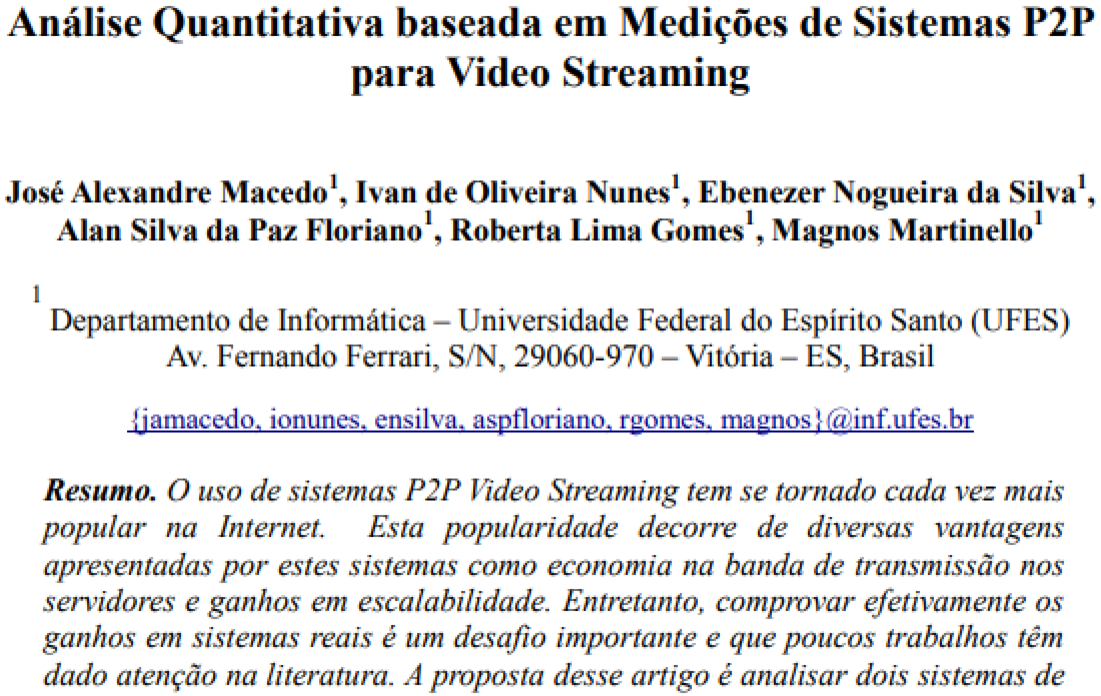
Ivan Oliveira Nunes[1], **Magnos Martinello**[2], **Antônio A. F. Loureiro**[1]

[1]Departamento de Ciência da Computação – Universidade Federal de Minas Gerais (UFMG)
Av. Antônio Carlos, 6627 – Pampulha, 31270-901- Belo Horizonte - MG

[2]Departamento de Informática– Universidade Federal do Espírito Santo (UFES)
ando Ferrari, 514 – Goiabeiras, 29075-910 - Vitória - ES

ive,loureiro}@dcc.ufmg.br, magnos@inf.ufes.

er presents the design of a prototype embedded wirele
for monitoring and controlling electronic devices in a
etwork is used to measure the devices' electrical ener
them on or off. In this specific case the goal is to
ption, but the proposed design methodology can be
Ns to perform other kinds of tasks. The system pro



Análise Quantitativa baseada em Medições de Sistemas P2P para Video Streaming

José Alexandre Macedo[1], **Ivan de Oliveira Nunes**[1], **Ebenezer Nogueira da Silva**[1], **Alan Silva da Paz Floriano**[1], **Roberta Lima Gomes**[1], **Magnos Martinello**[1]

[1] Departamento de Informática – Universidade Federal do Espírito Santo (UFES)
Av. Fernando Ferrari, S/N, 29060-970 – Vitória – ES, Brasil

{jamacedo, ionunes, ensilva, aspfloriano, rgomes, magnos}@inf.ufes.br

Resumo. *O uso de sistemas P2P Video Streaming tem se tornado cada vez mais popular na Internet. Esta popularidade decorre de diversas vantagens apresentadas por estes sistemas como economia na banda de transmissão nos servidores e ganhos em escalabilidade. Entretanto, comprovar efetivamente os ganhos em sistemas reais é um desafio importante e que poucos trabalhos têm dado atenção na literatura. A proposta desse artigo é analisar dois sistemas de*

Flávio Miguel Varejão

INTRODUÇÃO À PROGRAMAÇÃO:
UMA NOVA ABORDAGEM USANDO C

gen LTC

Exemplo: atividades durante a minha graduação (e as portas que elas abriram)

- **NOTA:** como aluno da UFES você pode fazer a diferença e impactar a universidade e a sociedade

Exemplo: atividades durante a minha graduação (e as portas que elas abriram)

- **NOTA:** como aluno da UFES você pode fazer a diferença e impactar a universidade e a sociedade

- Exemplos de projetos **idealizados por alunos** e executados com apoio de professores do DI/DEL (da minha epoca de UFES):

  - IntroComp

  - Nucleo de Cidadania Digital (NCD)

  - Equipe de Robotica da UFES (ERUS)
    - TRUFES

Conselhos gratuitos

1. “Todo mundo sobreviveu”:
   - Obs: Saúde (física e mental) em 1º lugar.
   - Todos bem!
   - Não desista: forme-se! Vale a pena. ☺

2. “O tempo passa pra todo mundo”:
   - Aproveite as oportunidades durante a graduação.

Conselhos gratuitos

1. “Todo mundo sobreviveu”:
   - Obs: Saúde (física e mental) em 1º lugar.
   - Todos bem!
   - Não desista: forme-se! Vale a pena. ☺

2. “O tempo passa pra todo mundo”:
   - Aproveite as oportunidades durante a graduação.

3. “Nunca fiz nada tão difícil quanto me formar na UFES”:
   - ***<u>Existe vida após a graduação.</u>***
   - Depois de formar a vida fica bem mais fácil.

Conselhos gratuitos

1. “Todo mundo sobreviveu”:
   - Obs: Saúde (física e mental) em 1º lugar.
   - Todos bem!
   - Não desista: forme-se! Vale a pena. ☺

2. “O tempo passa pra todo mundo”:
   - Aproveite as oportunidades durante a graduação.

3. “Nunca fiz nada tão difícil quanto me formar na UFES”:
   - ***<u>Existe vida após a graduação.</u>***
   - Depois de formar a vida fica bem mais fácil.

4. “Se eu não tivesse amigo, eu não tinha formado”:

Conselhos gratuitos

1. “Todo mundo sobreviveu”:
   - Obs: Saúde (física e mental) em 1º lugar.
   - Todos bem!
   - Não desista: forme-se! Vale a pena. ☺

2. “O tempo passa pra todo mundo”:
   - Aproveite as oportunidades durante a graduação.

3. “Nunca fiz nada tão difícil quanto me formar na UFES”:
   - ***<u>Existe vida após a graduação.</u>***
   - Depois de formar a vida fica bem mais fácil.

4. “Se eu não tivesse amigo, eu não tinha formado”:
   - Faça amigos: sua rede de conexões profissionais começa agora, na UFES.
   - Meus colegas de UFES sao amigos que eu levo para a vida inteira.

“Se eu não tivesse amigo, eu não tinha formado”

PASSAR EM CÁLCULO E ÁLGEBRA NO MESMO PERÍODO É ALGO QUE POUCOS CONSEGUIRAM
EU CONSEGUI NA 4ª TENTATIVA!

Depois da UFES

- Mestrado na UFMG (2014-2016)
- Doutorado – University of California Irvine (2016-2021)
- Atualmente:
    - Professor – Rochester Institute of Technology (RIT)

Parte 2: Oportunidades no RIT

ABOUT RIT

Our Story

- Private University
- Founded in 1829
- 10th largest private university in the U.S.
- 9 colleges, 18+ research centers
- 50+ MOU's and Partnerships.
- Campuses in Rochester, Croatia, Dubai & Kosovo

Student Body

- 19,000+ students
  - 15,900 undergraduate
  - 3,100 graduate
  - ~15% international students
- 118,000+ alumni

RIT | Rochester Institute of Technology

AACSB ACCREDITED

Accredited ABET Engineering Accreditation Commission

BEST GRAD SCHOOLS U.S.News & WORLD REPORT RANKINGS

FEATURED IN 2016 FISKE GUIDE TO COLLEGES

THE PRINCETON REVIEW 2016 THE 380 BEST COLLEGES

ABOUT ROCHESTER

3rd largest city in New York State

1.1M
total population

🚗

Buffalo = 1 hour

Niagara Falls = 1.5 hours

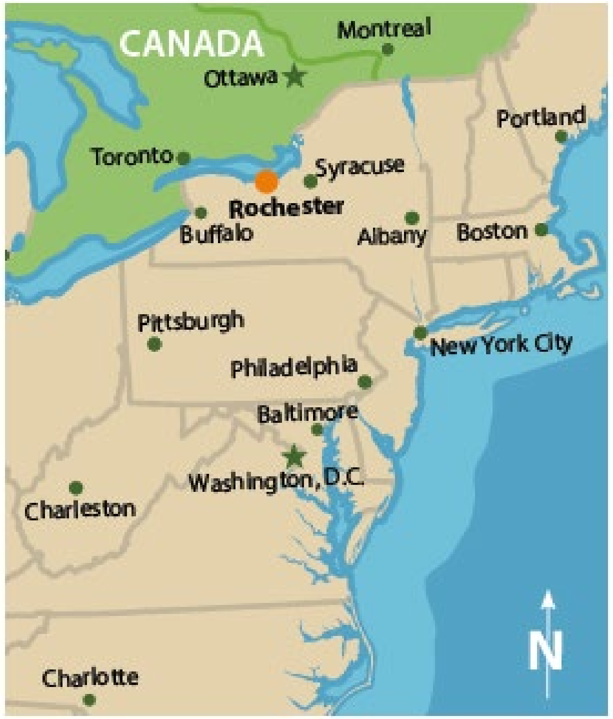
Toronto, Canada = 3 hours

✈

New York City = 1 hour

Boston = 1.5 hours

Chicago = 1.5 hours



RIT Rochester Institute of Technology

Global Cybersecurity Institute (GCI)

Cybersecurity is a wholistic outcome and is a multidisciplinary activity

- Computing Security is a core technical discipline but successful outcomes demands integration and collaboration across a broad range of disciplines
- Software engineering, computer science, HCI, gaming, business, cognitive psychology, public policy, mathematics, quantum computing etc.

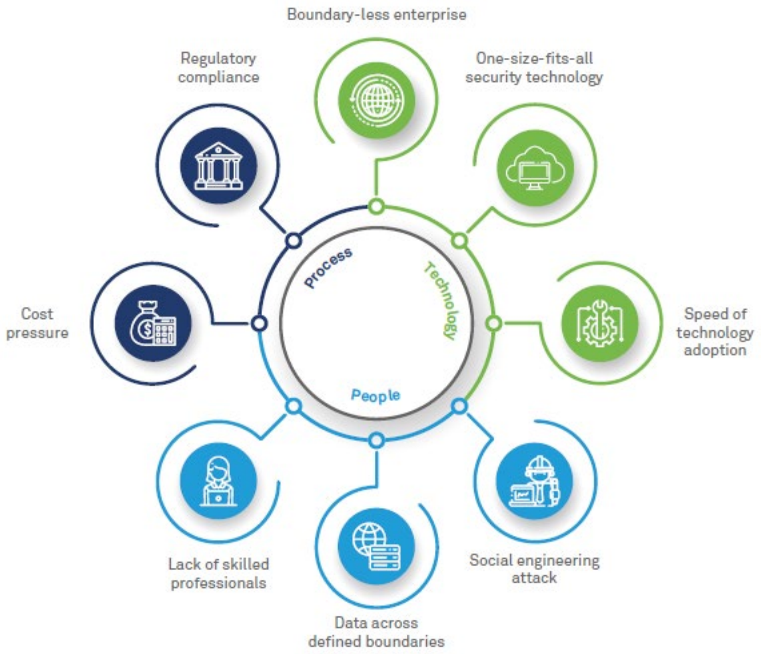

Boundary-less enterprise
Regulatory compliance
One-size-fits-all security technology
Cost pressure
Speed of technology adoption
Process
Technology
People
Lack of skilled professionals
Social engineering attack
Data across defined boundaries

Image courtesy Wipro
https://www.wipro.com/en-US/applications/eliminating-the-complexity-in-cybersecurity-with-artificial-intelligence/

Capitalize on existing strengths in education, research and outreach/impact by taking them to the next level with focus and intensity

- 500 students, leader in Collegiate Cyber Competitions
- $3M in yearly research grants & growing
- Eaton SAFE lab for penetration testing

GCI – Cybersecurity as a Global Endeavor

Goals: Experience, Expertise, Facility, & Opportunity.

Countries with Partner Institutions: United Kingdom, Czech Republic, Poland, Ireland, Netherlands, Italy, France, Germany, Ukraine, Taiwan, India, South Korea, Uruguay, Mexico, Brazil, and counting.

Activities:

- **CyberVSR**: Visiting students conducting research with GCI faculty in a culturally diverse environment.

https://www.rit.edu/cybersecurity/cybervsr

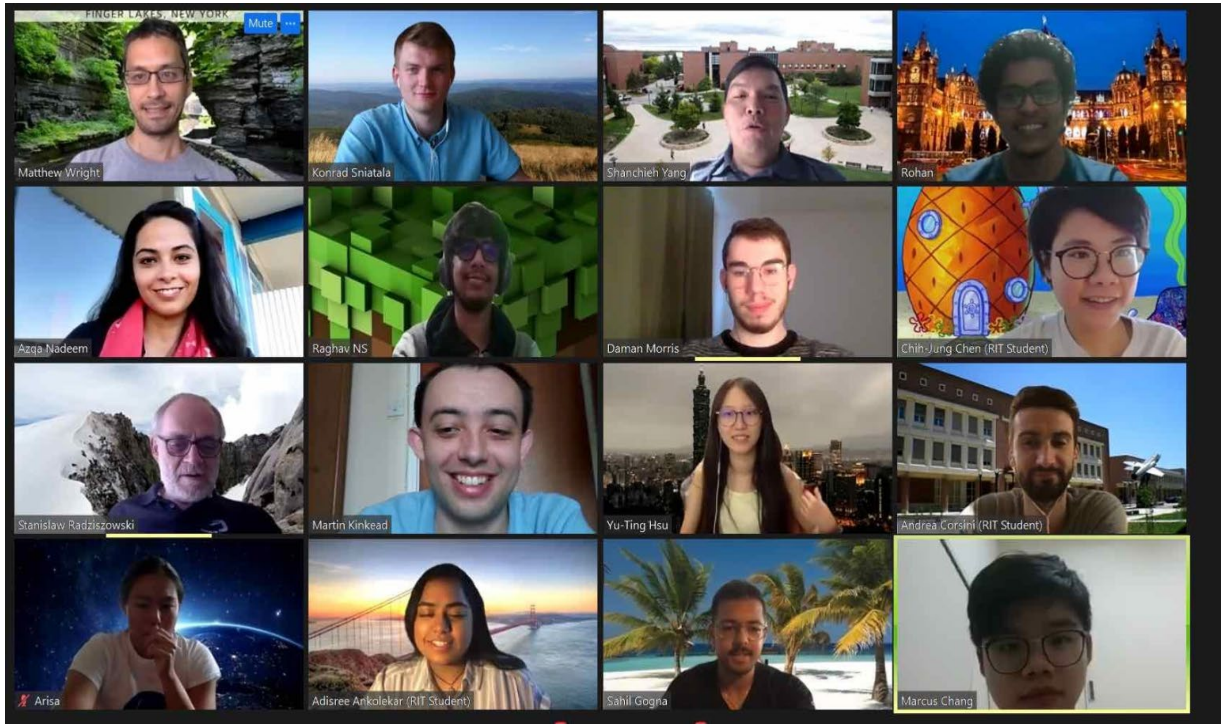
GCI – Cybersecurity as a Global Endeavor

Goals: Experience, Expertise, Facility, & Opportunity.

Countries with Partner Institutions: United Kingdom, Czech Republic, Poland, Ireland, Netherlands, Italy, France, Germany, Ukraine, Taiwan, India, South Korea, Uruguay, Mexico, Brazil, and counting.

Activities:

- **CyberVSR**: Visiting students conducting research with GCI faculty in a culturally diverse environment.
- **Joint student supervision**: undergraduate and graduate.
- **Collaborative grants**: CSIT @ Queen's University Belfast (UK), Poznan University of Technology / EUNICE (Poland), KPI University (Ukraine), & Gachon University (S. Korea).
- **Joint webinars and workshops**:
  - US-NI-RoI Workshop on IoT/CPS Cybersecurity
  - NATO AICA Conference
- **CPTC International**: RIT Dubai (Middle East), SiberX & Durham College (Canada), & Masaryk University (Europe).
- **Collaborative Training/Education**: ... in the works,



Global Cybersecurity Institute Virtual Tour

https://youtu.be/XdnRwwxcR7Y

The New World We Live In

A Global Digital Nervous System

The physical is digital and computers make autonomous decisions

Connected everywhere

**Greater access,
but less control**

**New technologies =
new vulnerabilities**

The Old Model of Security

Perimeter-based

A single layer or simply add more layers

Static, inflexible

Can we create a cyber immune system?

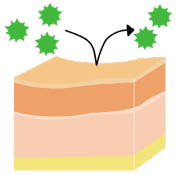
Assume constant attack

Innate detection and defenses

Both atomic and wholistic

Highly adaptive

The Immune System Metaphor

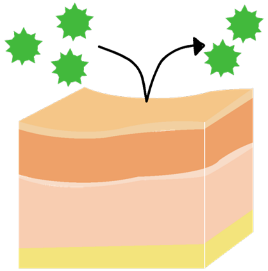


Barriers: Skin and cilia prevent invaders from entering

Innate: Fever, chemicals stop invaders from spreading

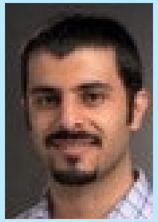
Adaptive: White blood cells attack invaders

Barriers: Stopping Airborne Attacks



Wireless Security

- Full-frame Encryption
- Physical-layer attributes



Hanif Rahbari

Robust & Secure System-on-a-Chip

- Jamming protection
- Eavesdropping protection

Amlan Ganguly

Innate: Security by Design

Combatting Architectural Weaknesses

- Finding & characterizing design flaws
- Working w/ MITRE's CWE

CWE

Mehdi Mirakhorli

Metrics for Software Vulnerabilities

- Understanding how they happen
- Better software patterns

Andy Meneely

Adaptive: Robust Detection

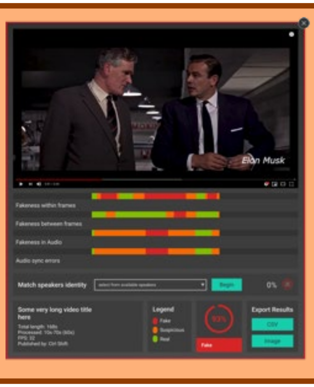
Attack Prediction & Modeling

- ML to extract adversary behavior
- Predictive modeling of attacks

S. Jay Yang

Adversarial ML

- More secure ML
- Deepfake detection



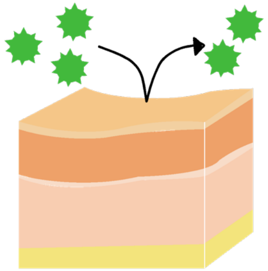
Some very long video title here

Total length: 168s
Processed: 10s-70s (60s)
FPS: 32
Published by: CH Shift

Matt Wright

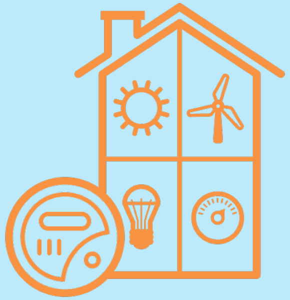
Adaptive Barriers: Cryptography

+



Privacy in Smart Meters

- Protect your activities
- Accurate, real-time data to providers



Sumita Mishra

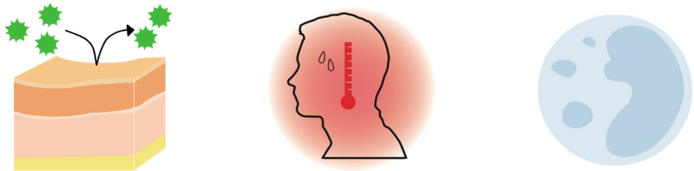
Encrypted Cloud

- Homomorphic Encryption
- Secure Analytics

Peizhao Hu

Cybersecurity Research @ GCI

Protecting our Digital System

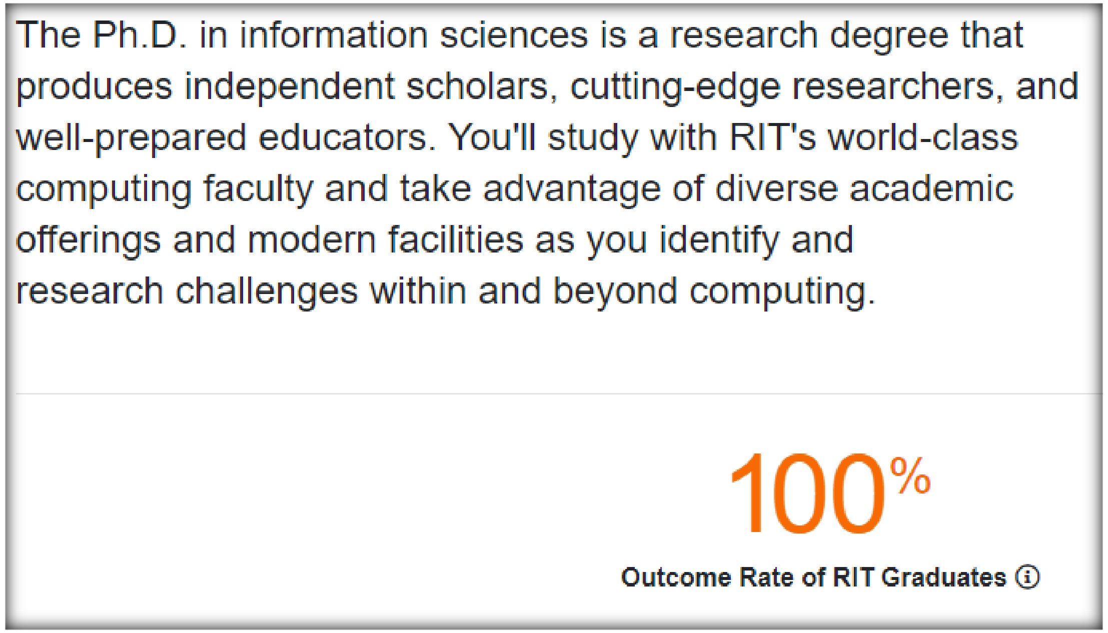


Intelligent and adaptive

Both atomic and wholistic

Providing innate protection

Ph.D. – Computing & Information Sciences

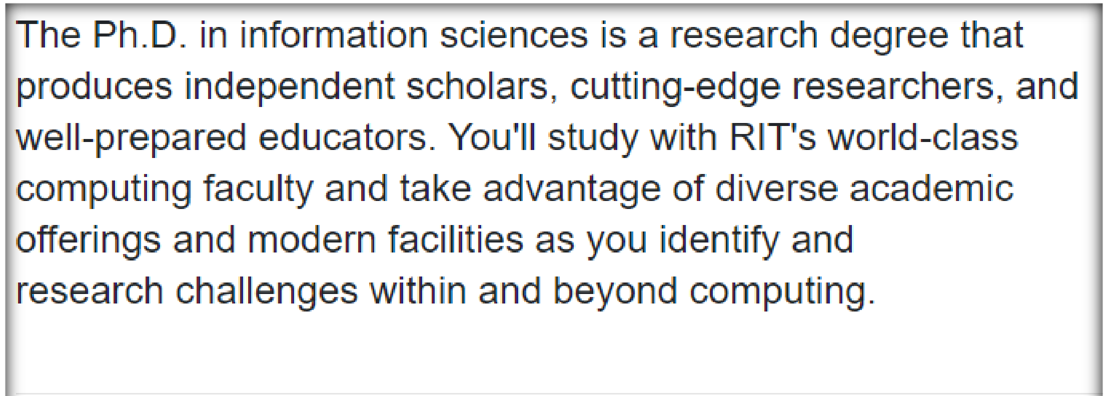


The Ph.D. in information sciences is a research degree that produces independent scholars, cutting-edge researchers, and well-prepared educators. You'll study with RIT's world-class computing faculty and take advantage of diverse academic offerings and modern facilities as you identify and research challenges within and beyond computing.

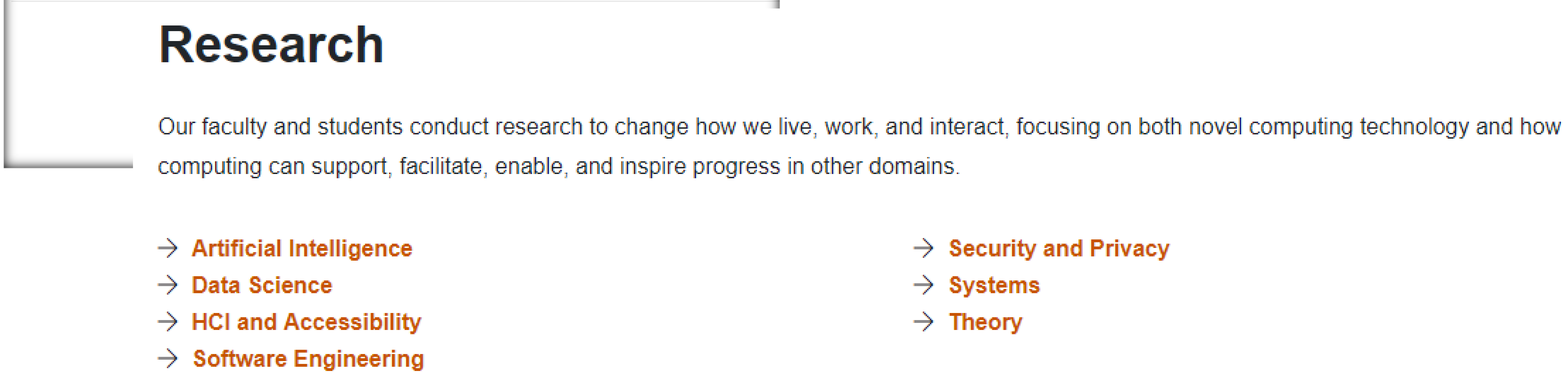
100%

Outcome Rate of RIT Graduates ⓘ

Ph.D. – Computing & Information Sciences



The Ph.D. in information sciences is a research degree that produces independent scholars, cutting-edge researchers, and well-prepared educators. You'll study with RIT's world-class computing faculty and take advantage of diverse academic offerings and modern facilities as you identify and research challenges within and beyond computing.

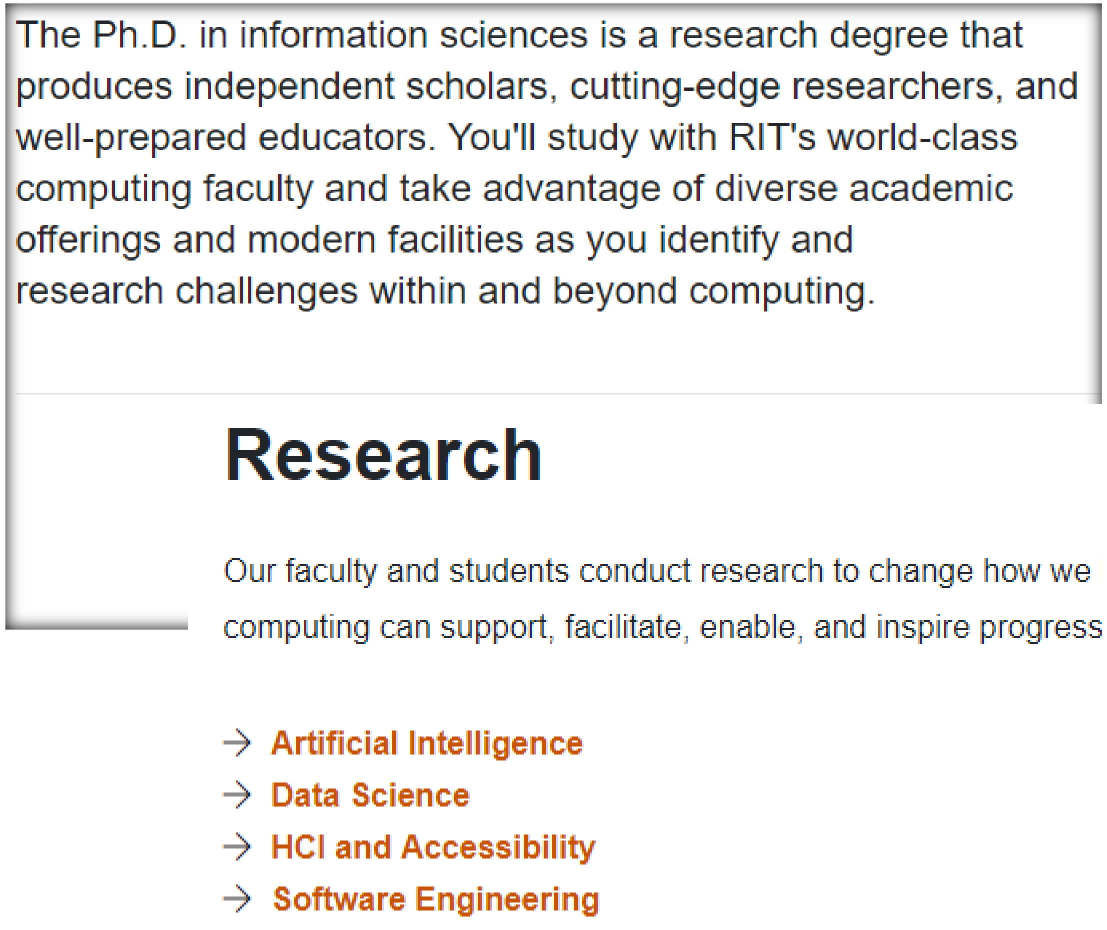


Research

Our faculty and students conduct research to change how we live, work, and interact, focusing on both novel computing technology and how computing can support, facilitate, enable, and inspire progress in other domains.

- → **Artificial Intelligence**
- → **Data Science**
- → **HCI and Accessibility**
- → **Software Engineering**
- → **Security and Privacy**
- → **Systems**
- → **Theory**

Ph.D. – Computing & Information Sciences

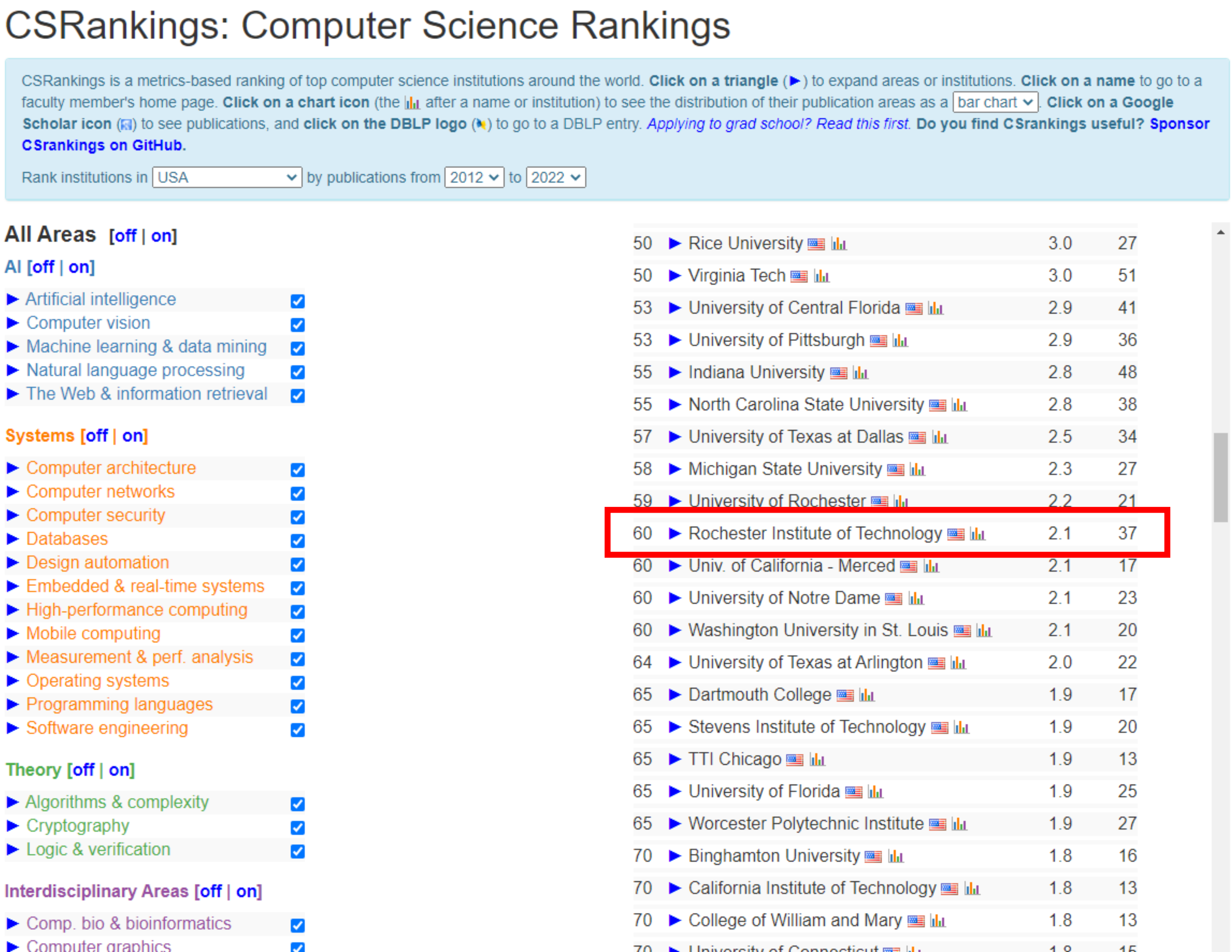


The Ph.D. in information sciences is a research degree that produces independent scholars, cutting-edge researchers, and well-prepared educators. You'll study with RIT's world-class computing faculty and take advantage of diverse academic offerings and modern facilities as you identify and research challenges within and beyond computing.

Research

Our faculty and students conduct research to change how we computing can support, facilitate, enable, and inspire progress

- → Artificial Intelligence
- → Data Science
- → HCI and Accessibility
- → Software Engineering



CSRankings: Computer Science Rankings

CSRankings is a metrics-based ranking of top computer science institutions around the world. **Click on a triangle** (▶) to expand areas or institutions. **Click on a name** to go to a faculty member's home page. **Click on a chart icon** (the 📊 after a name or institution) to see the distribution of their publication areas as a bar chart. **Click on a Google Scholar icon** to see publications, and **click on the DBLP logo** to go to a DBLP entry. *Applying to grad school? Read this first.* **Do you find CSrankings useful? Sponsor CSrankings on GitHub.**

Rank institutions in USA by publications from 2012 to 2022

All Areas [off | on]

AI [off | on]

- ▶ Artificial intelligence ☑
- ▶ Computer vision ☑
- ▶ Machine learning & data mining ☑
- ▶ Natural language processing ☑
- ▶ The Web & information retrieval ☑

Systems [off | on]

- ▶ Computer architecture ☑
- ▶ Computer networks ☑
- ▶ Computer security ☑
- ▶ Databases ☑
- ▶ Design automation ☑
- ▶ Embedded & real-time systems ☑
- ▶ High-performance computing ☑
- ▶ Mobile computing ☑
- ▶ Measurement & perf. analysis ☑
- ▶ Operating systems ☑
- ▶ Programming languages ☑
- ▶ Software engineering ☑

Theory [off | on]

- ▶ Algorithms & complexity ☑
- ▶ Cryptography ☑
- ▶ Logic & verification ☑

Interdisciplinary Areas [off | on]

- ▶ Comp. bio & bioinformatics ☑
- ▶ Computer graphics ☑

50	▶ Rice University	3.0	27
50	▶ Virginia Tech	3.0	51
53	▶ University of Central Florida	2.9	41
53	▶ University of Pittsburgh	2.9	36
55	▶ Indiana University	2.8	48
55	▶ North Carolina State University	2.8	38
57	▶ University of Texas at Dallas	2.5	34
58	▶ Michigan State University	2.3	27
59	▶ University of Rochester	2.2	21
60	▶ Rochester Institute of Technology	2.1	37
60	▶ Univ. of California - Merced	2.1	17
60	▶ University of Notre Dame	2.1	23
60	▶ Washington University in St. Louis	2.1	20
64	▶ University of Texas at Arlington	2.0	22
65	▶ Dartmouth College	1.9	17
65	▶ Stevens Institute of Technology	1.9	20
65	▶ TTI Chicago	1.9	13
65	▶ University of Florida	1.9	25
65	▶ Worcester Polytechnic Institute	1.9	27
70	▶ Binghamton University	1.8	16
70	▶ California Institute of Technology	1.8	13
70	▶ College of William and Mary	1.8	13
70	▶ University of Connecticut	1.8	15

Parte 3:

Um pouco sobre a minha pesquisa

IoT Device Security

What is an IoT device?

Loosely specified:

“It’s a thing” ...

AND

“It’s in the Internet (i.e., can communicate)” ...

=>

“It’s an IoT Device!”

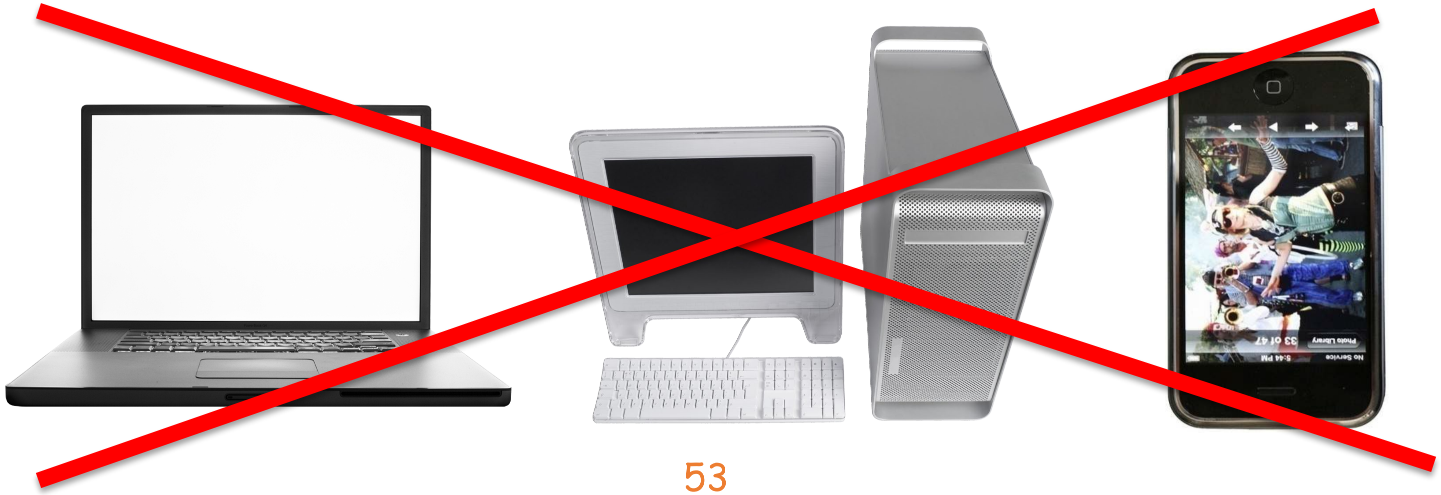
Not wrong, but too broad => Not very useful as a definition.

What is an IoT device?

Our context:

IoT devices have limitations when compared to your everyday general purpose devices.

(In our context) the following **general purpose** computers are ***not considered*** “IoT Devices”:

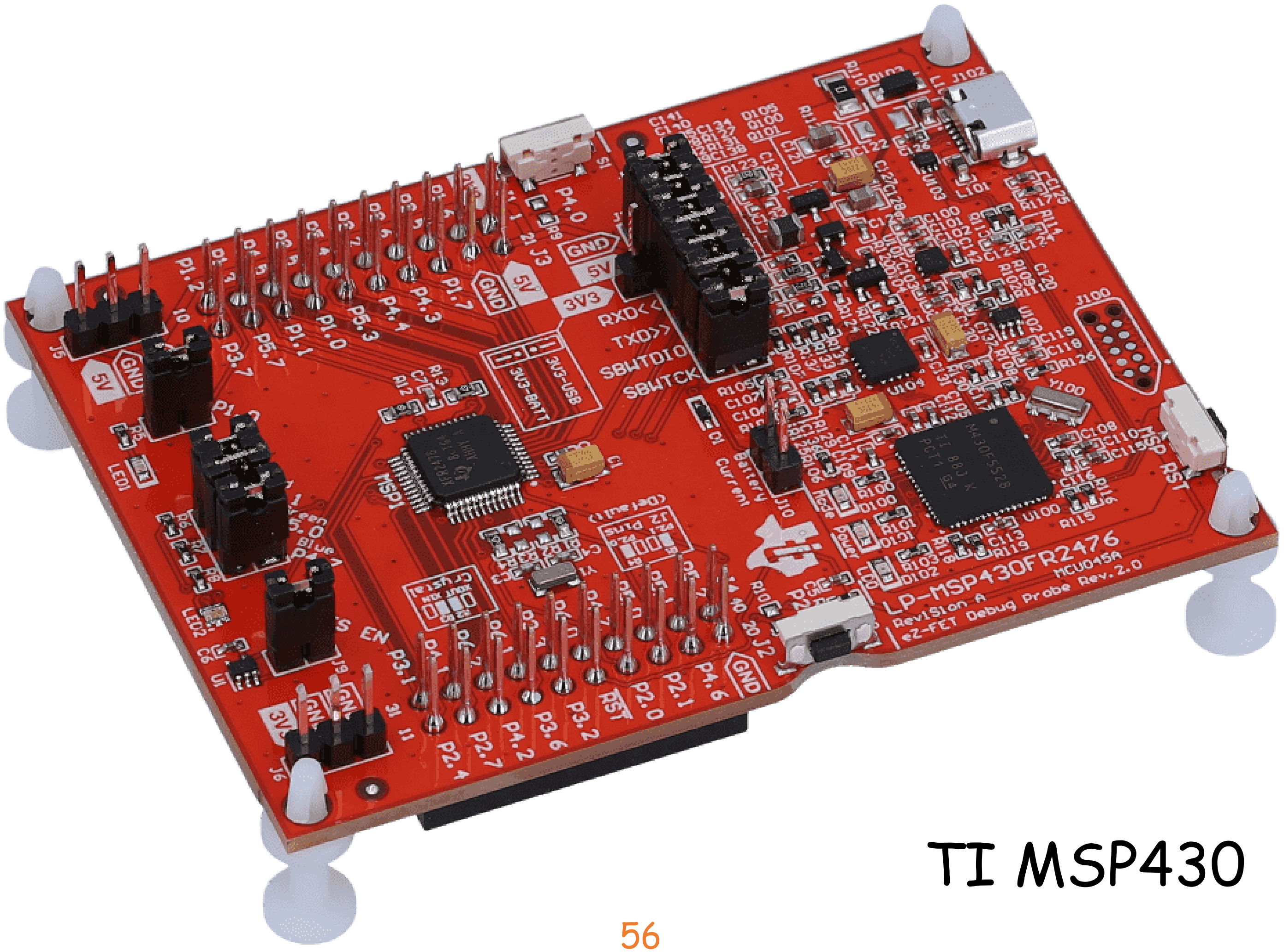


Wide range of Specialized Embedded Devices

Wide range of Specialized Embedded Devices

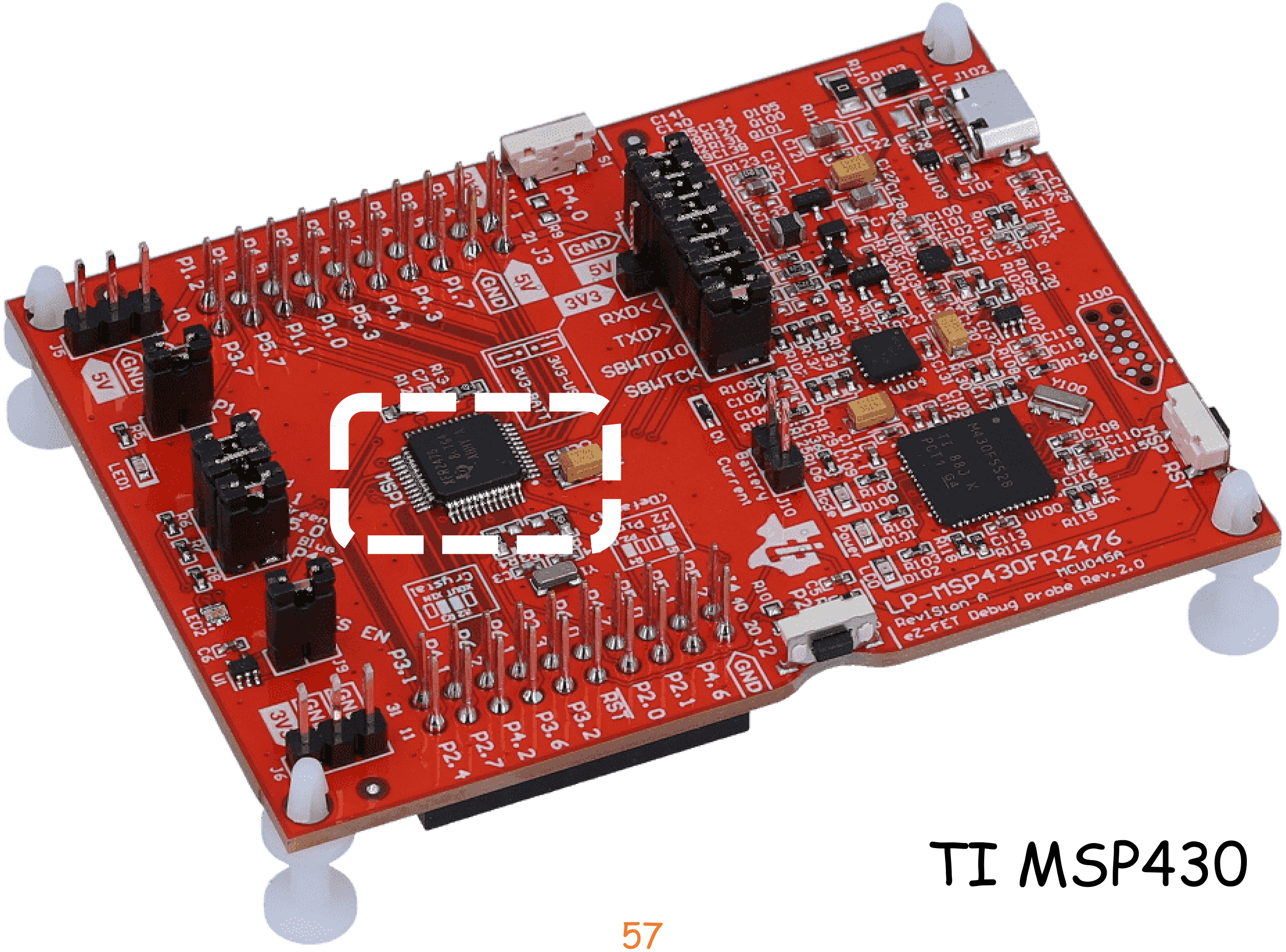
Usually implemented using Micro-Controller Units (MCUs)

Micro-Controller Unit (MCU)



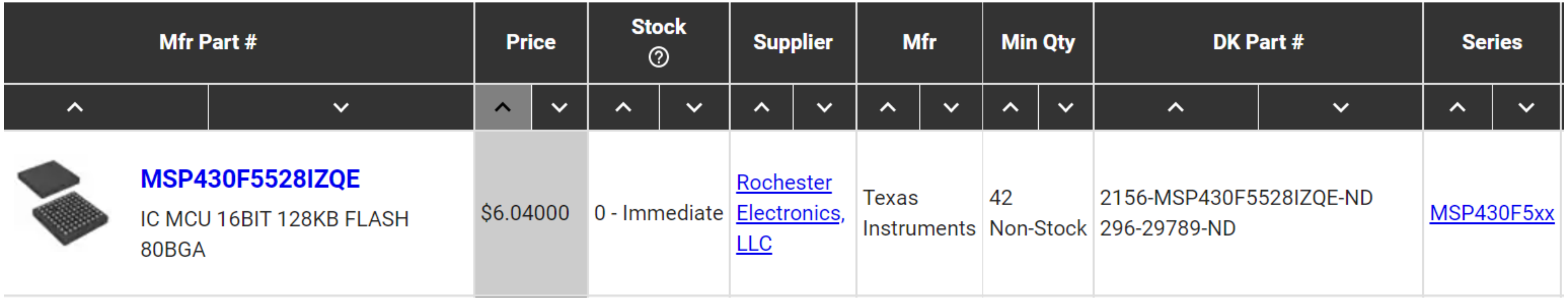
TI MSP430

Micro-Controller Unit (MCU)



TI MSP430

Micro-Controller Unit (MCU)



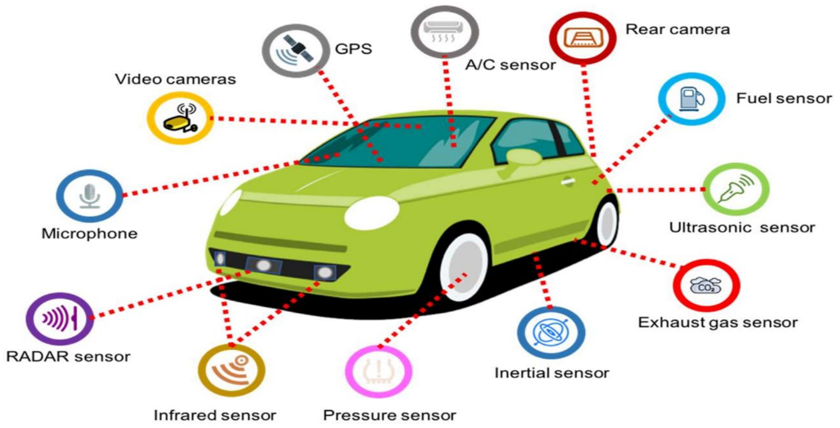
Mfr Part #	Price	Stock ?	Supplier	Mfr	Min Qty	DK Part #	Series
⌃ ⌄	⌃ ⌄	⌃ ⌄	⌃ ⌄	⌃ ⌄	⌃ ⌄	⌃ ⌄	⌃ ⌄
MSP430F5528IZQE IC MCU 16BIT 128KB FLASH 80BGA	$6.04000	0 - Immediate	Rochester Electronics, LLC	Texas Instruments	42 Non-Stock	2156-MSP430F5528IZQE-ND 296-29789-ND	MSP430F5xx

IoT & MCU Security

(why bother?)

IoT Applications

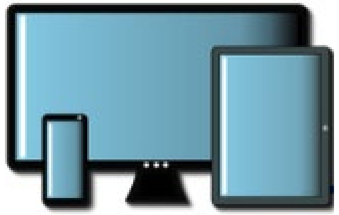
- Multitudes of interconnected devices
  - Control units
  - Sensors
  - Actuators
  - Network devices
- Examples
  - Industrial/office automation
  - Home automation
  - Vehicles
- Heterogeneous:
  <u>Typically, more sophisticated devices control simpler lower-end ones</u>



Safety Critical Embedded/Cyber-physical/IoT Systems

- Examples
  - Smoke detector in a household
  - Engine temperature sensor in a car

Controller
(Higher-end device)

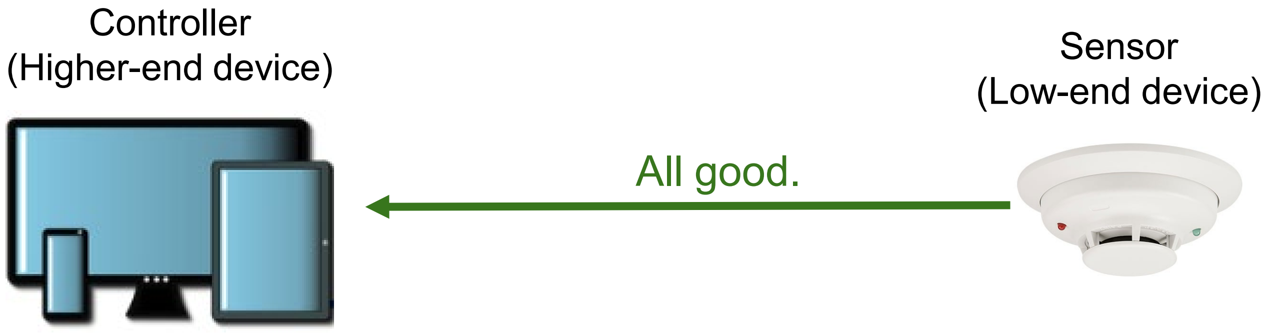


Sensor
(Low-end device)

Controllers rely on sensed values to make decisions (e.g., send help)

Safety Critical Embedded/Cyber-physical/IoT Systems

- Examples
  - Smoke detector in a household
  - Engine temperature sensor in a car



Controller
(Higher-end device)

All good.

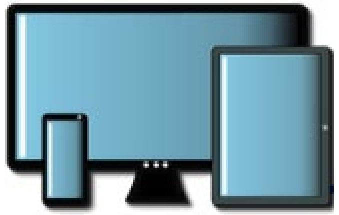
Sensor
(Low-end device)

<u>Controllers rely on sensed values to make decisions (e.g., send help)</u>

Safety Critical Embedded/Cyber-physical/IoT Systems

- Examples
  - Smoke detector in a household
  - Engine temperature sensor in a car

Controller
(Higher-end device)

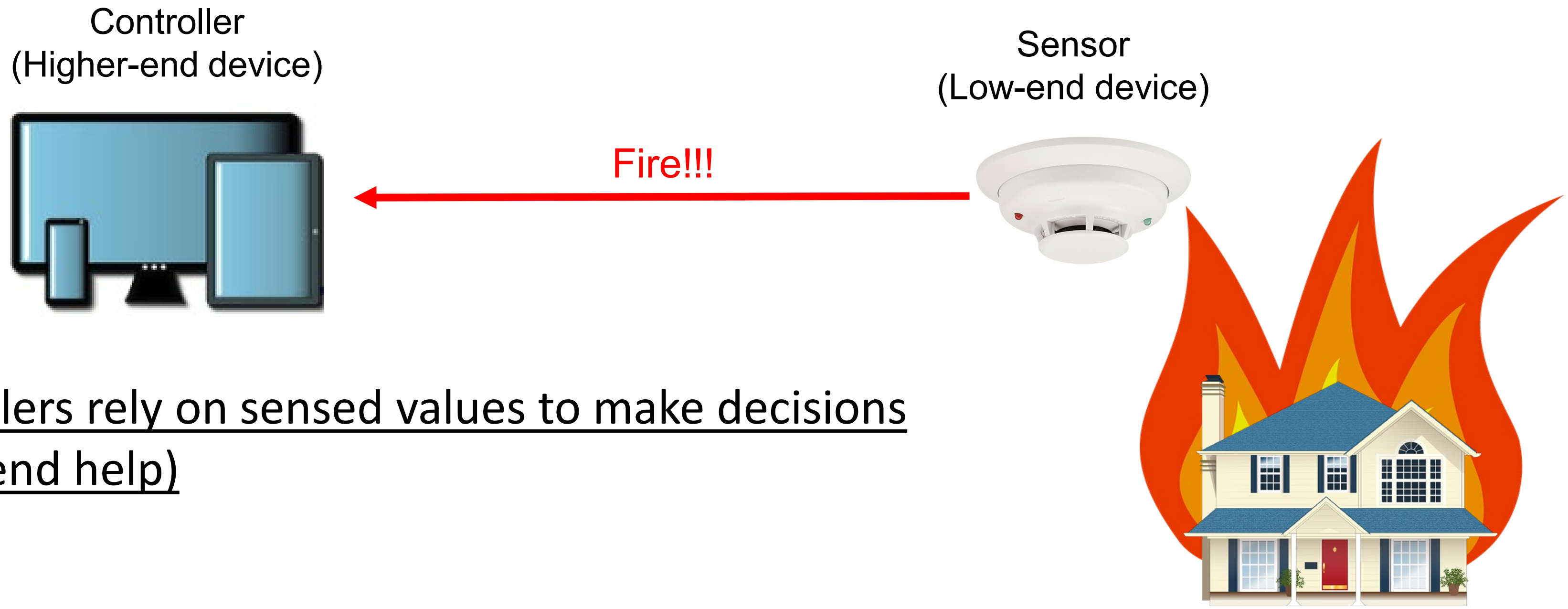


Sensor
(Low-end device)

Controllers rely on sensed values to make decisions (e.g., send help)

Safety Critical Embedded/Cyber-physical/IoT Systems

- Examples
  - Smoke detector in a household
  - Engine temperature sensor in a car

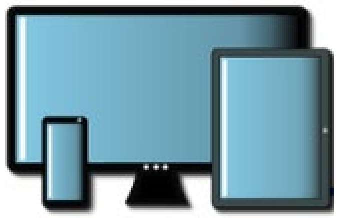


Controllers rely on sensed values to make decisions (e.g., send help)

Safety Critical Embedded/Cyber-physical/IoT Systems

- Examples
  - Smoke detector in a household
  - Engine temperature sensor in a car

Controller
(Higher-end device)



Sensor
(Low-end device)

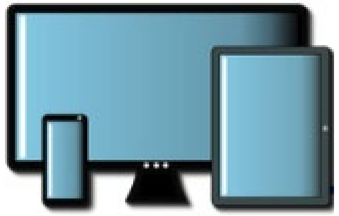
Infected
Sensor

<u>Problem:</u> <u>compromised software on the low-end sensor device might spoof sensed values</u>

Safety Critical Embedded/Cyber-physical/IoT Systems

- Examples
  - Smoke detector in a household
  - Engine temperature sensor in a car

Controller
(Higher-end device)



Sensor
(Low-end device)

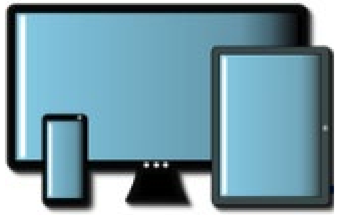
Infected
Sensor

Problem: compromised software on the low-end sensor device might spoof sensed values

Safety Critical Embedded/Cyber-physical/IoT Systems

- Examples
  - Smoke detector in a household
  - Engine temperature sensor in a car

Controller
(Higher-end device)



All good.

Sensor
(Low-end device)

Infected
Sensor

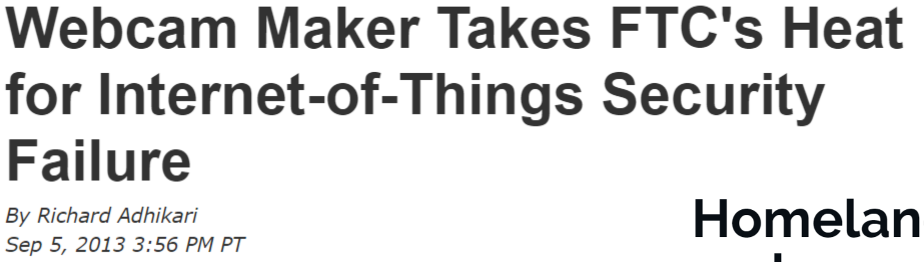
Problem: compromised software on the low-end sensor device might spoof sensed values

Safety Critical Embedded/Cyber-physical/IoT Systems

- Other examples:

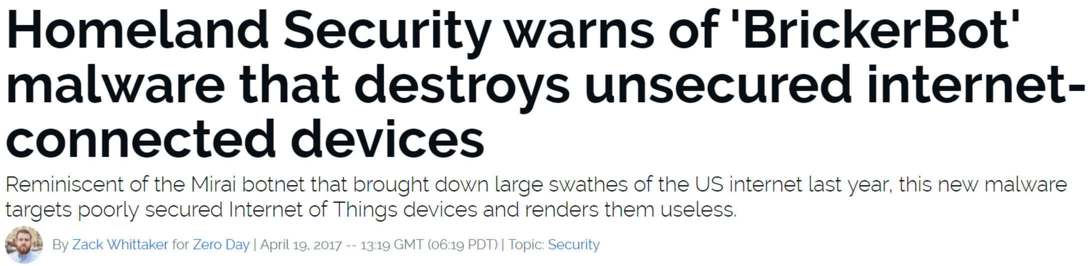
  - Implantable (battery powered) medical devices
  - Enviromental/chemical sensors in the rainforest (or underwater)
  - Energy meter or a household (for billing purposes and more)

IoT Attacks in the Wild



Webcam Maker Takes FTC's Heat for Internet-of-Things Security Failure

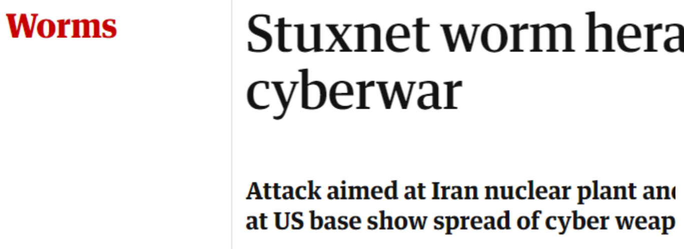
By Richard Adhikari
Sep 5, 2013 3:56 PM PT



Homeland Security warns of 'BrickerBot' malware that destroys unsecured internet-connected devices

Reminiscent of the Mirai botnet that brought down large swathes of the US internet last year, this new malware targets poorly secured Internet of Things devices and renders them useless.

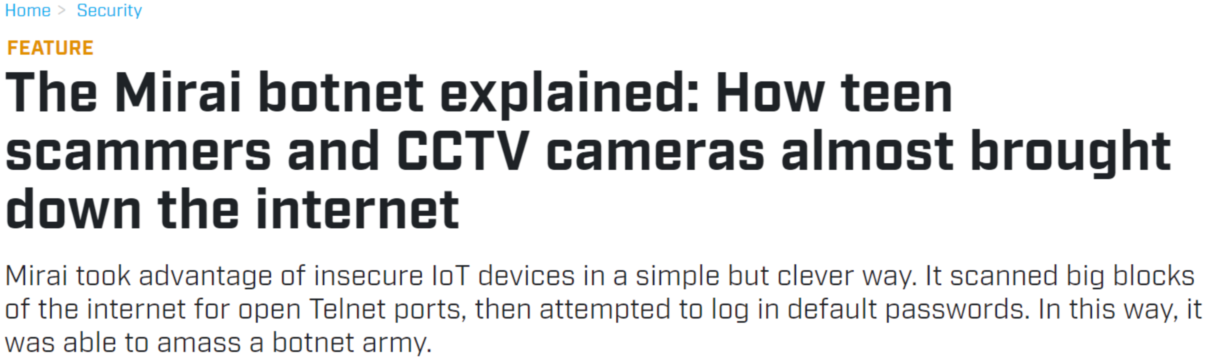
By Zack Whittaker for Zero Day | April 19, 2017 -- 13:19 GMT (06:19 PDT) | Topic: Security



Worms

Stuxnet worm heralds cyberwar

**Attack aimed at Iran nuclear plant an
at US base show spread of cyber weap**



Home > Security

FEATURE

The Mirai botnet explained: How teen scammers and CCTV cameras almost brought down the internet

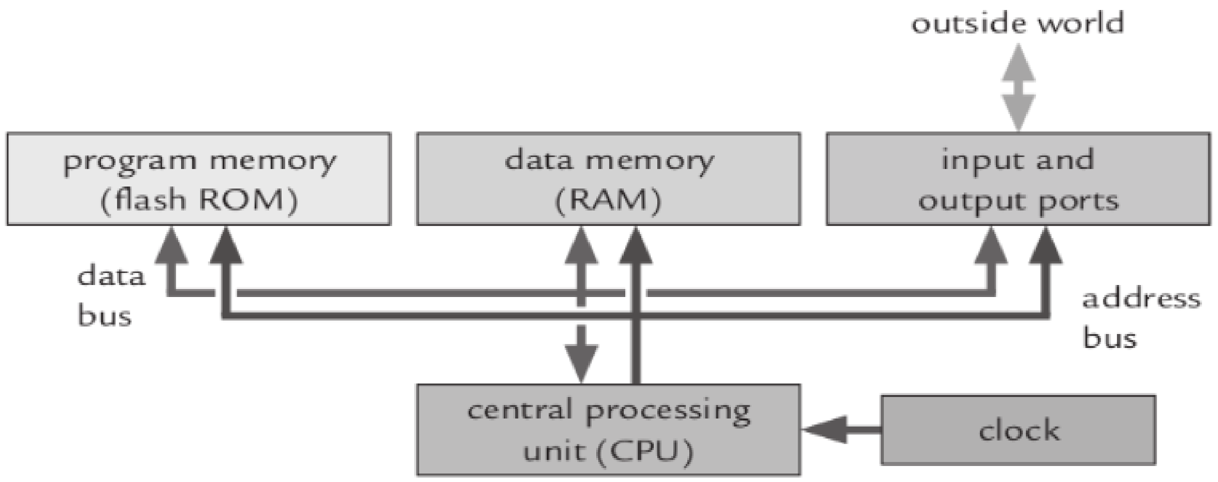
Mirai took advantage of insecure IoT devices in a simple but clever way. It scanned big blocks of the internet for open Telnet ports, then attempted to log in default passwords. In this way, it was able to amass a botnet army.

IoT-Specific Threats and Attacker Goals

- **Sensors:** Privacy
- **Actuators:** Security/Safety (e.g., Stuxnet)
- **Either:** DDoS, a.k.a., Zombification (e.g., Mirai)

And combinations thereof...

MCU Computational Resources
(The amoebas of the computing world)



- Designed for: **<u>Low-Cost</u>**, **<u>Low-Energy</u>**, **<u>Small-Size.</u>**
- Memory: Program (32 to 64kB) and Data (2 to 16 kB)
- Single core CPU (1 to 16MHz; 8 or 16 bits)
- Simple Communication Interfaces for IO (a Few kbps)
- Examples: TI MSP430, AVR ATMega32 (Arduino)

Other IoT Security Issues & Challenges

- Default PINs/Passwords (MIRAI BotNet)

Other IoT Security Issues & Challenges

- Default PINs/Passwords (MIRAI BotNet)
- **Hard to access and deployed in large numbers (Sensor Networks, PLC networks)**
  - may require remote operation and verification

Other IoT Security Issues & Challenges

- Default PINs/Passwords (MIRAI BotNet)
- **Hard to access and deployed in large numbers (Sensor Networks, PLC networks)**
  - may require remote operation and verification
- **Buggy software:**
  - often written in very efficient, but unsafe languages (usually C or Assembly)
  - Why?

Other IoT Security Issues & Challenges

- Default PINs/Passwords (MIRAI BotNet)
- **Hard to access and deployed in large numbers (Sensor Networks, PLC networks)**
  - may require remote operation and verification
- **Buggy software:**
  - often written in very efficient, but unsafe languages (usually C or Assembly)
  - Why?
- **Inadequate Hardware/Architectural support for security:**

Other IoT Security Issues & Challenges

- Default PINs/Passwords (MIRAI BotNet)
- **Hard to access and deployed in large numbers (Sensor Networks, PLC networks)**
  - may require remote operation and verification
- **Buggy software:**
  - often written in very efficient, but unsafe languages (usually C or Assembly)
  - Why?
- **Inadequate Hardware/Architectural support for security:**
  - Somewhat low-end, e.g., ARM Cortex M/R processors:
    - **primitive security support (MPU, but no MMU)**
  - Lowest-end/ultra low-energy, e.g, AtMega, MSP430, etc:
    - **no security support**
  - It’s a budgetary issue!

Some Open Questions and Research Problems

How to remotely assure that an MCU:

- is currently loaded with the expected software?
  - **Code integrity**

Some Open Questions and Research Problems

How to remotely assure that an MCU:

- is currently loaded with the expected software?
  - **Code integrity**
- is guaranteed to executed an expected function/operation?
- won’t ignore commands?
  - Safety-critical actuation, software update, etc...
  - **Availability**

Some Open Questions and Research Problems

How to remotely assure that an MCU:

- is currently loaded with the expected software?
  - **Code integrity**
- is guaranteed to executed an expected function/operation?
- won’t ignore commands?
  - Safety-critical actuation, software update, etc...
  - **Availability**
- produced some data through the proper execution of the expected operation?
  - e.g., a sensing task
  - **Execution integrity**

Some Open Questions and Research Problems

How to remotely assure that an MCU:

- is currently loaded with the expected software?
  - **Code integrity**
- is guaranteed to executed an expected function/operation?
- won't ignore commands?
  - Safety-critical actuation, software update, etc...
  - **Availability**
- produced some data through the proper execution of the expected operation?
  - e.g., a sensing task
  - **Execution integrity**
- won't spy on you or leak your data?
  - **Confidentiality/Privacy**
- and so on...

Some Open Questions and Research Problems

How to remotely assure that an MCU:

- is currently loaded with the expected software?
  - **Code integrity**
- is guaranteed to executed an expected function/operation?
- won't ignore commands?
  - Safety-critical actuation, software update, etc...
  - **Availability**
- produced some data through the proper execution of the expected operation?
  - e.g., a sensing task
  - **Execution integrity**
- won't spy on you or leak your data?
  - **Confidentiality/Privacy**
- and so on...

Bonus challenge:
Make all of that secure (provably so) and affordable enough to run in a resource-constrained amoeba!

Some Open Questions and Research Problems

How to remotely assure that an MCU:

- is currently loaded with the expected software?
  - **Code integrity**

For today, let's focus on this one!

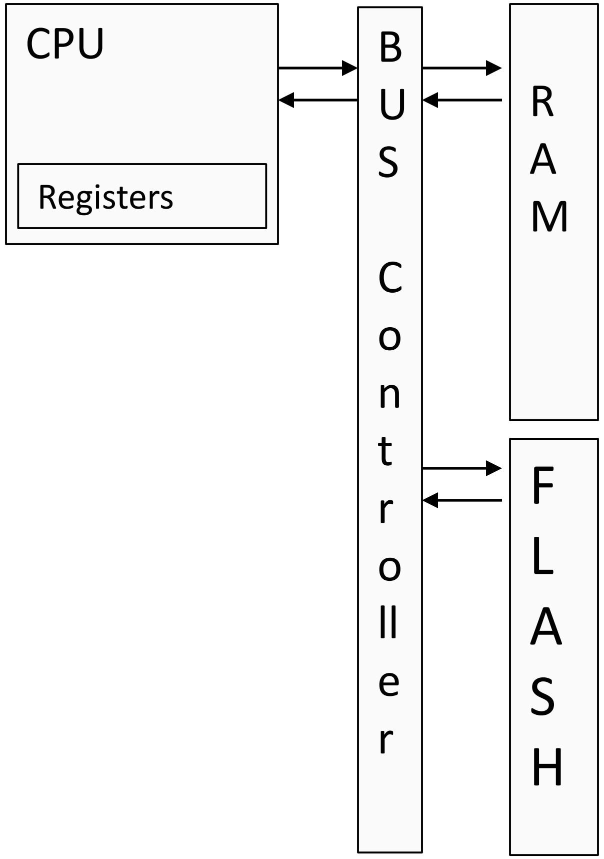
- is guaranteed to executed an expected function/operation?
- won't ignore commands?
  - Safety-critical actuation, software update, etc...
  - **Availability**
- produced some data through the proper execution of the expected operation?
  - e.g., a sensing task
  - **Execution integrity**
- won't spy on you or leak your data?
  - **Confidentiality/Privacy**
- and so on...

Bonus challenge:

Make all of that secure (provably so) and affordable enough to run in a resource-constrained amoeba!

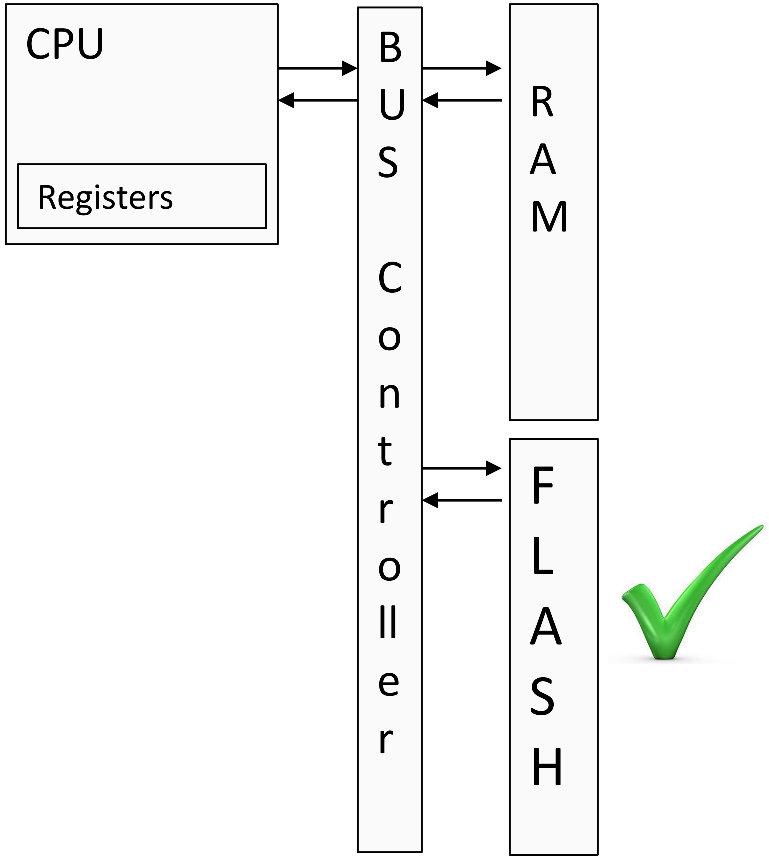
Software Integrity in IoT Devices

The Most Fundamental Question



- Without it nothing else makes sense:

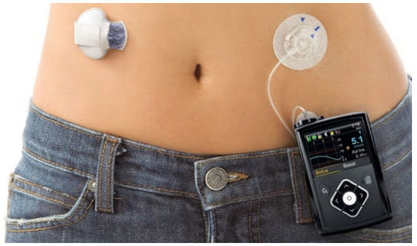
The Most Fundamental Question



- Without it nothing else makes sense:
    - ***<u>Is my IoT device currently installed with the correct/expected code?</u>***

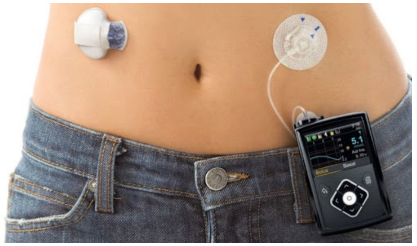
Threat Model

- What can the adversary do?



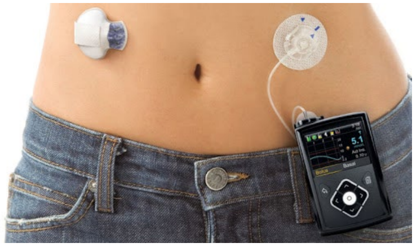
Threat Model

- What can the adversary do?
    - Access the device and re-program FLASH without the owner's knowledge or permission



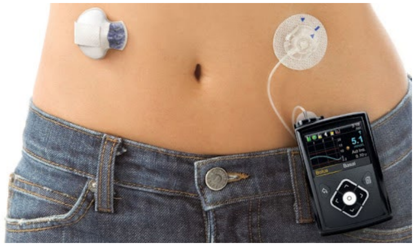
Threat Model

- What can the adversary do?
  - Access the device and re-program FLASH without the owner's knowledge or permission
  - Replace SD Card with pre-loaded malicious code



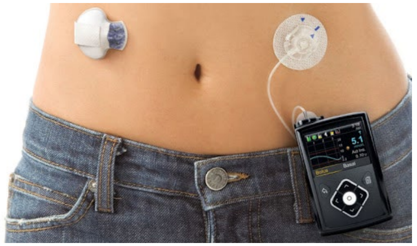
Threat Model

- What can the adversary do?
  - Access the device and re-program FLASH without the owner's knowledge or permission
  - Replace SD Card with pre-loaded malicious code
  - Example: Automated Insulin Pump
    - Change the FLASH code to never inject insulin
    - Change the FLASH code to overdose the user



Threat Model

- What can the adversary do?
    - Access the device and re-program FLASH without the owner’s knowledge or permission
    - Replace SD Card with pre-loaded malicious code
    - Example: Automated Insulin Pump
        - Change the FLASH code to never inject insulin
        - Change the FLASH code to overdose the user

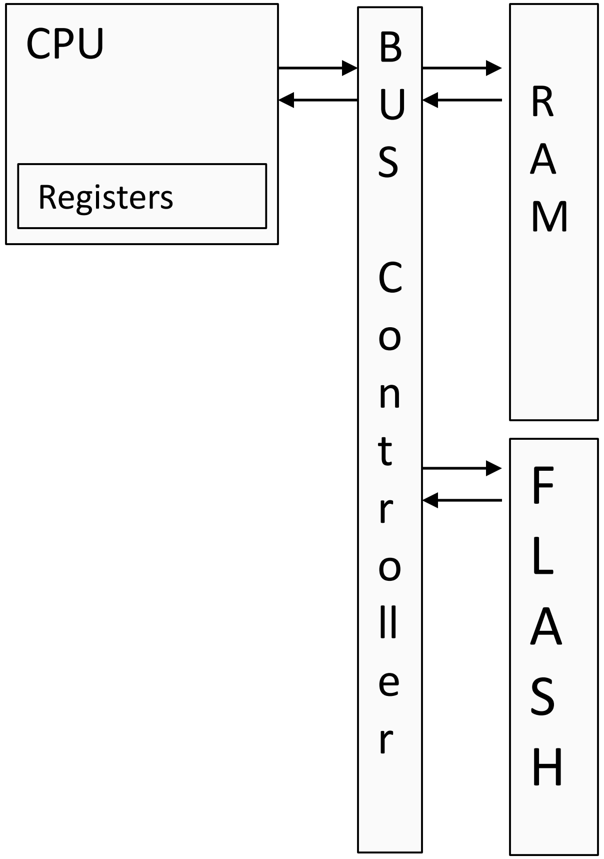


Anything modifiable can be modified by the Adversary. Hardware is not modifiable!

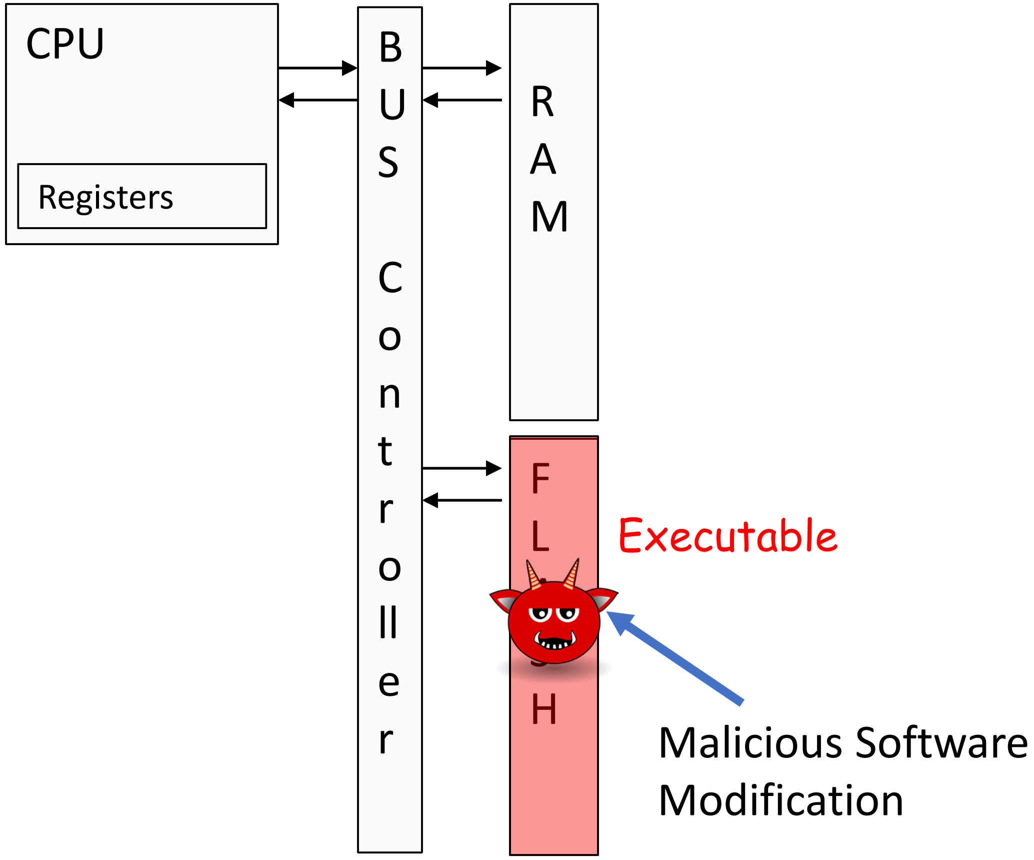
A.k.a.: Full-Software Compromise model!

One of the strongest threat models (and very applicable to IoT)

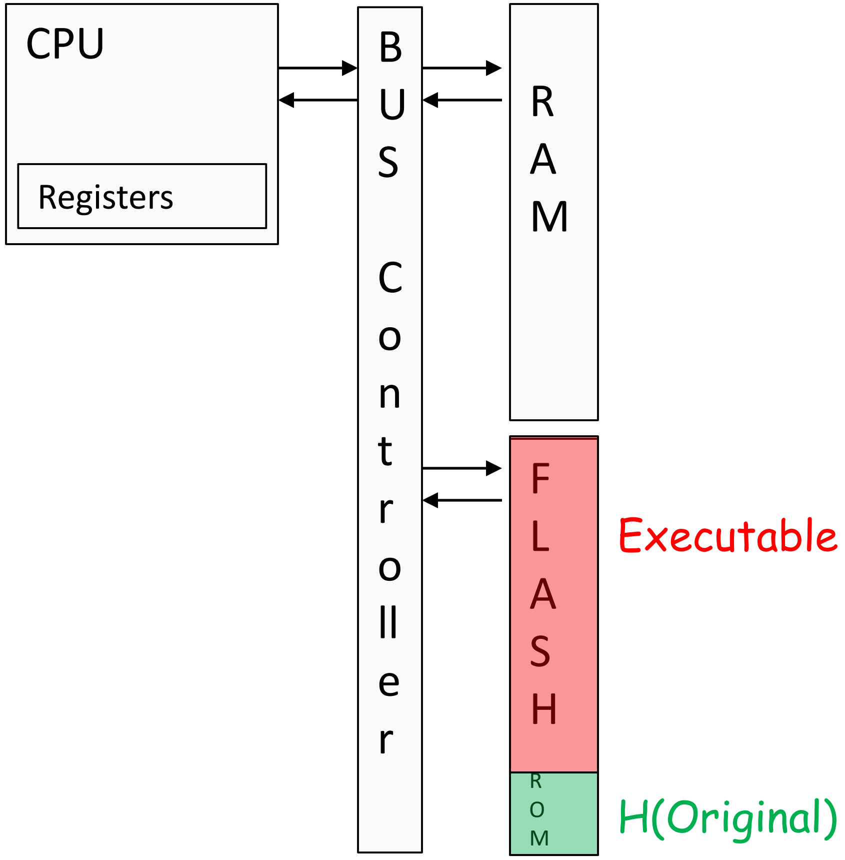
Secure Boot



Secure Boot


CPU
Registers
BUS Controller
RAM
FLASH
Executable
Malicious Software Modification

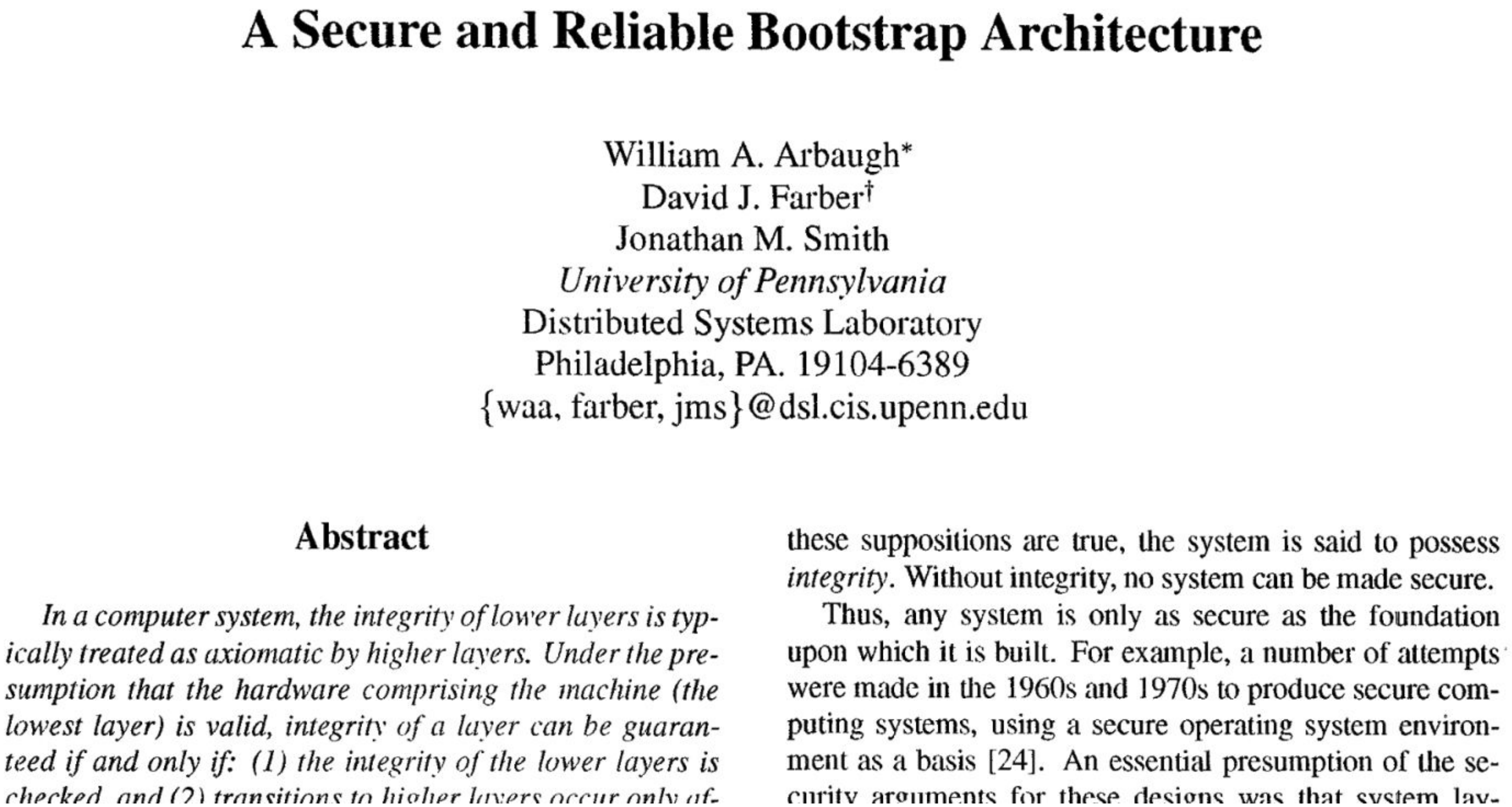
Secure Boot



- A simple idea:
  - Cryptographic Hash Functions
  - Store a hash of the Original in Read-Only Memory (ROM)
  - **At boot:** compute a hash of the executable and compare with the stored hash in ROM
    - Why does it work?

Secure Boot (history)

[IEEE S&P (Oakland) 1997]



A Secure and Reliable Bootstrap Architecture

William A. Arbaugh*
David J. Farber†
Jonathan M. Smith
University of Pennsylvania
Distributed Systems Laboratory
Philadelphia, PA. 19104-6389
{waa, farber, jms}@dsl.cis.upenn.edu

Abstract

In a computer system, the integrity of lower layers is typically treated as axiomatic by higher layers. Under the presumption that the hardware comprising the machine (the lowest layer) is valid, integrity of a layer can be guaranteed if and only if: (1) the integrity of the lower layers is checked, and (2) transitions to higher layers occur only af-

these suppositions are true, the system is said to possess *integrity*. Without integrity, no system can be made secure.

Thus, any system is only as secure as the foundation upon which it is built. For example, a number of attempts were made in the 1960s and 1970s to produce secure computing systems, using a secure operating system environment as a basis [24]. An essential presumption of the security arguments for these designs was that system lay-

Secure Boot ++

- **Secure boot:** guarantees that only authorized software boots

Secure Boot ++

- **Secure boot:** guarantees that only authorized software boots
- **Runtime Program Memory Immutability:** The authorized booted software can not be modified at runtime
- **Data Execution Prevention:** Unauthorized software may be injected into Data Memory… but it can never execute.

Secure Boot ++

- **Secure boot:** guarantees that only authorized software boots
- **Runtime Program Memory Immutability:** The authorized booted software can not be modified at runtime
- **Data Execution Prevention:** Unauthorized software may be injected into Data Memory… but it can never execute.

Are we done?
Did we solve MCU software integrity the problem?
Any issues remain?

Secure Boot ++

- Remote Software Updates:
  - Send the new software to the MCU (over the network)
  - New software must be received by an update function implemented as MCU software.
  - Update function overwrites program memory with the newly received software.

Secure Boot ++

- Remote Software Updates:
  - Send the new software to the MCU (over the network)
  - New software must be received by an update function implemented as MCU software.
  - Update function overwrites program memory with the newly received software.

Oops… We just killed remote software updates…

Secure Boot ++

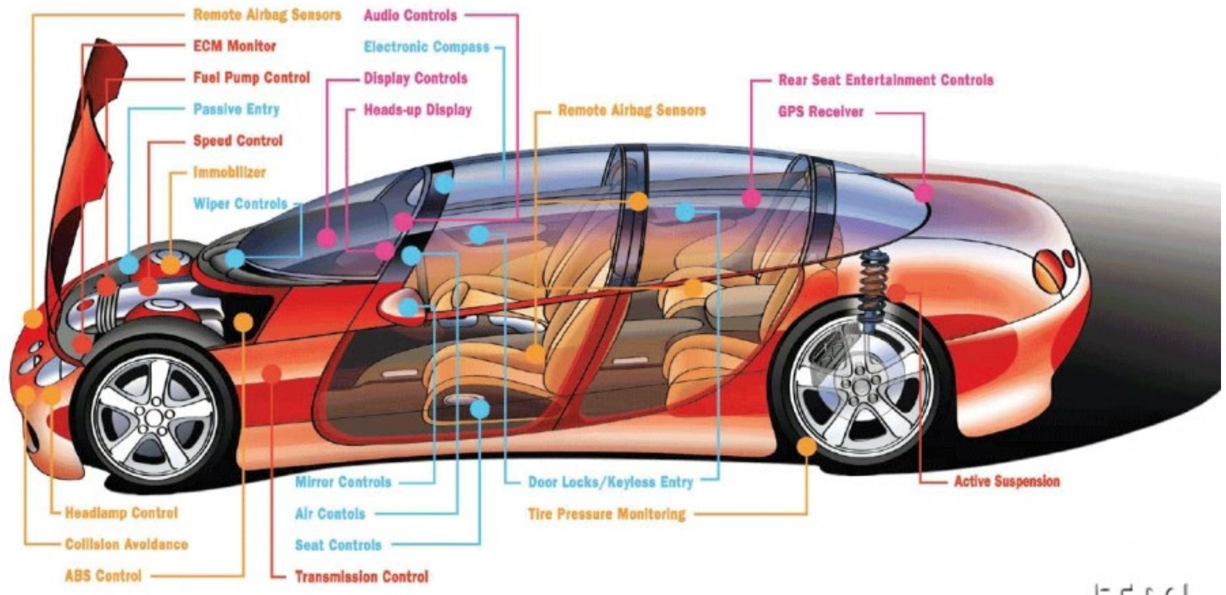
- Remote Software Updates:
  - Send the new software to the MCU (over the network)
  - New software must be received by an update function implemented as MCU software.
  - Update function overwrites program memory with the newly received software.

Oops… We just killed remote software updates…

Who cares?

Remote Software Updates

- These guys probably care:



4. Today's vehicles feature many more MCUs, which control numerous functions. *(Courtesy of Microchip Technology)*

MCU Software Integrity: Prevention vs. Detection

- Bottom-line:
  - Preventing malicious software modifications is hard!
  - Possible… but often too limiting…

MCU Software Integrity: Prevention vs. Detection

- Bottom-line:
  - Preventing malicious software modifications is hard!
  - Possible… but often too limiting…
  - An alternative approach.
    - **Detection-based integrity**

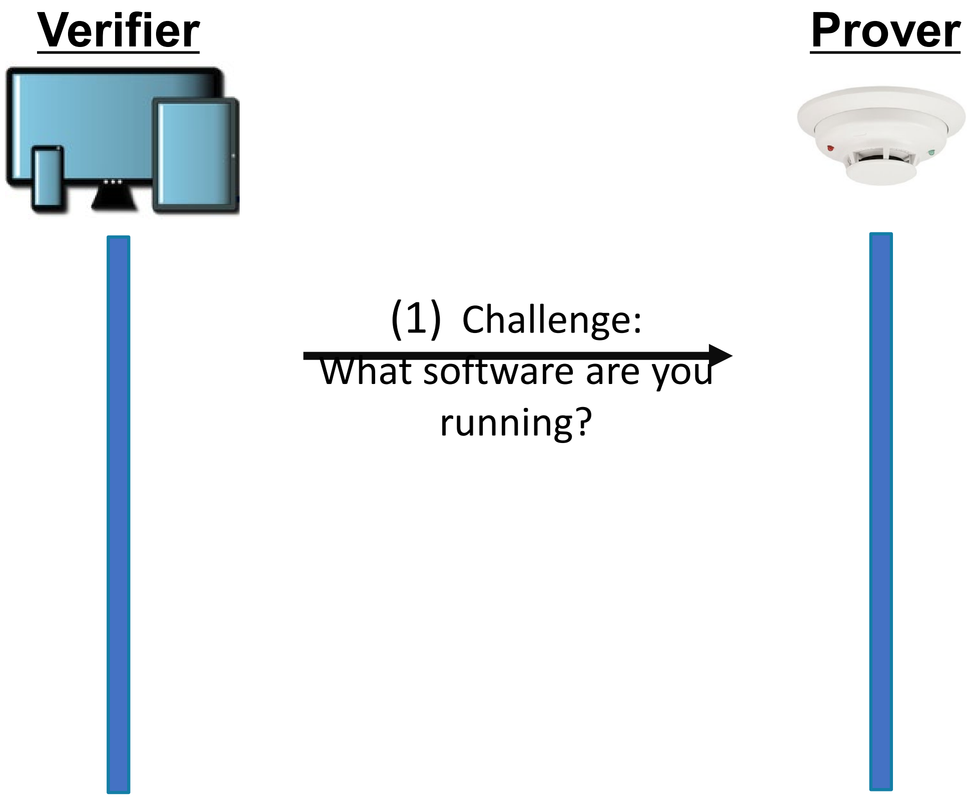
Allow software to change (for the good or for the bad)…
But always check/measure the software before using it!

Detection of Illegal IoT Code Modifications and

Remote Attestation

Remote Attestation (RA)

- General interaction between Verifier and Prover:



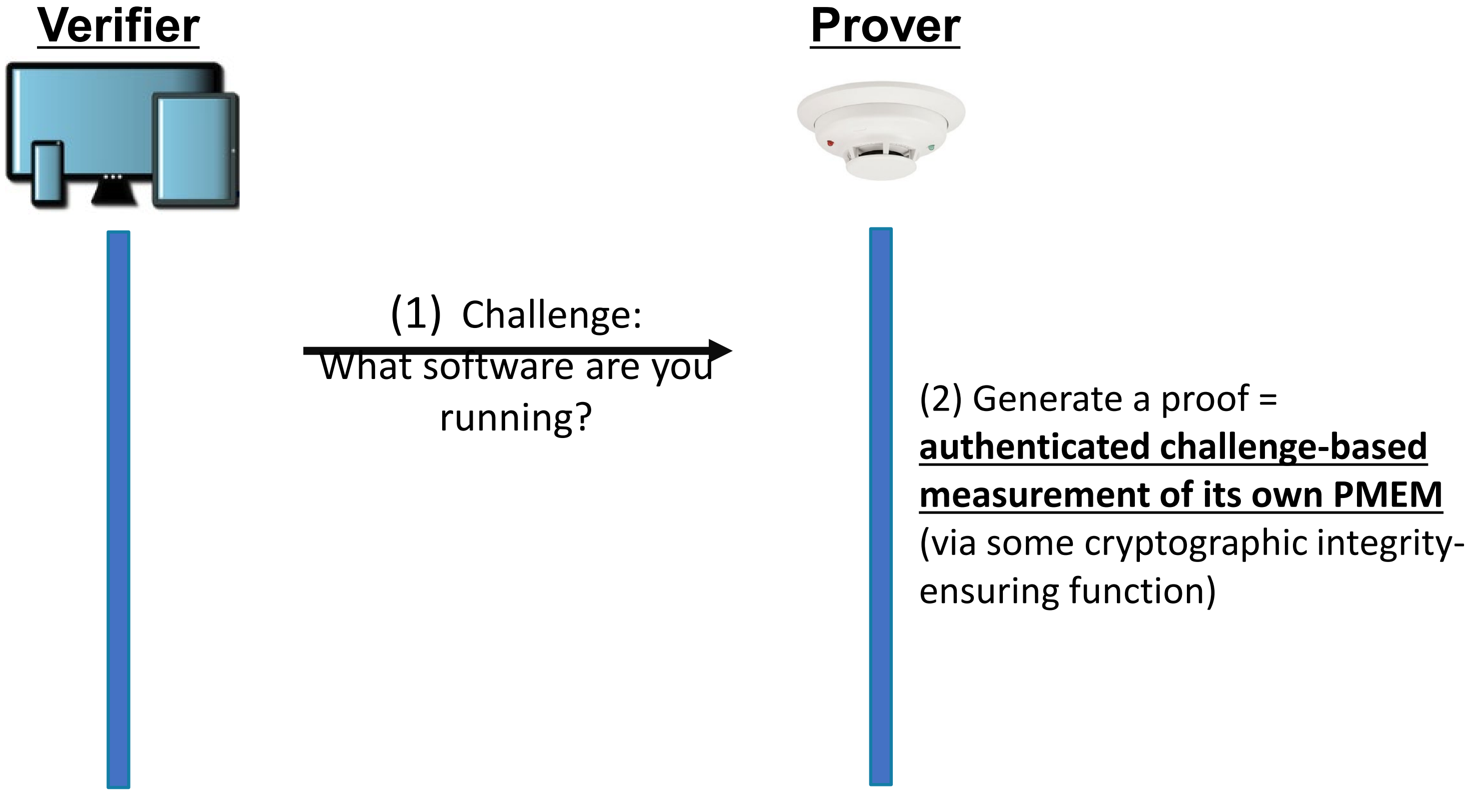
Verifier

Prover

(1) Challenge:
What software are you running?

Remote Attestation (RA)

- General interaction between Verifier and Prover:



Verifier

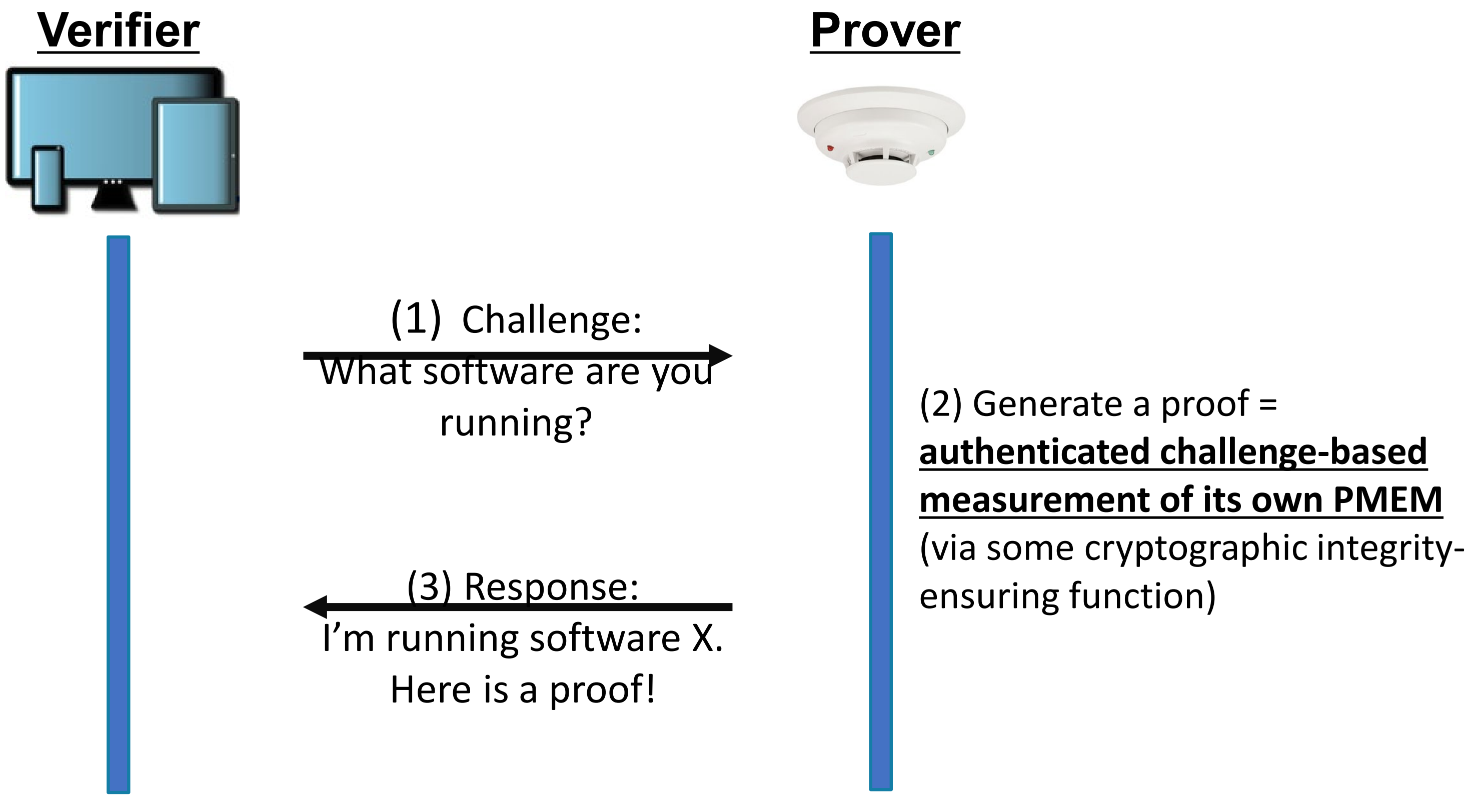
Prover

(1) Challenge: What software are you running?

(2) Generate a proof = **authenticated challenge-based measurement of its own PMEM** (via some cryptographic integrity-ensuring function)

Remote Attestation (RA)

- General interaction between Verifier and Prover:



Verifier

Prover

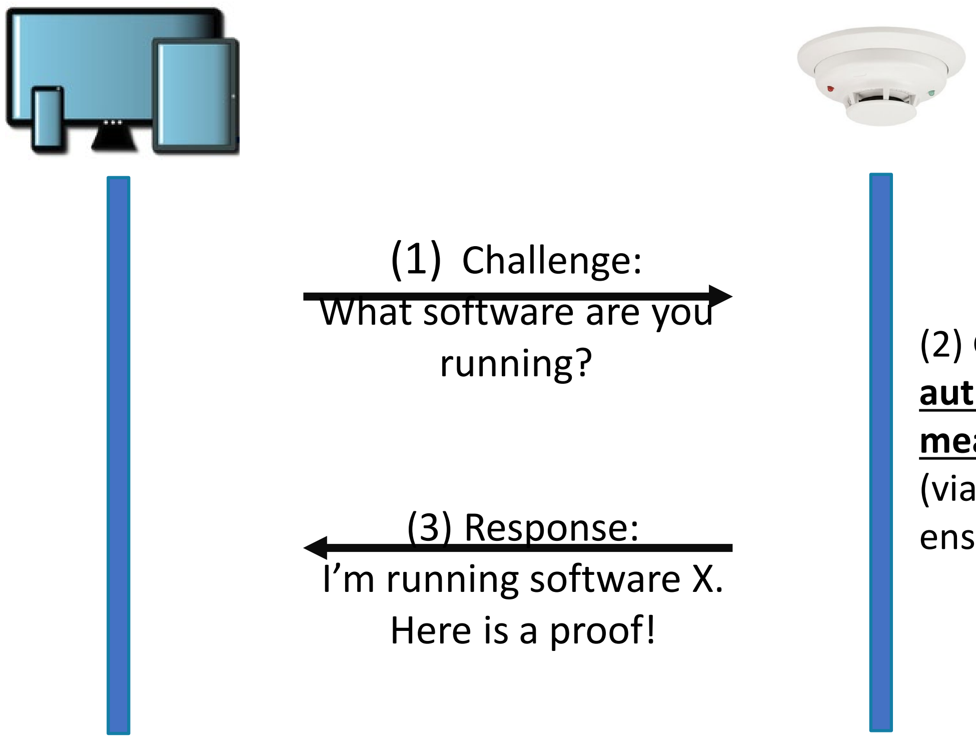
(1) Challenge:
What software are you running?

(2) Generate a proof = **authenticated challenge-based measurement of its own PMEM** (via some cryptographic integrity-ensuring function)

(3) Response:
I’m running software X.
Here is a proof!

Remote Attestation (RA)

- General interaction between Verifier and Prover:



Verifier

Prover

(1) Challenge:
What software are you running?

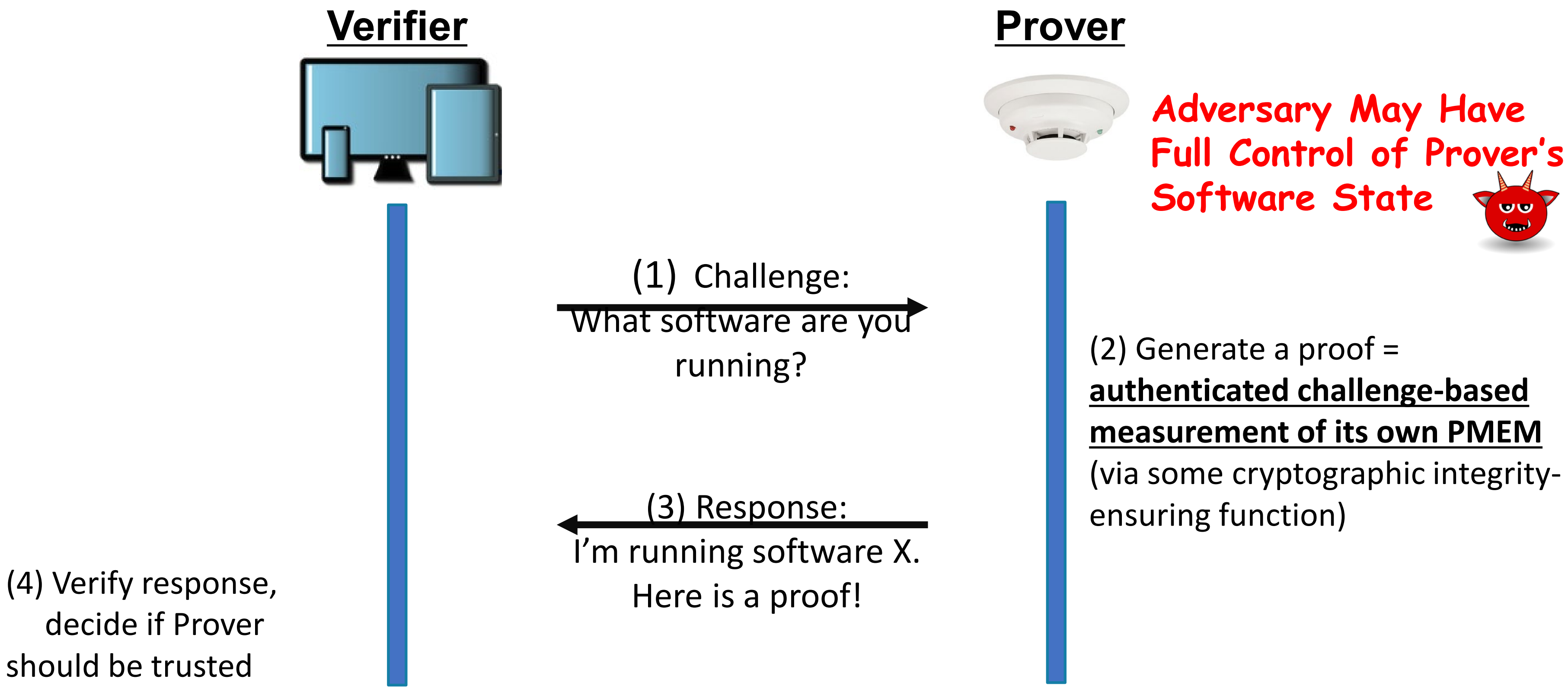
(2) Generate a proof = **authenticated challenge-based measurement of its own PMEM** (via some cryptographic integrity-ensuring function)

(3) Response:
I’m running software X.
Here is a proof!

(4) Verify response,
decide if Prover
should be trusted

Remote Attestation (RA)

- General interaction between Verifier and Prover:



Verifier

Prover

Adversary May Have Full Control of Prover's Software State

(1) Challenge: What software are you running?

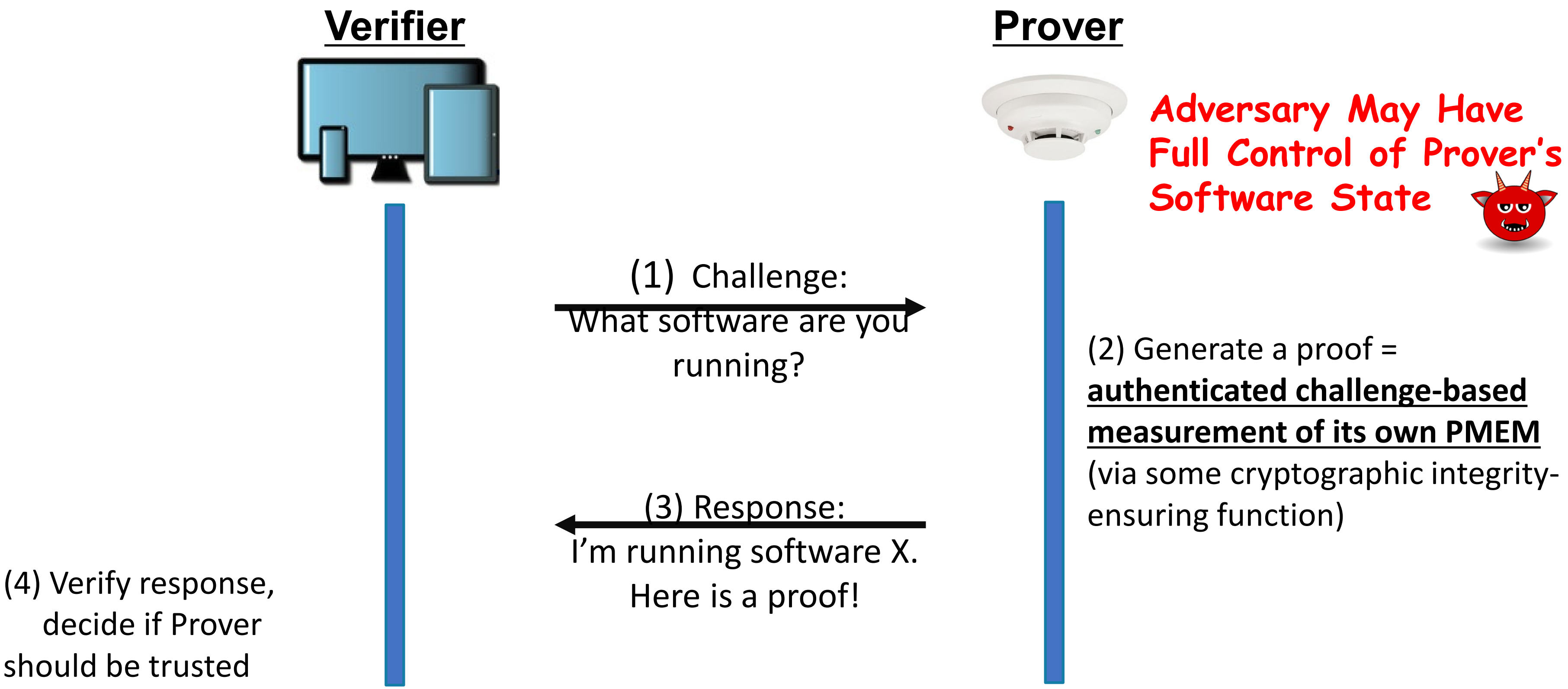
(2) Generate a proof = **authenticated challenge-based measurement of its own PMEM** (via some cryptographic integrity-ensuring function)

(3) Response: I'm running software X. Here is a proof!

(4) Verify response, decide if Prover should be trusted

Remote Attestation (RA)

- General interaction between Verifier and Prover:



Verifier

Prover

Adversary May Have Full Control of Prover's Software State

(1) Challenge: What software are you running?

(2) Generate a proof = **authenticated challenge-based measurement of its own PMEM** (via some cryptographic integrity-ensuring function)

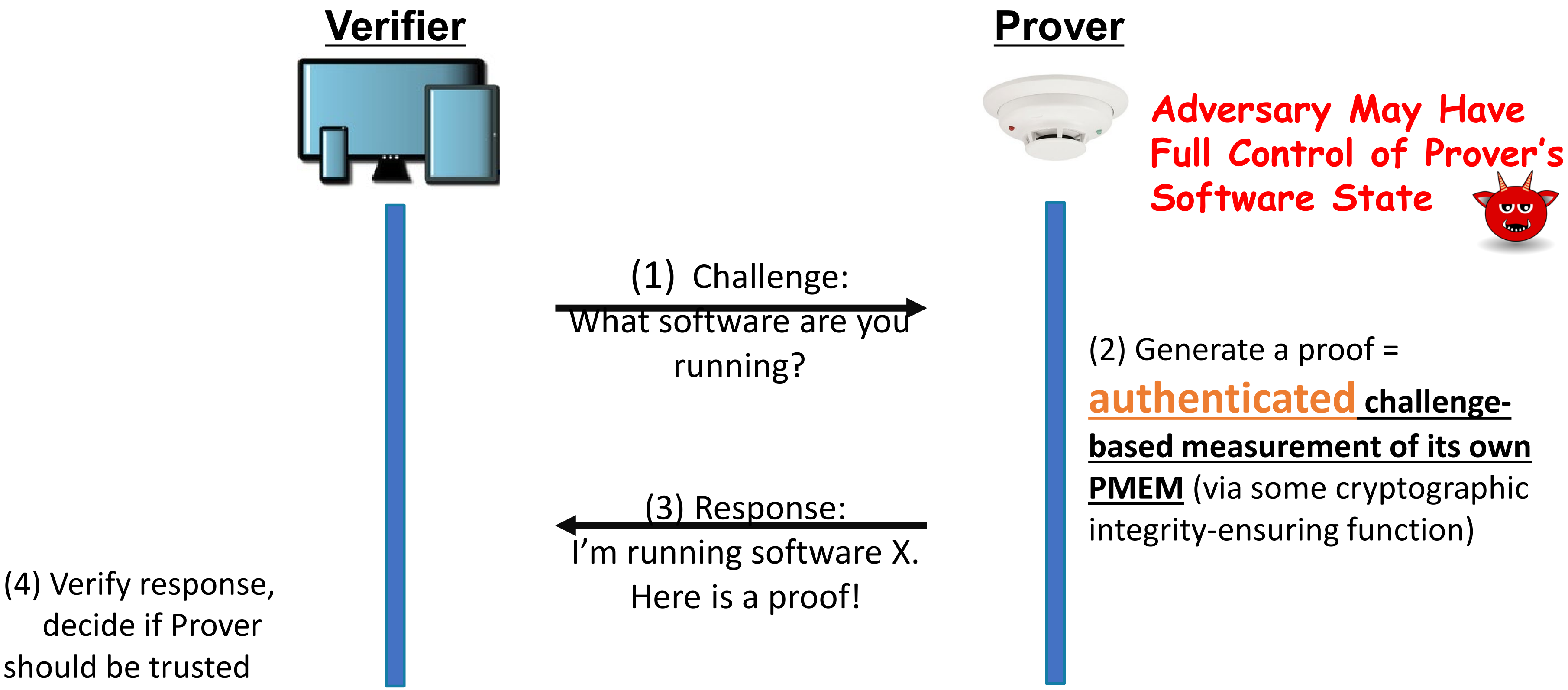
(3) Response: I'm running software X. Here is a proof!

(4) Verify response, decide if Prover should be trusted

Why is a secret required for this interrogation?

Remote Attestation (RA)

- General interaction between Verifier and Prover:



Verifier

Prover

Adversary May Have Full Control of Prover's Software State

(1) Challenge: What software are you running?

(2) Generate a proof = **authenticated challenge-based measurement of its own PMEM** (via some cryptographic integrity-ensuring function)

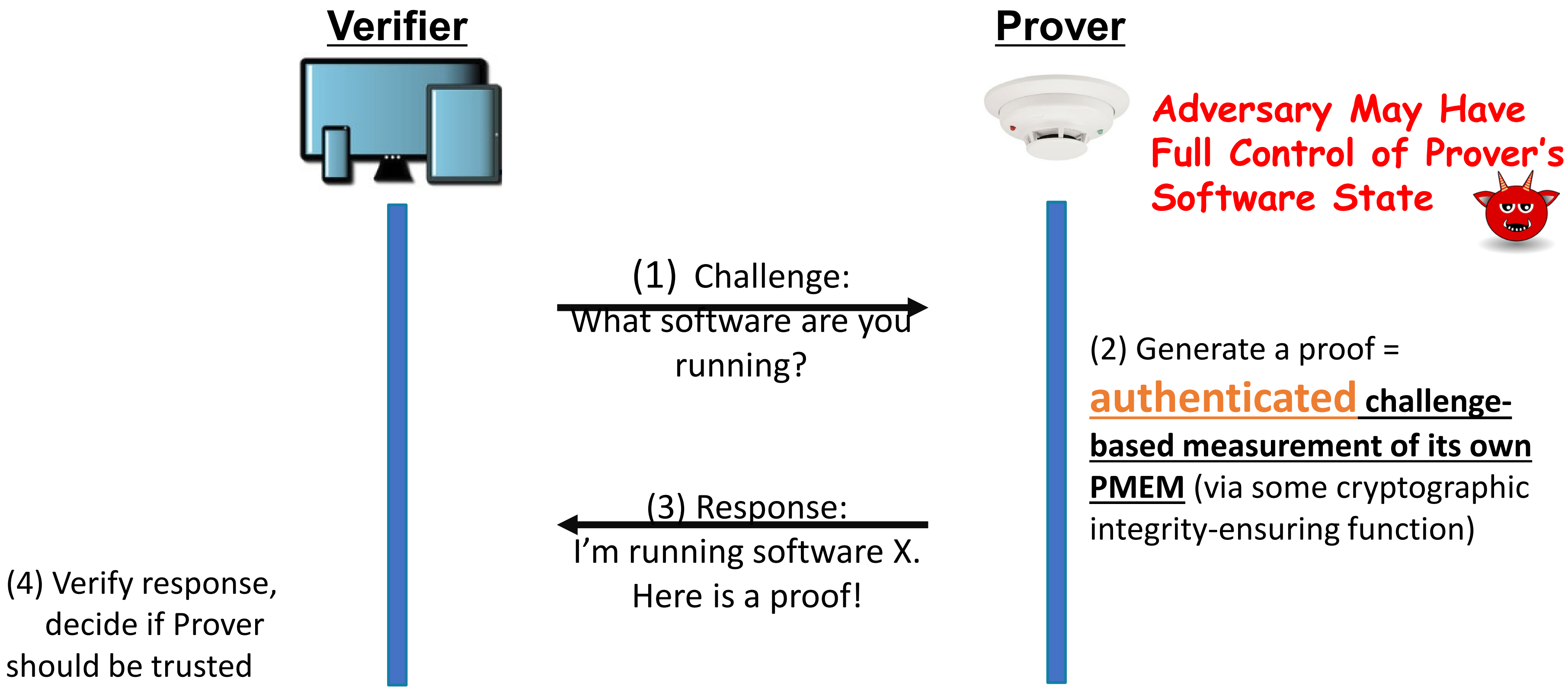
(3) Response: I'm running software X. Here is a proof!

(4) Verify response, decide if Prover should be trusted

Why is a secret required for this interrogation?

Remote Attestation (RA)

- General interaction between Verifier and Prover:



Verifier

Prover

Adversary May Have Full Control of Prover's Software State

(1) Challenge: What software are you running?

(2) Generate a proof = authenticated **challenge-based measurement of its own PMEM** (via some cryptographic integrity-ensuring function)

(3) Response: I'm running software X. Here is a proof!

(4) Verify response, decide if Prover should be trusted

Why is a secret required for this interrogation?

As in any interrogation: the guilty party might lie!

Remote Attestation (RA)

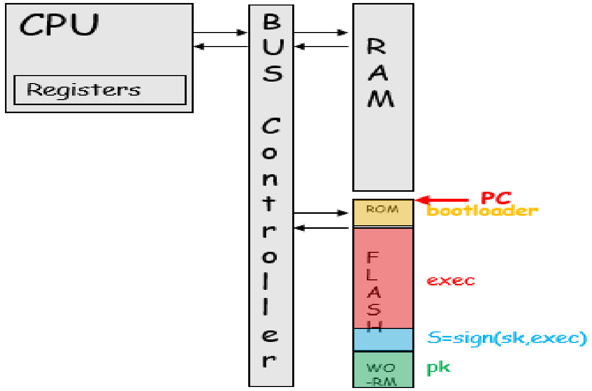
- **How to securely store and use secret keys in compromised devices?**

Remote Attestation (RA)

- **How to securely store and use secret keys in compromised devices?**

Option 1: Hybrid RA

Small modifications to this architecture's hardware & software to support secure computation on secrets

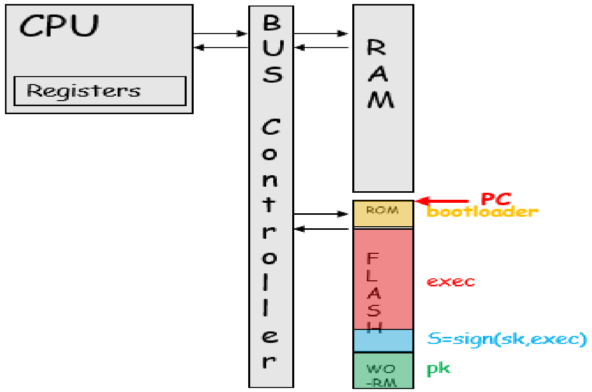


Tricky, but possible…

Remote Attestation (RA)

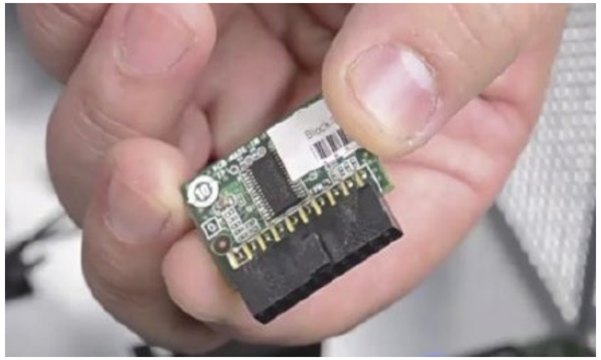
- **How to securely store and use RA keys in compromised devices?**

Option 1: Hybrid RA
Small modifications to this architecture's hardware & software to support secure computation on secrets



Tricky, but possible…

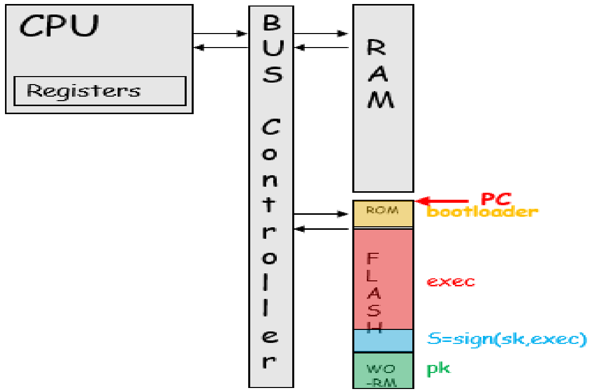
Option 2: Hardware-based RA
A separate purpose-specific cryptographic co-processor to store (and compute on) secrets. e.g., Trusted Platform Module (TPM).



Remote Attestation (RA)

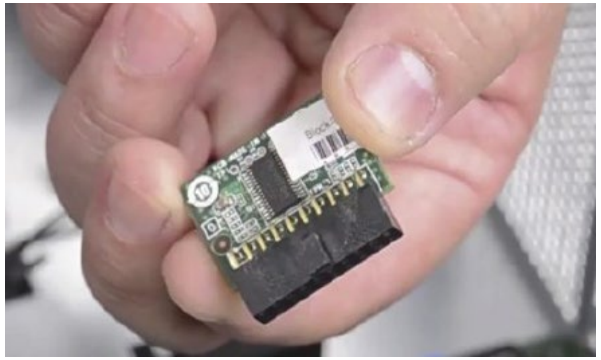
- **How to securely store and use RA keys in compromised devices?**

Option 1: Hybrid RA
Small modifications to this architecture's hardware & software to support secure computation on secrets



Tricky, but possible…

Option 2: Hardware-based RA
A separate purpose-specific cryptographic co-processor to store (and compute on) secrets. e.g., Trusted Platform Module (TPM).

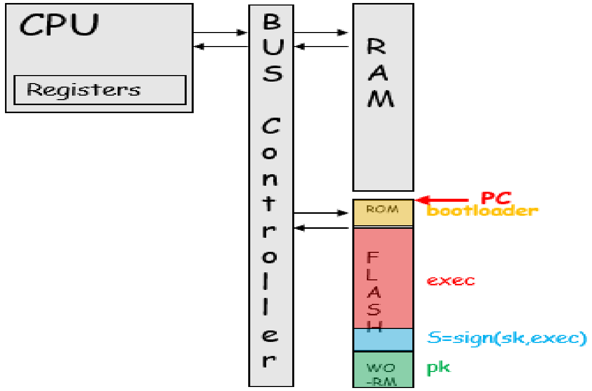


This is how high-end computers do it!
But too costly for MCUs…
One TPM costs a lot more than a typical MCU.

Remote Attestation (RA)

- **Two ways to implement a secure RA RoT:**

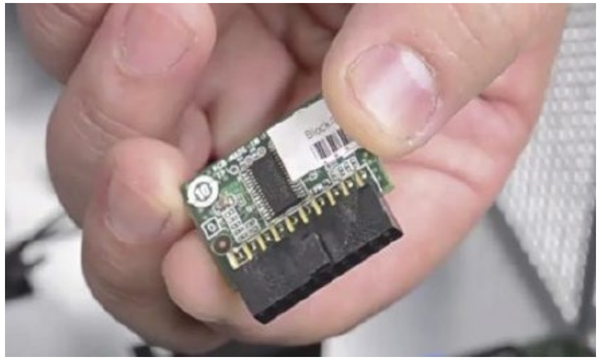
Option 1: Hybrid RA
Small modifications to this architecture's hardware & software to support secure computation on secrets



Tricky, but possible…

Best fit for resource-constrained MCUs…

Option 2: Hardware-based RA
A separate purpose-specific cryptographic co-processor to store (and compute on) secrets. e.g., Trusted Platform Module (TPM).



This is how high-end computers do it!
But too costly for MCUs…
One TPM costs a lot more than a typical MCU.

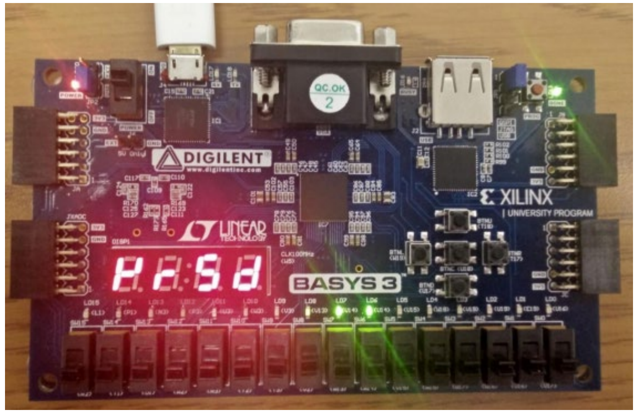
IoT Remote Attestation Architectures

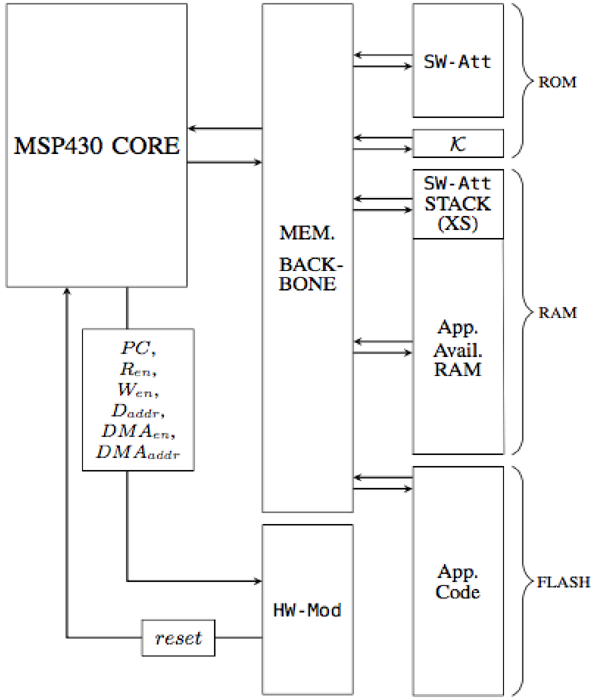
(Designing an affordable RA RoTs for resource-constrained MCUs)

[NDSS’12] SMART: Secure and Minimal Architecture for (Establishing Dynamic) Root of Trust.
[USENIX’13] Sancus: Low-cost trustworthy extensible networked devices with a zero-software trusted computing base.
[USENIX’19] VRASED: A verified hardware/software co-design for remote attestation.
[CCS’21] On the TOCTOU problem in remote attestation.

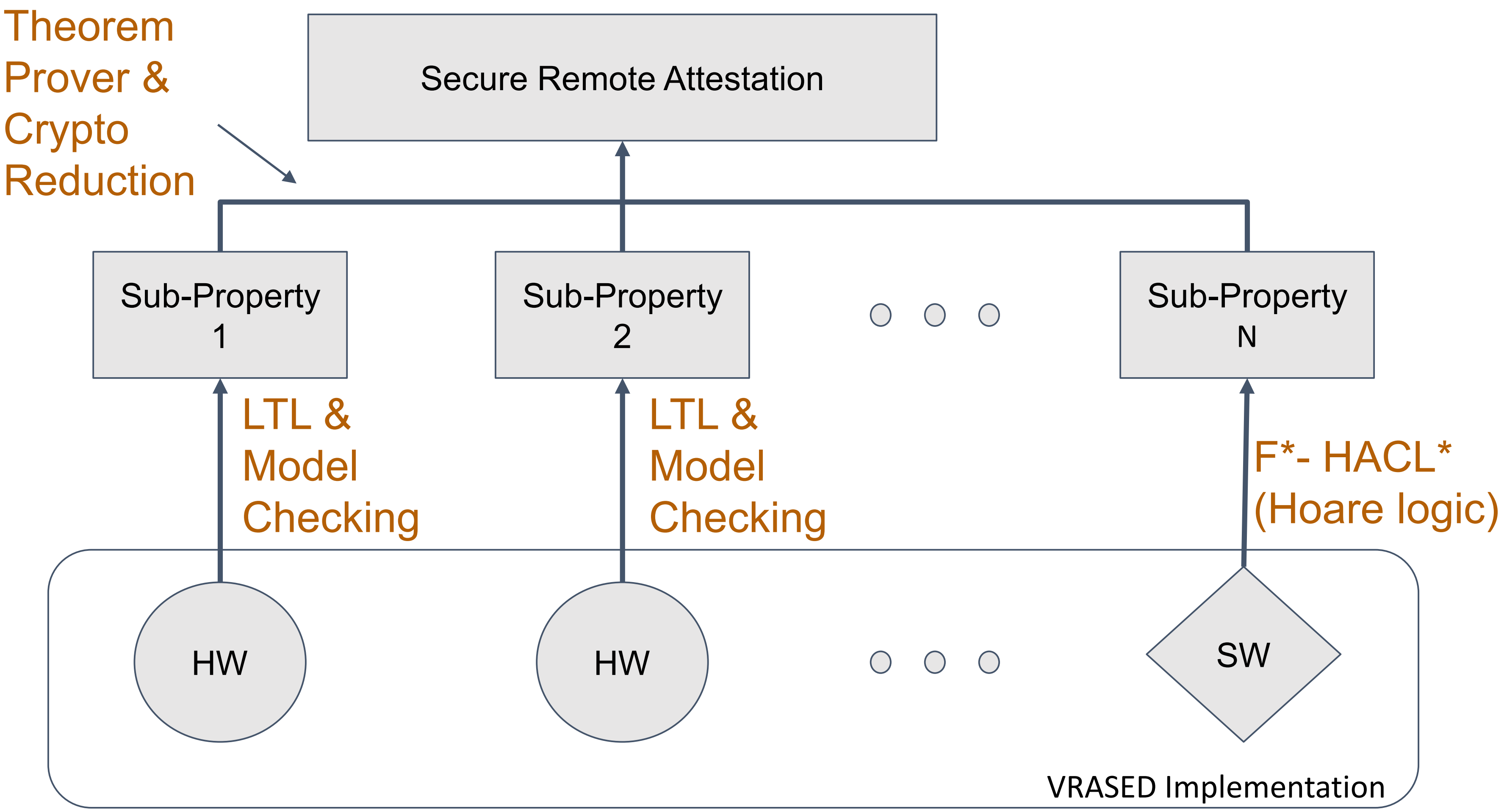
VRASED Hybrid RA Architecture

- VRASED: real RA implementation
  - Verilog Hardware Description Language (HDL)
  - Synthesized on the Basys 3 Field-Programable Gate Array (FPGA)
  - On top of the OpenMSP430 MCU
  - Formally Verified
- Open-source:
  - https://github.com/sprout-uci/vrased





Verifying Hybrid RA

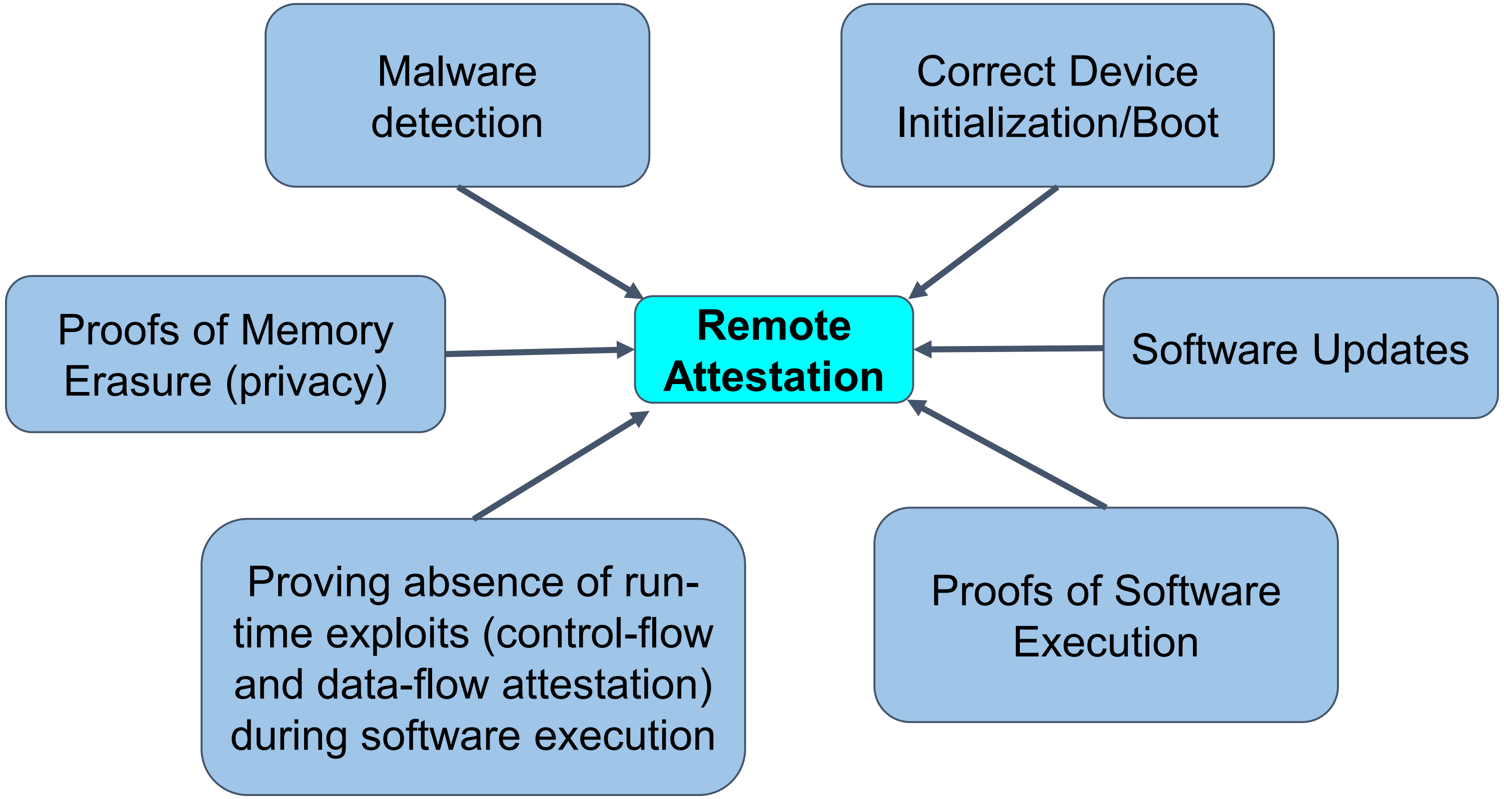


1) Define end-to-end secure RA property
2) Break it down into multiple sub-properties
3) Prove that sub-properties together imply end-to-end security
4) Implement VRASED HW/SW
5) Prove that each hw/sw module satisfies each sub-property

Based on (1-5), VRASED implementation is secure

See VRASED paper for details!

RA-based Security Services for IoT



Malware detection

Correct Device Initialization/Boot

Proofs of Memory Erasure (privacy)

Remote Attestation

Software Updates

Proving absence of run-time exploits (control-flow and data-flow attestation) during software execution

Proofs of Software Execution