Connected Vehicle Security: Successes and Challenges

Dr. Virendra Kumar, OnBoard Security
vkumar@onboardsecurity.com

January 30th, 2018

OnBoardSecurity

Background

Traffic Safety

- 32k US road deaths, and 3.8M injuries annually
- Fatalities and injuries = $300B/year
- Congestion = $230B/year
- Leading cause of death for ages 15-34 in US

Technology Evolution

Passive → Active → Proactive

United States Department of Transportation

NHTSA

Ratings | Recalls | Risky Driving | Road Safety | Equipment | Technology & Innovation | REPORT A PROBLEM | Research & Data | Laws & Regulations | Information For

← NEWS

U.S. DOT advances deployment of Connected Vehicle Technology to prevent hundreds of thousands of crashes

Share:

December 13, 2016 | Washington, DC

Proposed rule would mandate vehicle-to-vehicle (V2V) communication on light vehicles, allowing cars to 'talk' to each other to avoid crashes

Citing an enormous potential to reduce crashes on U.S. roadways, the U.S. Department of Transportation issued a proposed rule today that would advance the deployment of connected vehicle technologies throughout the U.S. light vehicle fleet. The Notice of Proposed Rulemaking would enable vehicle-to-vehicle (V2V) communication technology on all new light-duty vehicles, enabling a multitude of new crash-avoidance applications that, once fully deployed, could prevent hundreds of thousands of crashes every year by helping vehicles "talk" to each other.

"We are carrying the ball as far as we can to realize the potential of transportation technology to save

ADDITIONAL RESOURCES

- NOTICE OF PROPOSED RULEMAKING ON V2V COMMUNICATIONS
- THE NHTSA V2V COMMUNICATIONS NPRM
- HOW CONNECTED VEHICLES WORK
- CONNECTED VEHICLE BENEFITS
- DEDICATED SHORT-RANGE COMMUNICATIONS (DSRC)

source: nhtsa.gov

TABLE I–1—COSTS * AND BENEFITS IN YEAR 30 OF DEPLOYMENT

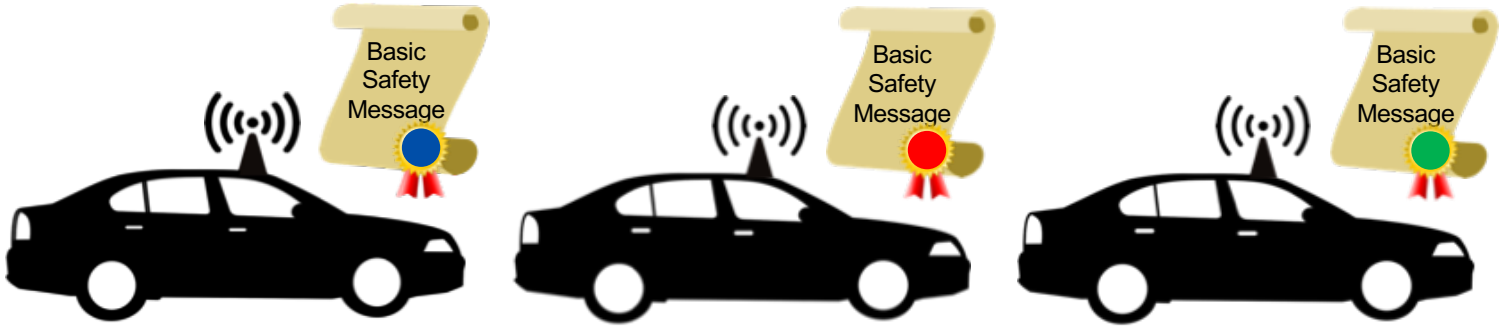
[2051]

Total annual costs	Per vehicle costs	Crashes prevented and lives saved	Monetary benefits (billions)
$2.2 billion–$5.0 billion	$135–$301	Crashes: 424,901–594,569 Lives: 955–1,321	$53–$71

* Note: Does not include spectrum opportunity costs, which will be included in the analysis of the final rule.

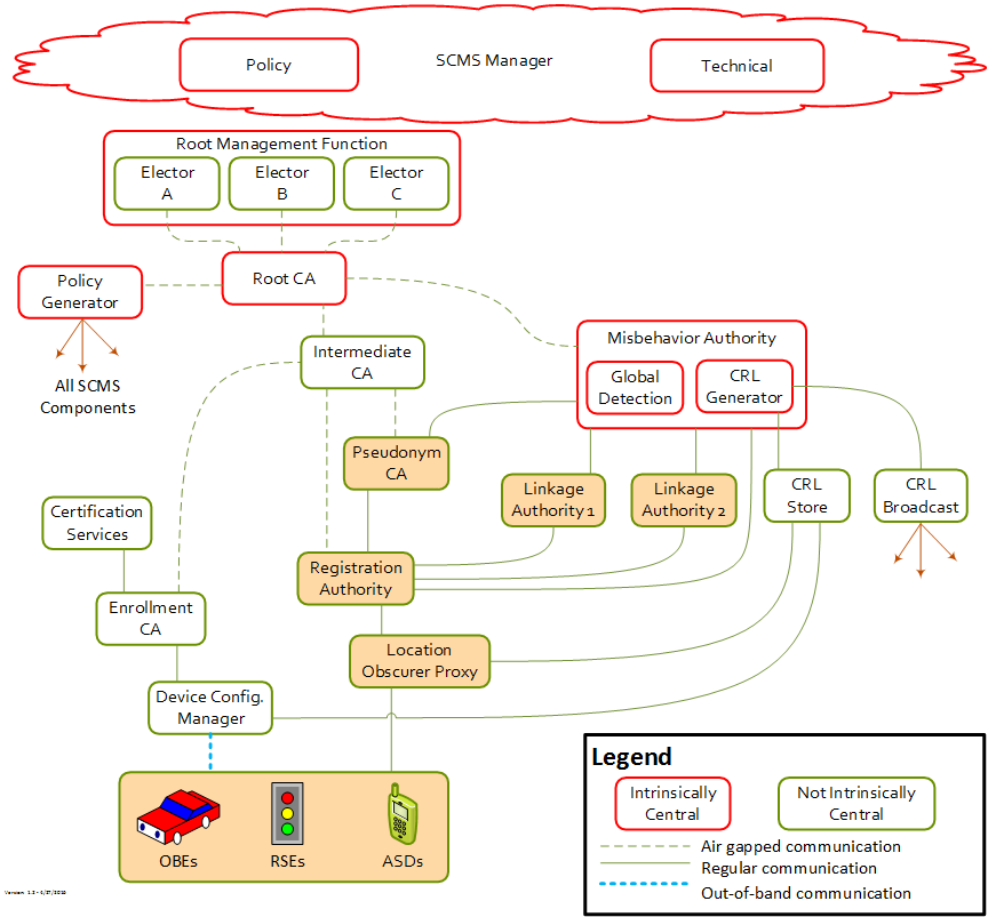
source: gpo.gov

V2V Communications Security



SCMS Design

- Security Credential Management System (SCMS – think PKI-on-steroids) for V2V includes privacy-preserving mechanisms
- Shuffle at RA to protect against CA learning certificates
- Linkage authorities to allow tracing misbehaving devices without revealing their identity, and revoking in a way that only allows them to be tracked after revocation
- Organization separation ensures no single insider / no single database breach can track any car



Pilot Deployments

NYCDOT Pilot

New York City DOT Pilot

THEA Pilot

Tampa-Hillsborough Expressway Authority Pilot

WYDOT Pilot

Wyoming DOT Pilot

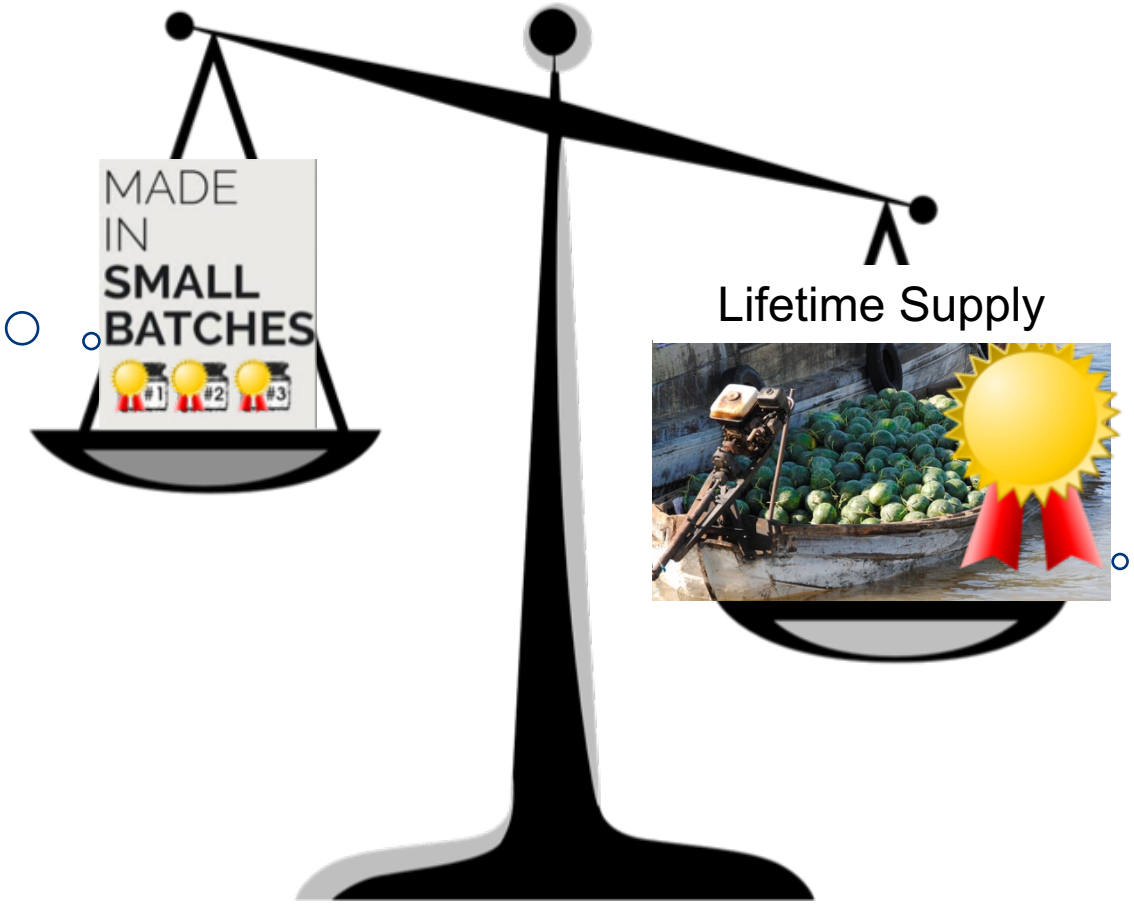
source: cvp.nyc, its.dot.gov

Binary Hash Tree Based Certificate Access Management

(joint work with Jonathan Petit and William Whyte, appeared at ACM WiSec 2017)

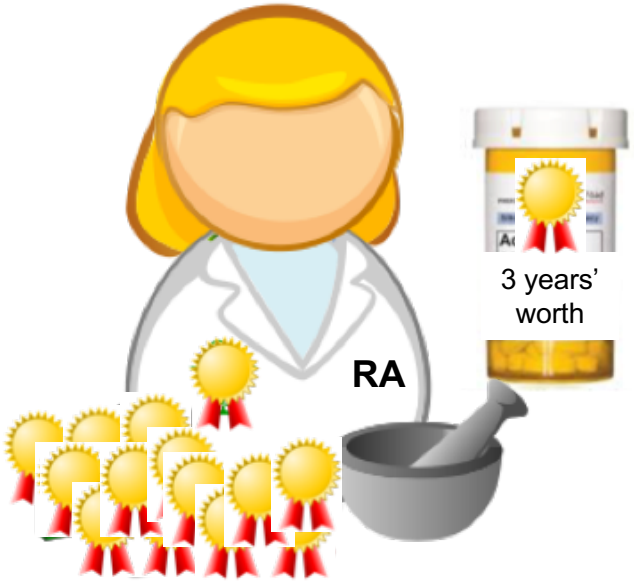
The Big Dilemma

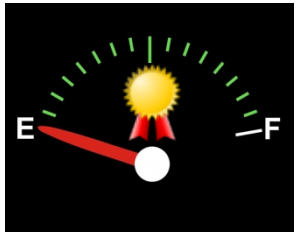
Who pays for 2-way connectivity?



What happens if the vehicle is hacked?

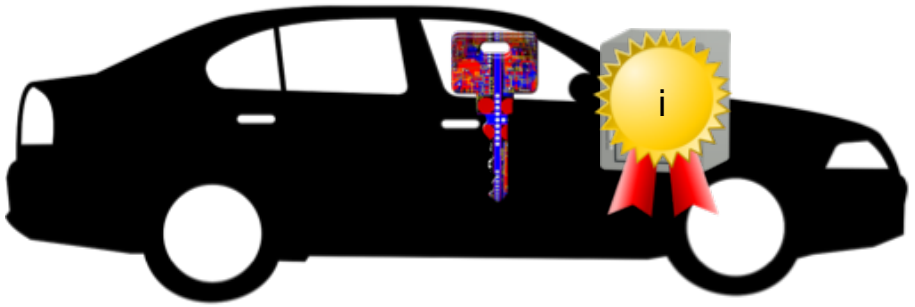
Current Certificate Model



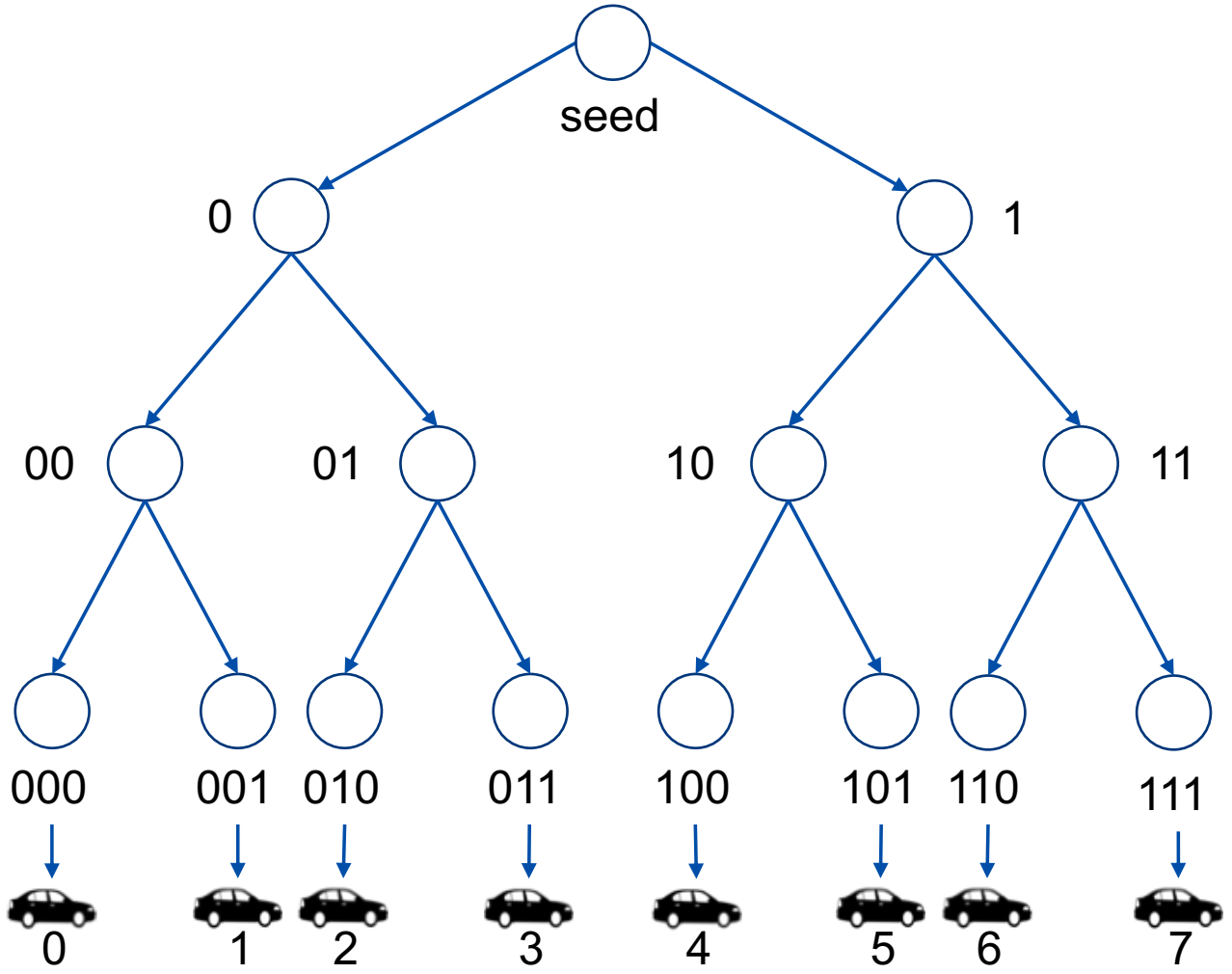


Encrypted Batches of Certificates

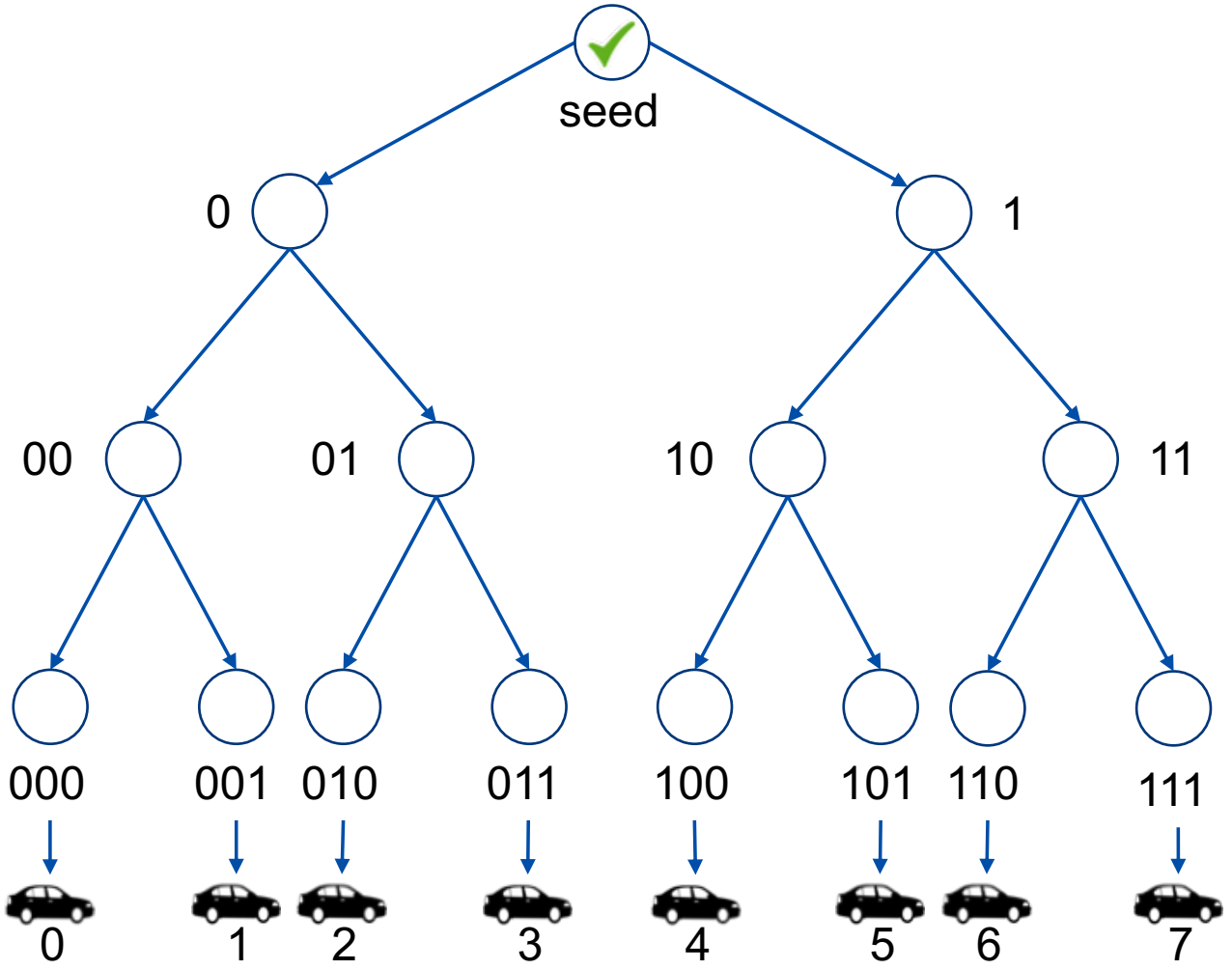
Periodic Key Updates



Compression using Binary Hash Trees



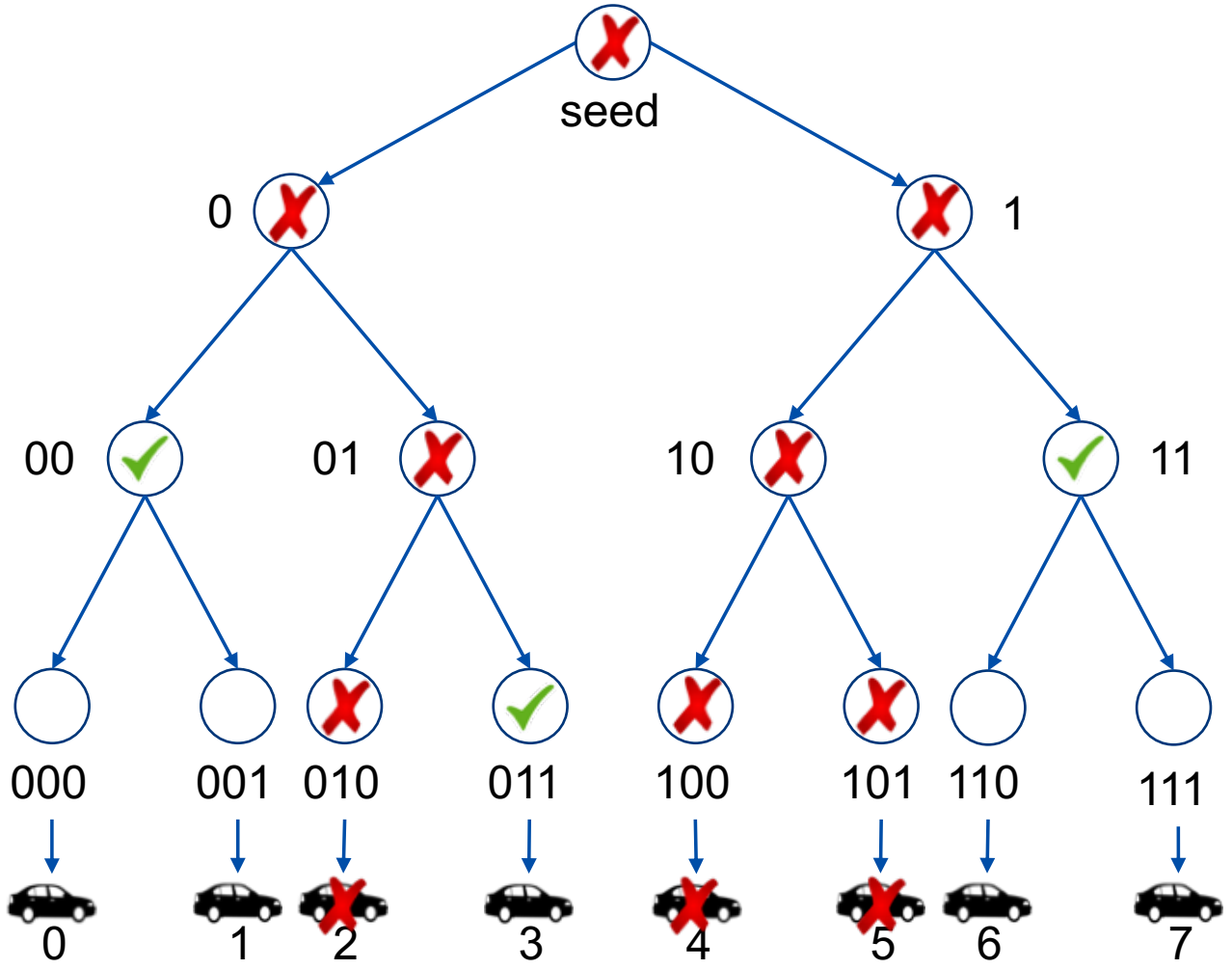
Day 1: No Revocation



Published node

Derived node

Day 2: Vehicles 2, 4, 5 Revoked

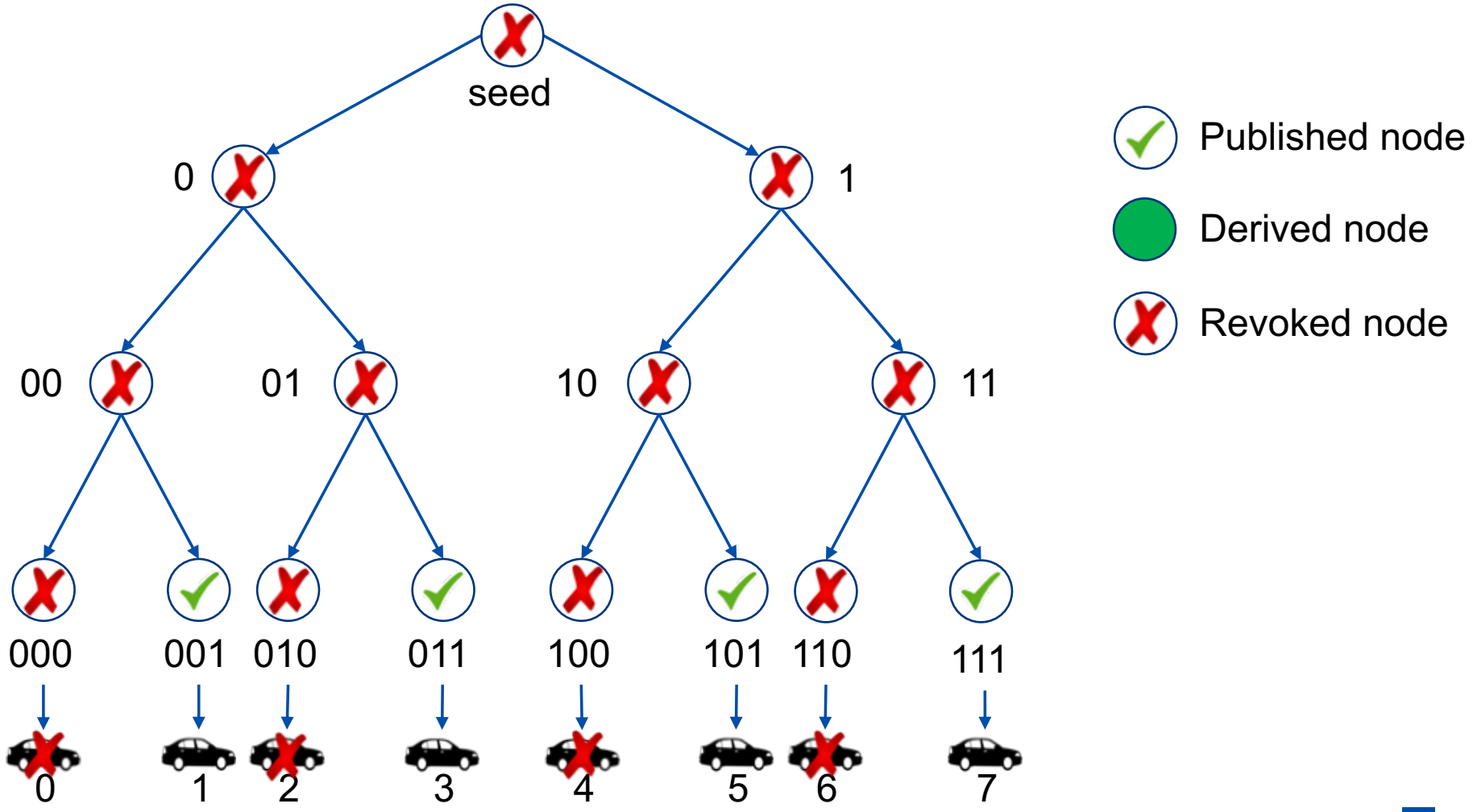


Published node

Derived node

Revoked node

Pathological: Every Other Vehicle Revoked



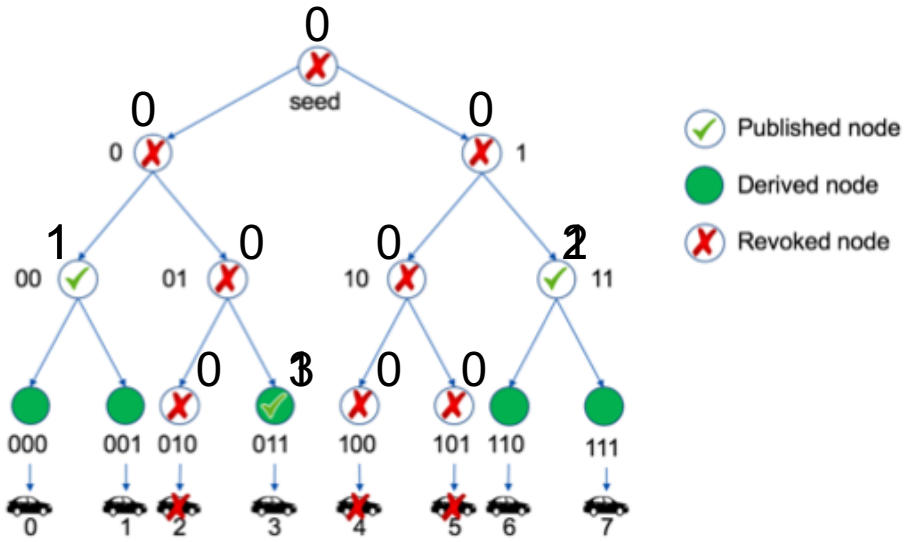
Binary Tree Encoding

n: number of leaf-nodes, r: number of revoked vehicles, $1 \le r \le n/2$

	Encoding Size	Decoding Time
Unique index of each published node	$\underbrace{r * \log_2(n/r)}_{\text{number of published nodes}} * (\log_2(n) + 1)$	Same as searching
Unique index of each revoked leaf node	$r * \log_2(n)$	No efficient algorithm known

Can we get the best of both worlds?

A New Algorithm for Full Binary Trees



Encoding:

1. Start from root with an empty string.
2. Do breadth-first traversal.
    1. Append 0 for revoked node.
    2. Append 1 for published node.
    3. Do nothing for derived node.

Observations:

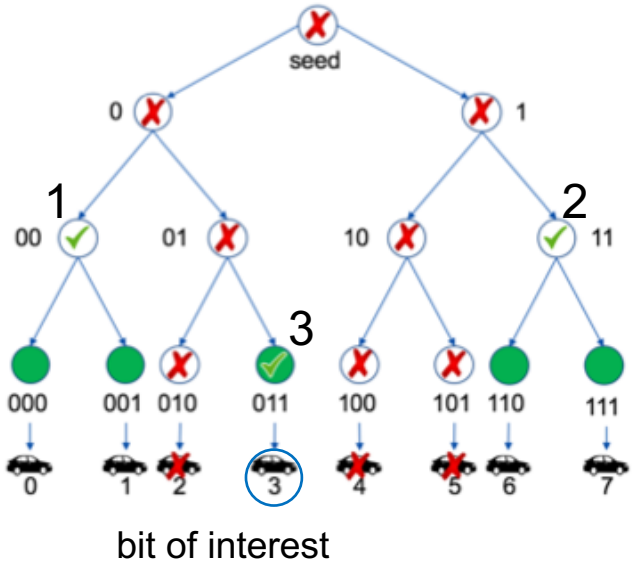
1. Topology known, only need to know which nodes are published and which are omitted.
2. Subtree of a published node can be ignored without any loss of information.

Encoded string: 0 00 1001 0100

Published nodes: 00, 11, 011

Disclaimer: Authors are not aware of any prior art with equivalent encoding sizes and decoding times.

A New Algorithm for Full Binary Trees Contd.



Encoding: 0 00 1001 0100

Bits at a level:

bits before bit of interest: 0

bits after bit of interest: 2

Decoding:

1. Start from root and process 1 level at a time.
2. At every level, look at the bit of interest
    1. If 0, go to next level.
    2. If 1, output the number of 1s so far, and stop.

Vehicle ID bit

Example (vehicle 3 ➔ 011): 3

Rules for going to next level:

1. # bits before = 2 * (# 0s in bits before bit of interest)
2. Add 1 to (# bits before), if next bit of vehicle ID is 1.
3. # bits after = 2 * (# 0s in bits after bit of interest)
4. Add 1 to (# bits after), if next bit of vehicle ID is 0.

Disclaimer: Authors are not aware of any prior art with equivalent encoding sizes and decoding times.

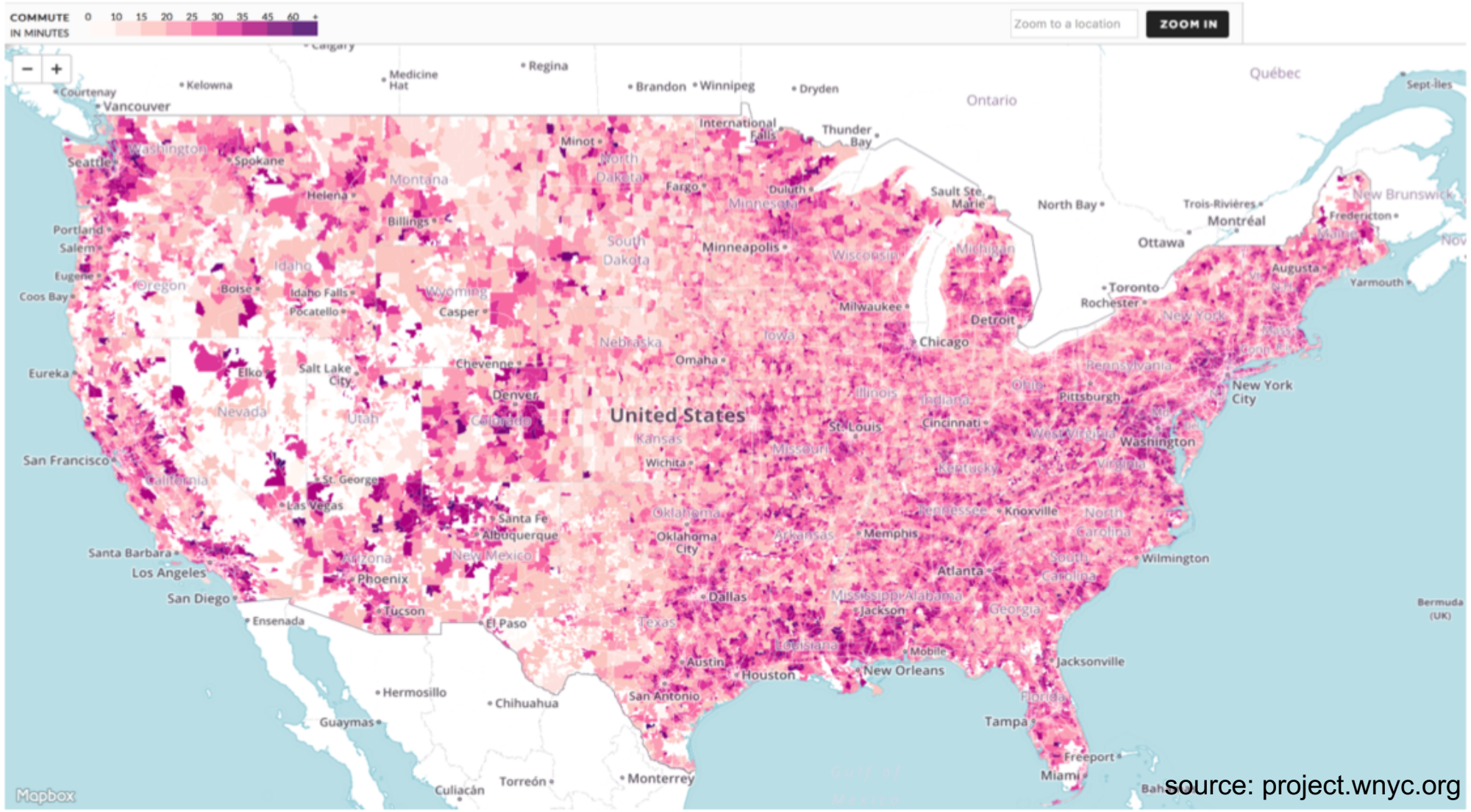
Efficiency of Encoding Algorithm

n: number of leaf-nodes, r: number of revoked vehicles, $1 \le r \le n/2$

- Encoding size
  - # published nodes ≈ # revoked nodes, i.e. encoding has roughly the same number of 0s and 1s.
  - Size ≈ $2*r*\log_2(n/r)$
  - For $n=2^{40}$, r=1,000, encoding takes less than 1% of the full packet, i.e. about 20 times smaller than using unique index of each published node.
- Decoding time
  - Breadth-first but queue size ≤ r.
  - For $n=2^{40}$, r=10,000, a consumer laptop (2.7 GHz Intel Core i7, 16GB RAM) takes less than 3 milliseconds on average.

Some Open Problems

Pseudonym Certificate Change Algorithm



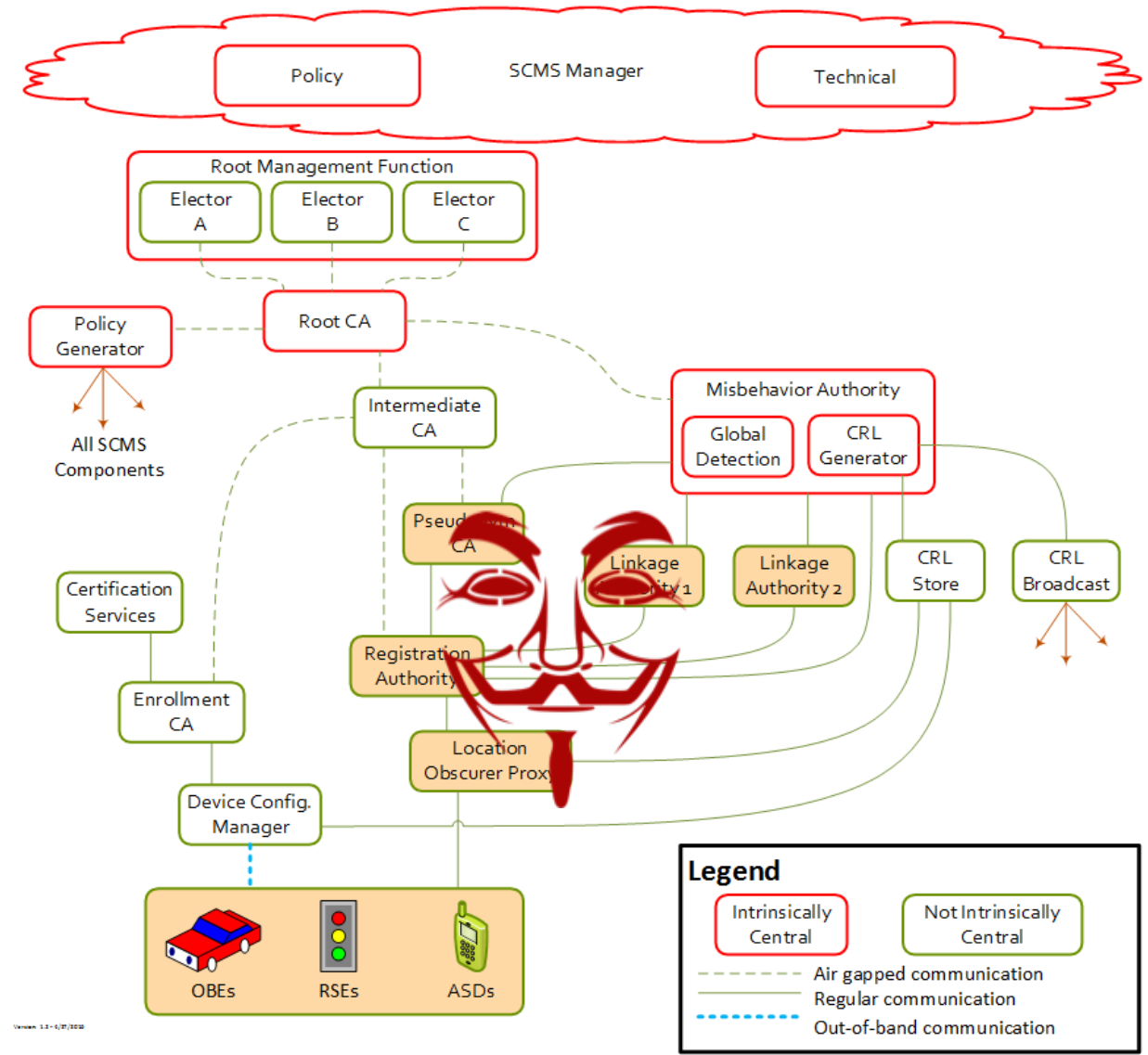
source: project.wnyc.org

Perfect Anonymity

- With the current approach, certificate batch size will be 3*365*24*3600*10 ≈ 1 Billion!
- Can we guarantee perfect anonymity, e.g. via group signatures?
  - Will the performance of such a signature scheme be acceptable?
  - Will it provide protection against insider attacks?
  - Will it provide efficient revocation?
  - Will it provide efficient addition of group members?
  - …

Group Revocation

Malicious Insiders



Misbehavior Identification

- Misbehaving vehicles can frequently change their pseudonym certificates
  - Can we correlate all the misbehavior by a given vehicle?
  - Can we do it while respecting the privacy of honest vehicles?
  - Can we do it when the Misbehavior Authority is malicious?
  - …

Secure Computation

Real Life Computation Problems

Solution: Trusted third party

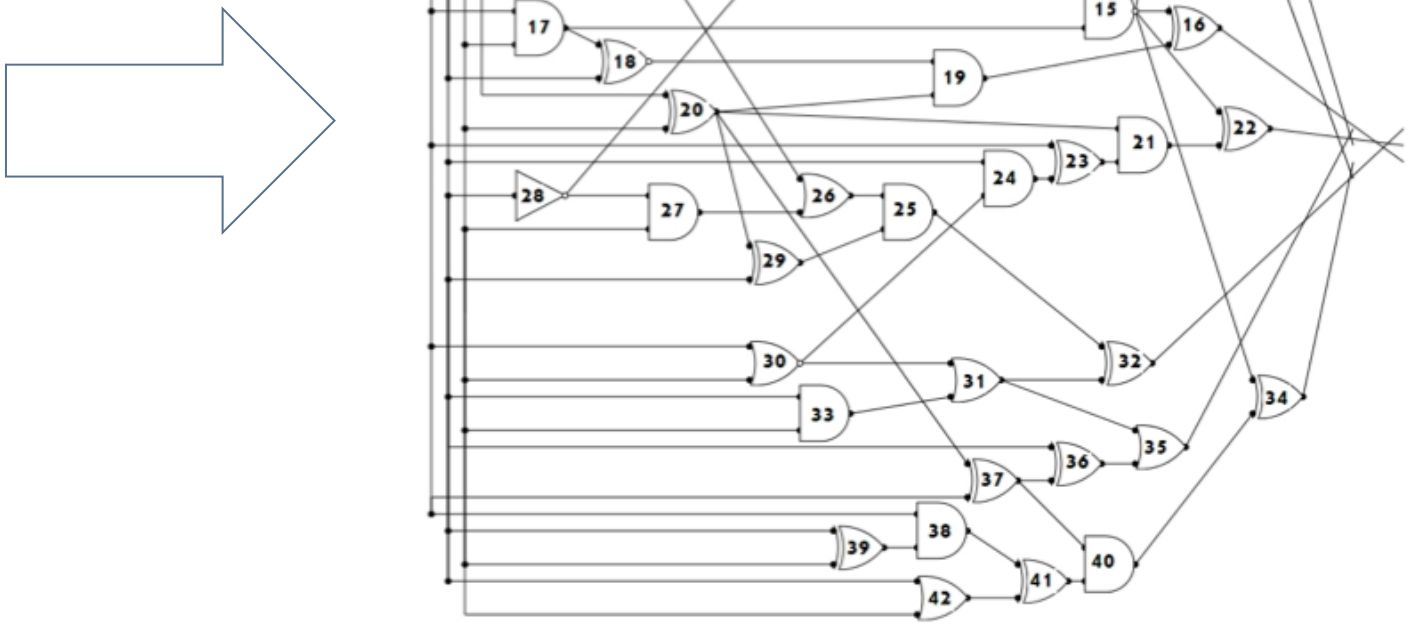
But, do we really have to?

Secure Computation

- Parties $P_1, P_2, \ldots, P_n$ with private inputs $x_1, x_2, \ldots, x_n$ can jointly compute any arbitrary function $f(x_1, x_2, \ldots, x_n)$, s.t.
  - Correctness: Output is guaranteed to be correct.
  - Privacy: Inputs are guaranteed to remain private.
  - …
- [Yao ’82] achieved this for n = 2.
- [Goldreich-Micali-Wigderson ’87] achieved this for $n \geq 2$.
- Active area of cryptographic research.

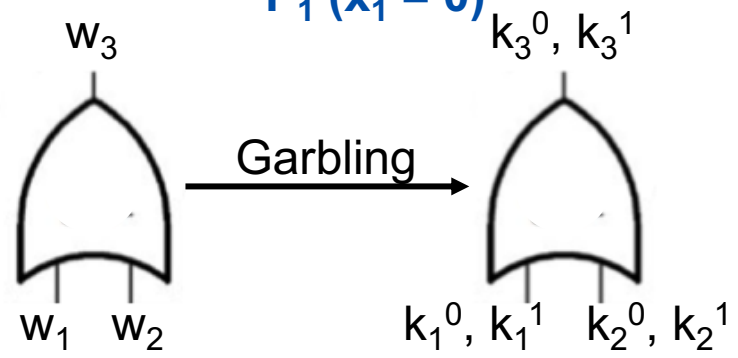
Garbled Circuits [Yao '82]

$f(x_1, x_2)$



Garbled Circuits contd.

P_1 ($x_1 = 0$)



w_1	w_2	w_3	Garbling
0	0	0	$G_1 = E(k_1^0, k_2^0, k_3^0)$
0	1	1	$G_2 = E(k_1^0, k_2^1, k_3^1)$
1	0	1	$G_3 = E(k_1^1, k_2^0, k_3^1)$
1	1	1	$G_4 = E(k_1^1, k_2^1, k_3^1)$

G_2, G_1, G_3, G_4

$(k_3^0, 0), (k_3^1, 1)$

k_1^0

OT for k_2^1

Output = 1

P_2 ($x_2 = 1$)

1. Try to decrypt $G_1, \ldots, G_4$.
2. With k_1^0 and k_2^1, can decrypt only G_2 to obtain k_3^1.
3. k_3^1 maps to 1, so the output is 1.

Why are the last two problems still open?

- There are more than 250 million vehicles in the US, and each one of them needs at least 1000 certificates per year.
- Even with the state of the art, garbled circuits for linkage value generation would be more than 5000 Exabytes in size and require more than 1 million high-end computers to generate them.
  - Google has only 15 Exabytes of data!
  - 1 Exabyte = 1024*1024*1024 Gigabytes.

Thank you!

We have opening for interns!