Energy-Aware Security Mechanisms for Dynamic Networks of Resource-Constrained Devices

Vinod Ganapathy and Ulrich (Uli) Kremer

Department of Computer Science

Rutgers University

Evolution of Handheld Devices

Communication tool

General-purpose computing platform

Cameras

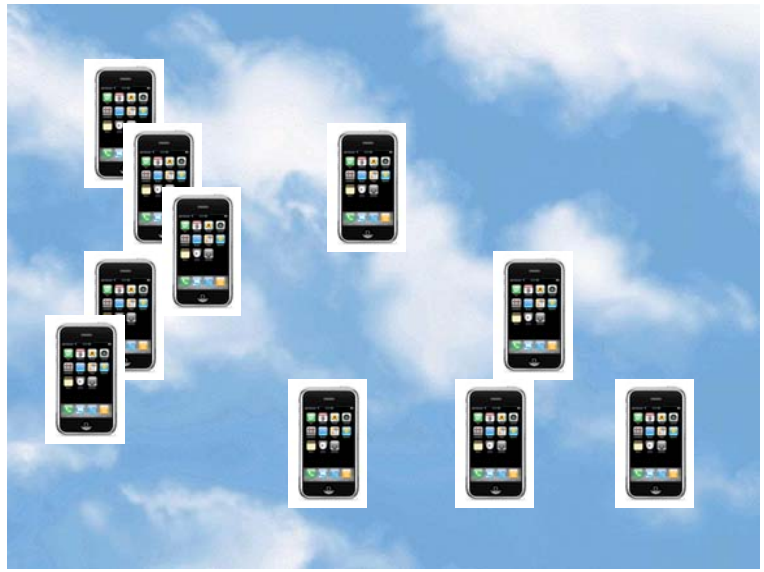
GPS

Temperature sensors

Communication (internet, email)

Data Centers vs. Cloud of Handhelds

Storage Resources



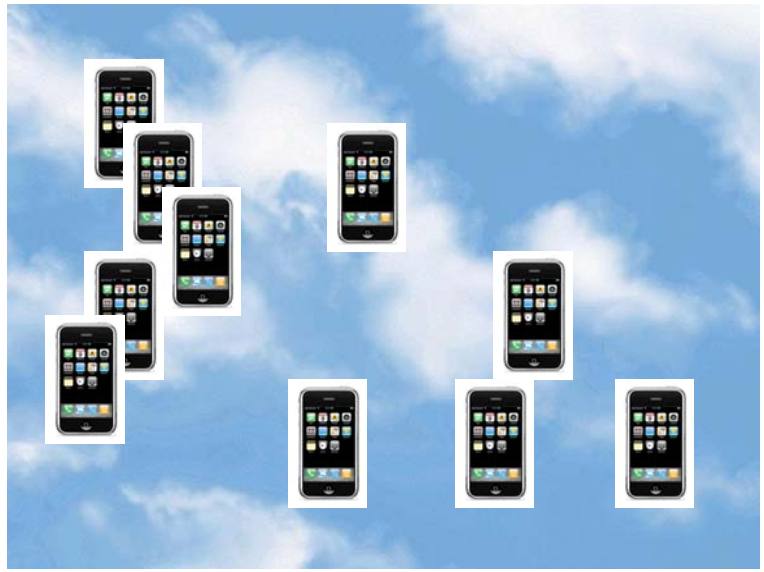
- 30 servers per rack
- 500GB storage per server
- data centers between 100s and 1000s of racks

- 260 million smart phones (2007-08)
- 8 GB storage
- assume phone users are willing to share 10% of storage (0.8GB)

15,000 server racks worth of storage (225 PetaBytes)

Data Centers vs. Cloud of Handhelds

Why GREEN?



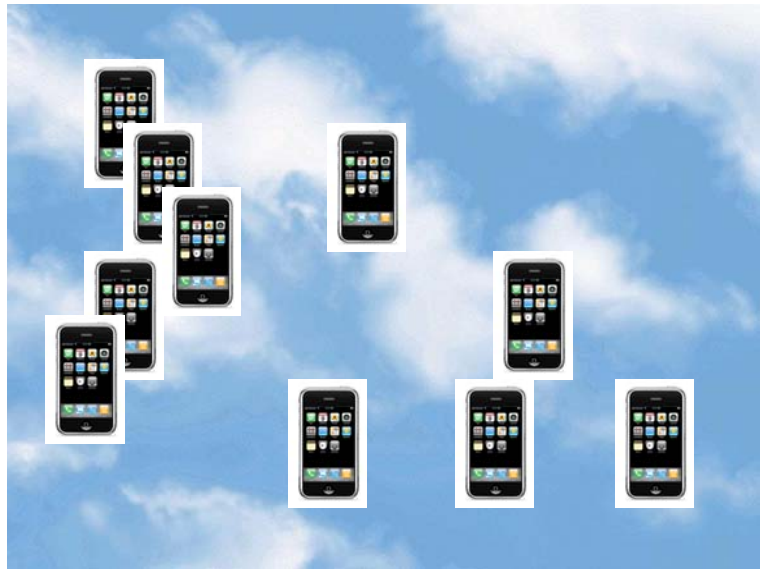
- large physical infrastructures (buildings, cooling, power grid)
- permanent maintenance personnel
- high dismantling costs

- no dedicated physical infrastructure
- owners maintain their devices
- existing recycling culture

Fewer power/energy/thermal/environmental issues

Data Centers vs. Cloud of Handhelds

Why GREEN?



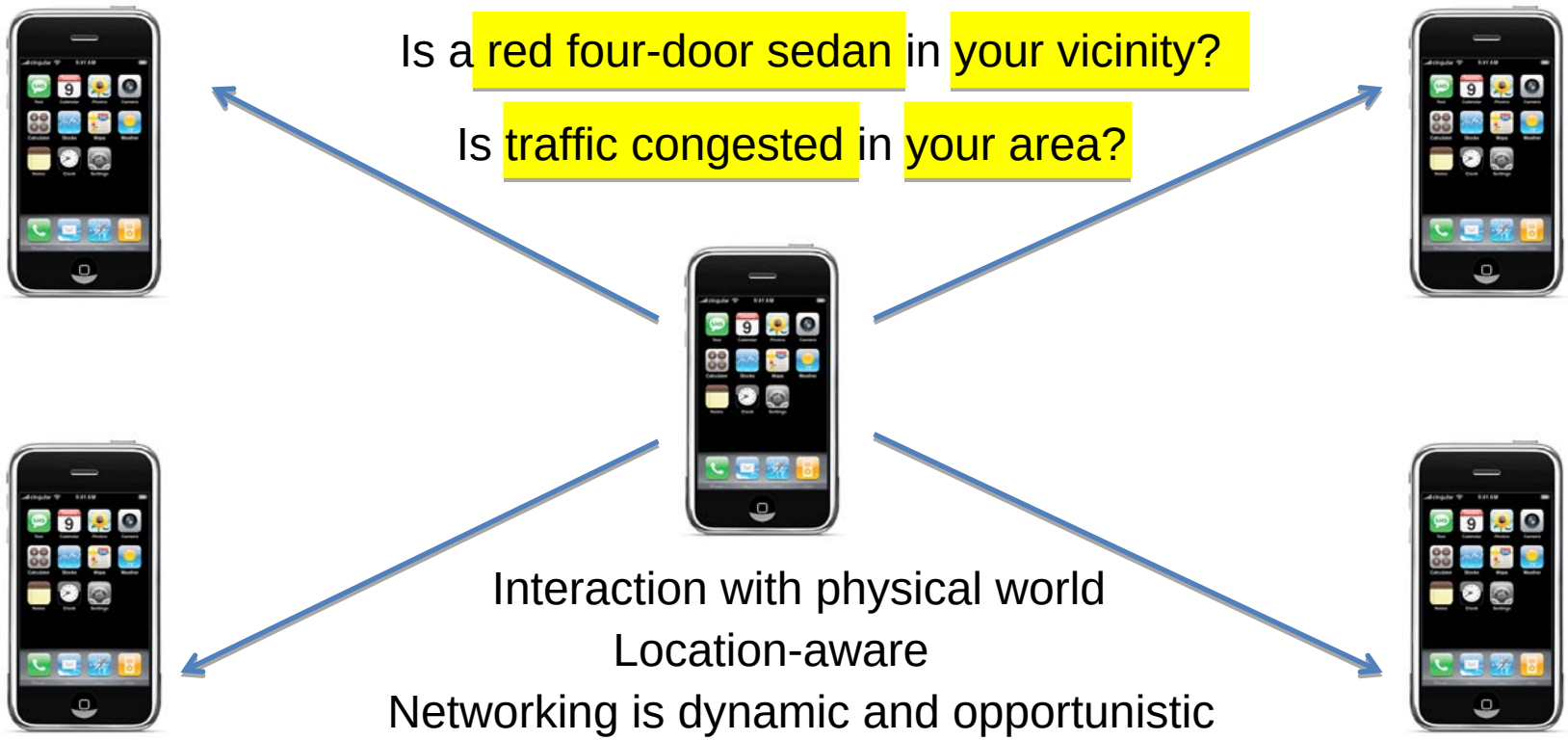
Large application domain that involves/requires

- interaction with physical world
- location awareness
- in-network data acquisition, processing, storage, and result delivery

Distributed solutions are more energy efficient

An Opportunity

Cyber-Physical Cloud Computing with Handhelds



Interaction with physical world

Location-aware

Networking is dynamic and opportunistic

Resource sharing is required

In-network data processing and result distribution

Dynamic Network Programming with Sarana

Is a red four-door sedan in your vicinity?

```
1.  public static void main(String[] args)
2.          SearchAttributes searchAttr = new SearchAttributes();
3.          searchAttr.parse(args); ..... // Record attributes of search target e.g., "red four-door sedan"
4.          Container carInfo = new Container();
5.          spatialview sv1 = camera @ Target_Geographic_Location
6.          visiteach cam ∈ sv1 every 30 secs within 3 hours
7.                  boolean successfulMatch = false;
8.                  Image img = cam.getImage(); ..... // Acquire image
9.                  Time time = System.currentTime(); ..... // Time of image acquisition
10.                 Location loc = System.currentLocation(); ..... // Location of camera
11.                 spatialview sv2 = imageUnderstandingCode @ new Circle(loc, 200m);
12.                 visitone imageAnalysis ∈ sv2
13.                         if (imageAnalysis.processImage(searchAttr, img) = match)
14.                                 successfulMatch = true;
15.                 if (successfulMatch)
16.                         carInfo.addElement(loc, time, img);
17.                         spatialview sv3 = AmberAlertDisplay @ new Circle(loc, 100m);
18.                         visiteach participant ∈ sv3
19.                                 participant.display(loc, time, img)
20.         carInfo.displayAll(); ..... // Display all images on launcher device
```

Rutgers/Princeton Collaborative Grant

An Opportunity

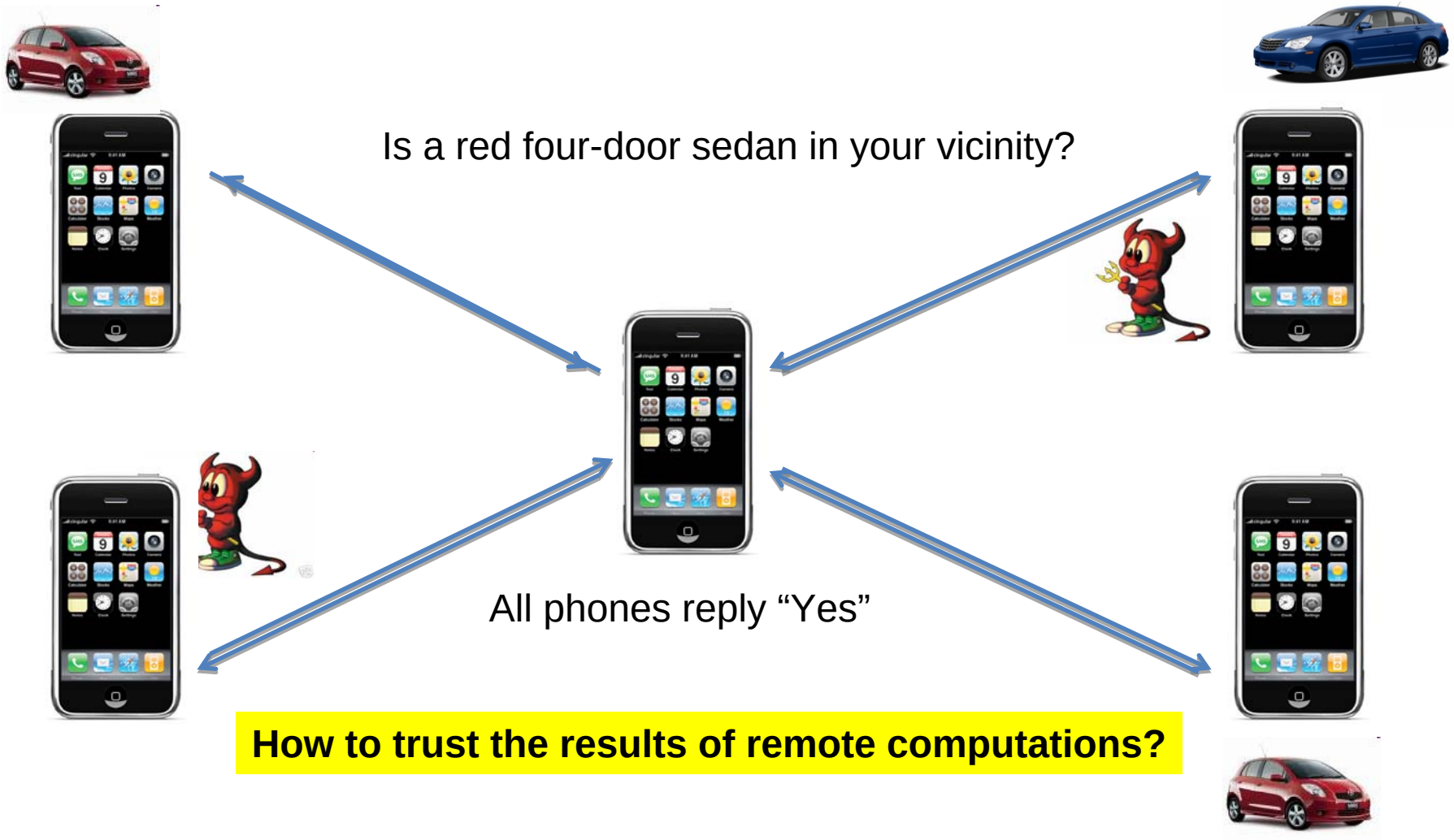
Cyber-Physical Cloud Computing with Handhelds

The Challenge

- how to protect against malicious “client code”?
- how to trust other nodes executing code on my behalf?
- how to trust sensor readings?

Energy and resource aware security mechanisms are a crucial enabling technology

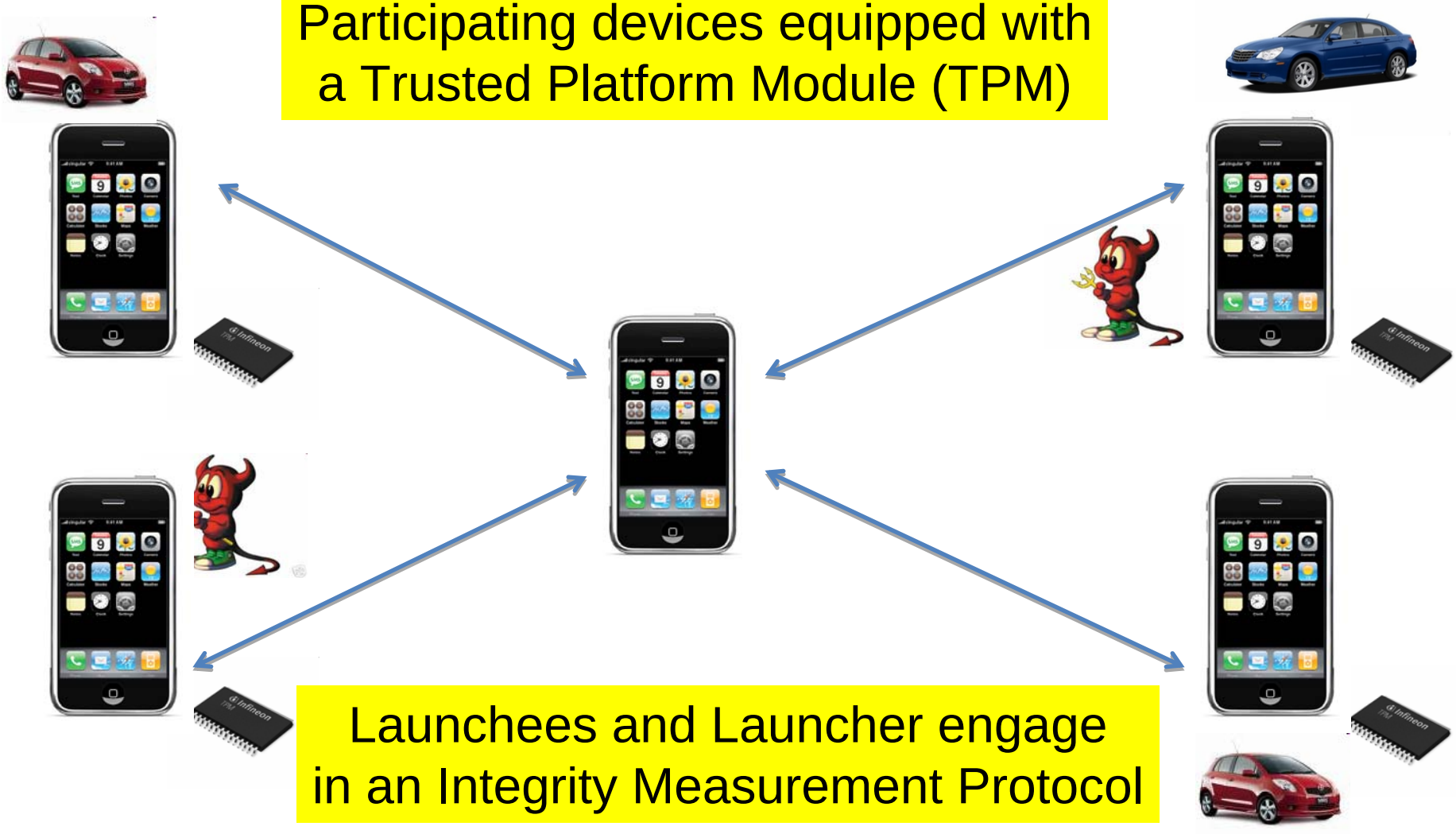
Key Challenge: Establishing Integrity of Remote Computations



How to trust the results of remote computations?

Trusted Computing for Mobile Devices

Participating devices equipped with a Trusted Platform Module (TPM)



Launchees and Launcher engage in an Integrity Measurement Protocol

Integrity Measurement Protocol

launchee

launcher

Prove to me that your software stack is not malicious

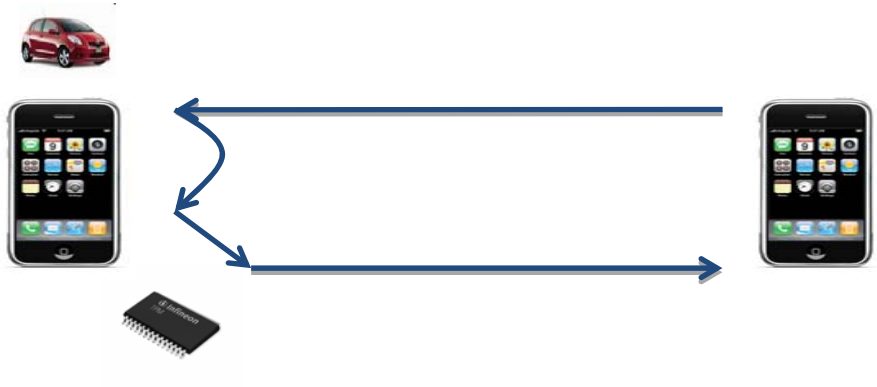
Compute hash chain (H) and metadata (M) about software stack

Digitally sign H and M

Send H and M

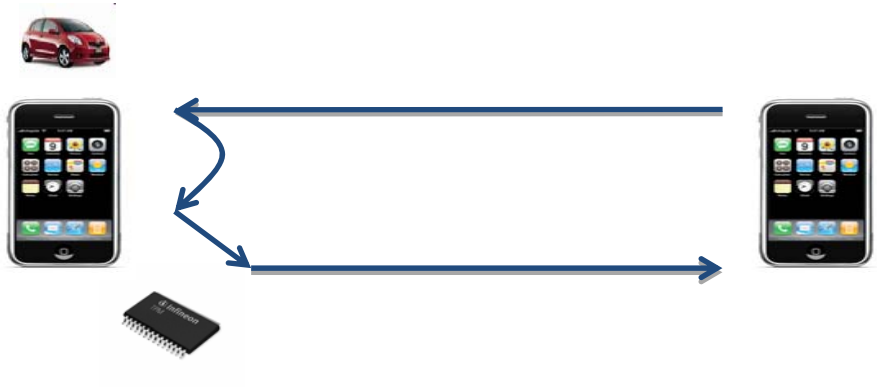
Verify Integrity

Problems with Integrity Measurement



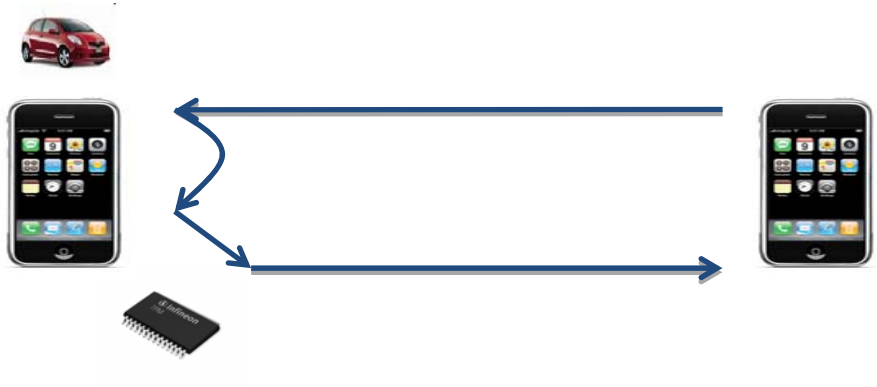
- Protocol is interactive:
  - Frequent communication between launcher and launchees
  - Costs: Communication bandwidth and battery

Problems with Integrity Measurement



- Protocol requires integrity measurement computations at launchee:
  - Compute hash of software stack and digital signatures.
  - Costs: Battery

Problems with Integrity Measurement



- Protocol requires integrity verification at launcher:
  - Receive integrity measurements
  - Verify digital signatures and hash chains
  - Store acceptable values and compare
  - Costs: Bandwidth, battery and storage

Problems to Investigate

- Problem: Protocol is interactive
  - Insight: Use Merkle hash trees to batch integrity measurements and make protocol less interactive
- Problem: Frequent integrity measurements
  - Insight: Batch and reuse integrity computations. Provide probabilistic freshness guarantee
- Problem: Cost-intensive integrity verification
  - Insight: Offload integrity verification to other cloud computing services

Verification of security tradeoffs through Sarana implementation and physical power measurements

Thanks