Cadence -
A Simulator for Human
Movement-based
Communication Protocols

Harel Berger, Georgetown University
Micah Sherr, Georgetown University
Adam Aviv, The George Washington University
The Worst Part of Censorship is
Motivation

- 64% of internet users express concern over governmental censorship (Go Globe).
- The Internet can be disrupted in times of crisis, either due to natural (e.g., natural disasters) or human causes (e.g., war).
Anti-Censorship Methods

• Alternate names and addresses (DNS)
• Mirrors, Proxies
• Traffic obfuscation
• Sneakernets
Sneakernets

- USB/BLE/NFC/Mesh Wifi

- Encounters between physical devices.

- The most well known uses of sneakernets were in Iraq and Hong Kong, 2014.
Why Sneakernets?

- Bypassing Internet Censorship
- Anonymity and Privacy
- Limited Digital Footprint
- Reduced Dependency on Online Infrastructure
Sneakernets – How it works?

From: Alice
To: Charlie
Content: Let’s go outside!
Sneakernets – How it works?

Alice

Bob
Sneakernets – How it works?
Sneakernets – How it works?
Sneakernets – How it works?
Open Questions

- When should Alice transfer A to Bob?
- When should Bob transfer A to Charlie?
- How the addressing works?
- How the communication works?
Cadence

- Simulator for human movements-based sneakernet.
- Utilizes different routing algorithms to simulate message transferring.
- Allows online visualization of a current simulation.
• **Protocol expressiveness**
  – Simple yet flexible API that enables researchers to concisely specify their routing algorithms.

• **Behavior modeling**
  – Different types of nodes (participants), including potentially malicious actors are simulated.
• **Mobility modeling**
  – Simulate messaging in a dynamic network consisting of mobile, human users.

• **Metrics**
  – Collect statistics throughout the simulation and features an interactive reporting module.
• **Performance**
  – Operate on computing platforms with varying resources.
  – Golang (strong support of parallelism)

• **Repeatability and scientific validity**
  – Repeatable experiments and deterministic execution.

• **Free, open source software**
Lens

- Easy interface for handling lots of different datasets.

```go
type Lens interface {
    Init(logger *logger.Logger)

    Import(path string, dataSetName string) error

    // gets the type of location used for this dataset
    GetLocationType() model.LocationType
}
```
Lens

- Implemented lenses for:
  - Geolife
  - Cabspotting
  - MDC
  - Disasters datasets (partial)
• Linear movement model
  – Given a node $n$ and two consecutive events $(t_i, \vec{v}_i), (t_j, \vec{v}_j)$;
    - $t_i < t_j$
    - $\vec{v}_x$: location of the node at time $t_x$
  – Cadence infers the location $\vec{v}_\alpha$ of the node at $t_\alpha$, where $t_i < t_\alpha < t_j$ as:

$$
\vec{v}_\alpha = \vec{v}_i + (t_\alpha - t_i) \frac{\vec{v}_j - \vec{v}_i}{t_j - t_i}.
$$
Physics Engine

- Defines possible encounters between two nodes:
  - Bounded by a conditions file (JSON)

```json
{
    "conditions" : [
        {
            "name" : "distance200m",
            "type" : "distance",
            "params" : {
                "dist" : 200.0
            }
        },
        {
            "name" : "probability25",
            "type" : "probability",
            "params" : {
                "prob" : 0.25
            }
        }
    ]
}
```
Physics Engine

- Defines possible encounters between two nodes:
  - Determines the closest point between any two nodes during a given time period.
  - Checks the conditions for this point. If the conditions are met, encounter.
Routing

• Address versatility:
  – Unicast
  – Geocast

• Routing is controlled by a Logic engine
Logic Engine

- Implement the following interface:

```go
type Logic interface {
    Init(log *logger.Logger)

    // stores the initial copy of a message at a node
    PlaceMessage(id model.NodeId, message *Message)

    // callback function for every time a node finds itself in a new position
    // `b` is the marshalled form of the location
    NewPositionCallback(nodeId model.NodeId, t model.LocationType, b []byte)

    // potentially does a message exchange between nodes when an encounter occurs.
    HandleEncounter(encounter *model.Encounter,
        messageDBChan chan *model.MessageDB,
        receivedmessageDBChan chan *model.DeliveredMessageDB) float32

    // ... supporting routines omitted for brevity ...
}
```
Logic Engine

• **PlaceMessage** - Bootstraps messaging by specifying the origination of a message.

• **NewPositionCallback** is called whenever a node is in a new position.
  – Useful for simulating certain protocols (e.g., Aviv et al.’s HumaNets protocol).

• **HandleEncounter** governs how an encounter should be handled.
Logic Engine (Flooding)

**PlaceMessage**: put a message in a node’s queue.

**NewPositionCallback**: do nothing

**HandleEncounter**:

![Diagram](image)
Logic Engine

• **General factors**
  – Distance
  – Probability of an Encounter (global or per message)
  – Connection Duration

• **Device factors**
  – Battery Consumption
  – Memory Usage
Behavior Modeling

• Attacker nodes
  – Flooder
  – Dropper

```go
type Attacker interface {
  Init(log *logger.Logger)

  // runs an attack of the attacker node, using its queue of messages, 
  // during an encounter
  Attack(encounter *model.Encounter, attacker model.NodeId, 
           attackerMap *sync.Map)

  // ... supporting routines omitted for brevity ...
}
```
Behavior Modeling

• Flooder
  – Injects spurious messages whenever it meets a new node.
  – A sufficiently large flooder population can create a version of the Coremelt attack.
Behavior Modeling

• **Dropper**
  – Does not pass any message to any node and instead drops every message it receives.
  – Creates a typical DOS attack.
Reporting Module

• A web service
• Allows researchers to run various reports and plot.
• Currently supported reports include plots of delivery rates, contact rates, and node lifetimes.
Reporting Module

• Adding a new report requires registering a unique URL (e.g., “/avg-throughput”) and an event handler.

• Formatting reports is accomplished through Golang’s built-in HTML template language.
Reporting Module

Messages Transfer Times

- **broadcast_no_constraints_narrow_times**

![Graph showing messages transfer times](image)
Future Work

• Optimization
  – Running dense datasets require enormous computational resources
    o Moving to Encounter-based simulator instead of events based.
  – Relational DB backend (i.e., MySQL)
    • Moving to memory-backed data stores (e.g., Redis).
Future Work

• Protocols
  – New routing protocols
  – Currently, the simulator supports only a single configuration per execution.
    o Different protocols in parallel, ”parallel worlds”.

• Documentation of new routing protocols and node behaviours.
Summary

• Cadence is a simulator for human movement-based communication.
• It supports different datasets, routing algorithms, addresses and node behaviors.
• Cadence is an ongoing project, and will be extended in the future.
Availability

• Cadence is available as free and open-source software and can be downloaded from [https://github.com/GUSecLab/cadence](https://github.com/GUSecLab/cadence).
Acknowledgments

• This material is based upon work supported by DARPA under Contract No. FA8650-22-C-6424.

• Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of DARPA.
Any Questions?