# Wireless Network Security

## Spring 2016

Patrick Tague

Class #3 – Project Discussion;
OMNET++ Intro

# Waitlists

- If you're currently registered for this class, but not planning to stay: **please drop**

- If you're currently on the waitlist:
  1) Make sure you're on the correct waitlist (see the previous slide)
  2) Send me an email (tague@cmu.edu) detailing:
     1) **What year/term** of your program are you in? Priority will go to students closer to graduation.
     2) **What degree requirements** (if any) does this course fulfill?
     3) **Why** you want to take this course?
     4) **What prereqs/qualifications** do you have?

# Class #3

- Detailed discussion of course project

- Tutorial: Intro to OMNET++

# Projects

# Project Goals

- The course project provides an opportunity to apply topics from class to an in-depth study of a specific topic area
  - Not just a broad survey of what has been done
  - A chance to do something novel and make a real contribution to advance the state of the art
  - Aim to submit a conference paper, poster, or demo

- Experience with an end-to-end project
  - Ideation, survey, hypothesis, experimentation, analysis, and presentation of process and results

# Project Deliverables

- Four presentations:
  - Project intro presentation – in class **Feb 2 & 4** (depending on how many teams we end up with)
  - Statement of Work presentation – in class **Feb 25** (these are short)
  - Progress presentation – in class **Mar 31** (also short)
  - Final presentation – in class **Apr 26 & 28**
- Two reports:
  - Statement of work – due **Feb 25**
  - Final report – due **May 5**

# Intro Presentation

- Presentation of project area, potential project topic, and background / related work
  - What is the broadly defined problem? Why is it interesting? What has been done so far? What questions have not yet been answered?
  - Presentation should include figures to illustrate the problem idea, approach, etc. in a straightforward way (i.e., not a lot of text)
  - Ideally, plan for 6-8 slides with a total duration around 15 minutes per team (including every team member)

# SoW Presentation

- In-class presentation of statement of work:
  - High-level presentation of problem statement, project goals, approach, outcome/deliverables, etc.
  - Description of tasks and timeline
  - Outline of experimentation plan

  - Presentation template will be provided (likely 2 slides per team – one for a nice figure to illustrate the problem, one to summarize the above)

# Statement of Work

- Written SoW document
  - Due the same day as the SoW presentation

  - Essentially a much more detailed written version of your SoW, including similar content to the presentation
    - Problem statement, motivation, challenges, figures, expected outcomes / deliverables, timeline, experimentation plan
    - Include related work to frame your contributions against

  - MAX 2 page, IEEE 2-column conference format

# Deliverable Grading

- Presentations:
  - Grade will be based on 1) whether you included everything that we asked you to include, 2) use of the time allotted, 3) balance of presenters across team, 4) clarity of presentation

- Reports:
  - Grade will be based on 1) clear presentation of project aspects, 2) inclusion of all necessary components, 3) use of figures, data, etc. as appropriate

# Project Support

- Each project will have an advisor/mentor
  - Any faculty member, researcher, or suitable PhD student can mentor a project – let me know if you want to arrange an external project sponsor

- Some hardware can be made available for experimentation

# Project Teams

- Forming teams and choosing topics:
  - These two things are not independent
  - Try to choose team members with common interests, different backgrounds, etc., **not just your friends**
  - Multiple teams cannot work on the same project

# What topic should I choose?

# Project Topics

- Projects must:
  - Relate to systems covered in class and focus on some aspect of wireless network security
  - Strive for new research/development contributions – plan to submit a conference paper, poster, or demo
  - Not be a project you're working on for another course (no double-dipping)

- Examples of past projects

# Example Project I

- Project from Spring 2014: “Analysis of location privacy provided by mix-zones in VANET”

# Example Project I

## Proposed Solution - Changing pseudonyms in mix-zones

- Vehicles entering a mix-zone stop sending messages and change their pseudonyms.
- They resume communication after leaving the zone.
- Hence vehicles simultaneously exiting a mix-zone form an anonymity group.

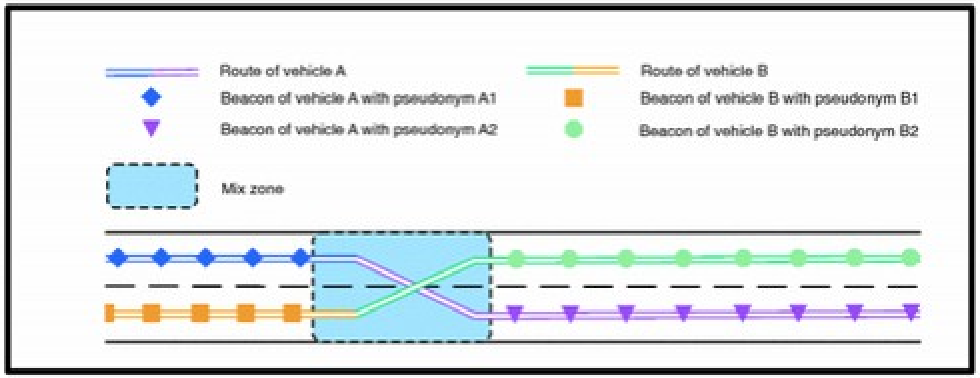
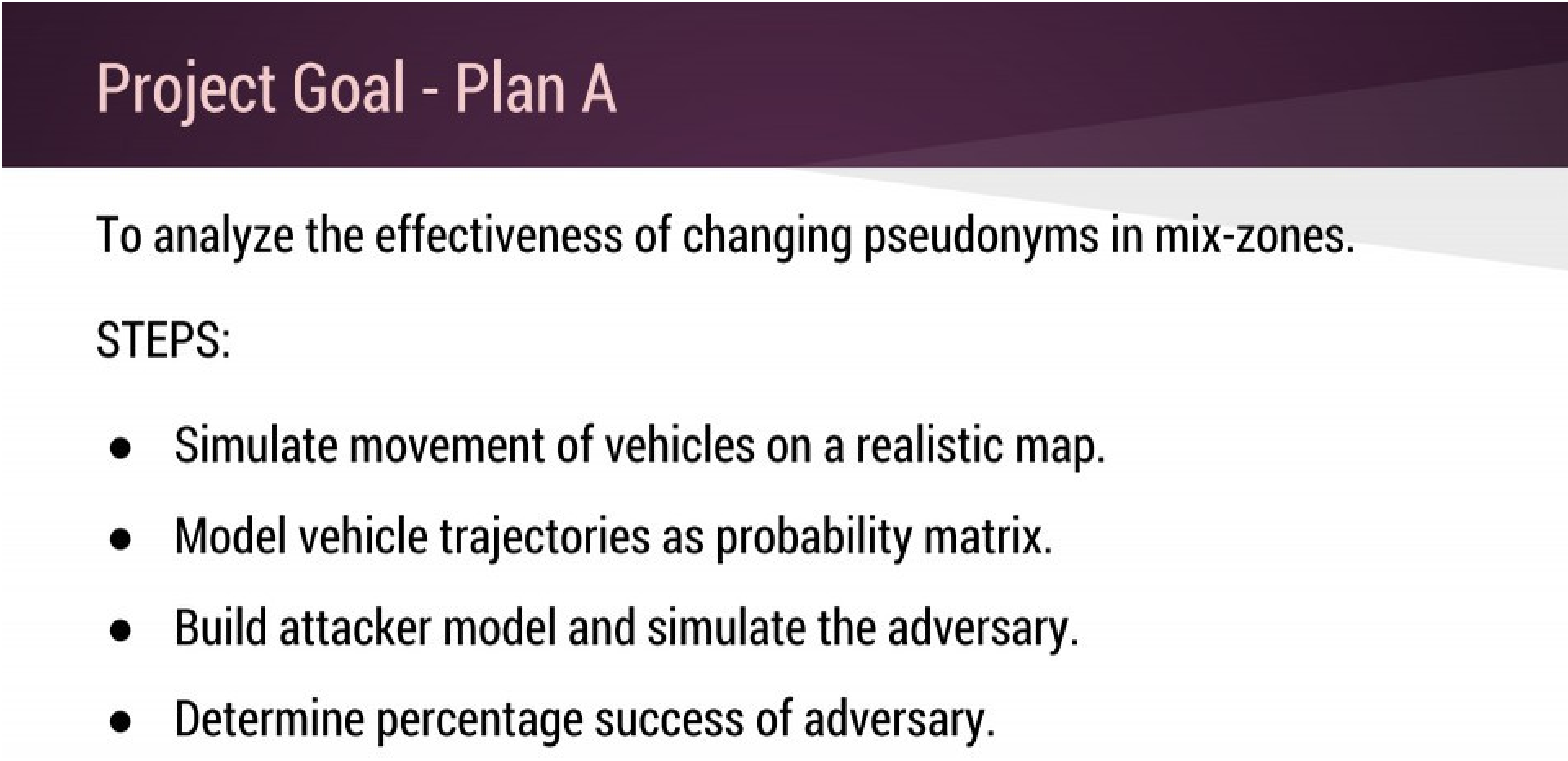

Route of vehicle A
Beacon of vehicle A with pseudonym A1
Beacon of vehicle A with pseudonym A2
Route of vehicle B
Beacon of vehicle B with pseudonym B1
Beacon of vehicle B with pseudonym B2
Mix zone

Fig 3: A pseudonym change inside a mix-zone.

# Example Project I



## Project Goal - Plan A

To analyze the effectiveness of changing pseudonyms in mix-zones.

STEPS:

- Simulate movement of vehicles on a realistic map.
- Model vehicle trajectories as probability matrix.
- Build attacker model and simulate the adversary.
- Determine percentage success of adversary.

# Analysis Of Location Privacy provided by Mix-Zones in VANETs

Ramya Balaraman, Vedant Bhatt, Venkatesh Sriram
{rbalaram,vbhatt,vsriram}@andrew.cmu.edu

## Background

The nature of communications in VANETs makes it very easy for an attacker to compromise the location privacy of a vehicle. Changing pseudonyms in mix-zones is the major solution proposed to combat this problem. We implement a so-called 'transition attack' on this solution.

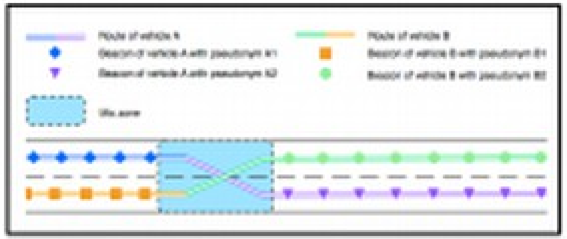

Route of vehicle A
Beacon of vehicle A with pseudonym A1
Beacon of vehicle A with pseudonym A2
Mix zone
Route of vehicle B
Beacon of vehicle B with pseudonym B1
Beacon of vehicle B with pseudonym B2

Fig. 1. Pseudonym changes in a mix-zone

## Mix-Zone simulation

We used SUMO (Simulation for Urban Mobility) to simulate the movement of vehicles at a mix-zone which is usually positioned at highway intersections. We settled upon a particular intersection in Mountain View as our candidate of choice for the mix-zone.

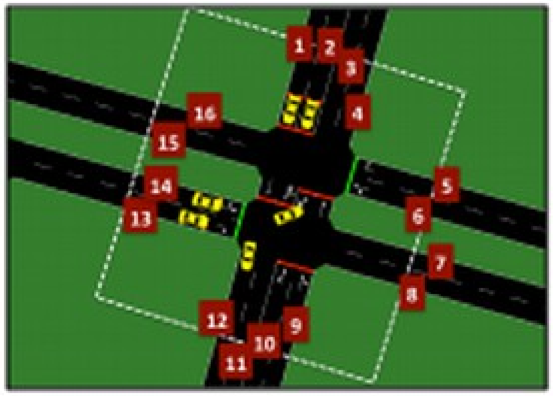


Fig. 2. Representation of mix-zone with lane numbers.

## Attacker Simulation

The attacker studies vehicle transit patterns at the mix-zone and creates a probability matrix characterizing the vehicular movement. This is a 2D matrix with rows representing entry lane numbers, and columns representing exit lane numbers, with reference to Fig. 2.

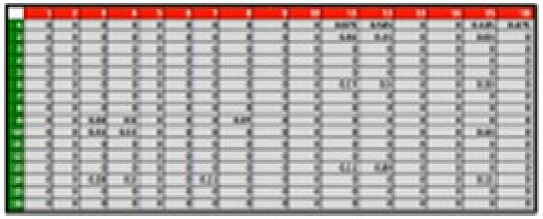


Fig. 3. Probability matrix representation of mix-zone traffic.

The attack consists of two steps-

1) Match entry lanes to exit lanes using knowledge of vehicle transit patterns i.e. essentially the probability matrix.
2) Chronological timing attack on the unmatched pairs.

## Results

We tested our implementation of the transition attack with test data from the SUMO mobility trace. Our algorithm maps entry lanes to exit lanes given lists of entry and exit lanes. The percentage success in this correlation is used as a metric to determine the success of the attack.

| Input Lane | Predicted Output | Actual Output | Correct? |
|---|---|---|---|
| 6 | 15 | 15 | YES |
| 9 | 4 | 4 | YES |
| 10 | 3 | 3 | YES |
| 1 | 12 | 12 | YES |
| 9 | 4 | 4 | YES |
| 6 | 12 | 12 | YES |
| 2 | 11 | 11 | YES |

100% success

Fig. 4. Attacker output given a list of 7 entry & exit events.

| Input Lane | Predicted Output | Actual Output | Correct? |
|---|---|---|---|
| 9 | 4 | 4 | YES |
| 10 | 13 | 13 | YES |
| 9 | 4 | 4 | YES |
| 1 | 16 | 16 | YES |

100% success

Fig. 5. Attacker output given a list of 4 entry & exit events.

| Input Lane | Predicted Output | Actual Output | Correct? |
|---|---|---|---|
| 1 | 12 | 12 | YES |
| 2 | 11 | 11 | YES |
| 1 | 11 | 12 | NO |
| 14 | 4 | 4 | YES |
| 6 | 12 | 12 | YES |
| 9 | 4 | 4 | YES |
| 10 | 15 | 15 | YES |
| 9 | 8 | 8 | YES |
| 2 | 12 | 11 | NO |
| 10 | 3 | 3 | YES |

80% success

Fig. 7. Attacker output given a list of 10 entry & exit events.

## Conclusions and Future Work

The attacker can successfully use his knowledge of the mix-zone to perform a combination of transition and timing attacks. This can help him correlate entry and exit events, thereby subverting location privacy provided by changing pseudonyms in mix-zones.

Scaling the attacker model to incorporate larger, real world data sets would be the next step going forward. The attack can also be made more robust by incorporating behavior modeling and machine learning into the attack.

### References

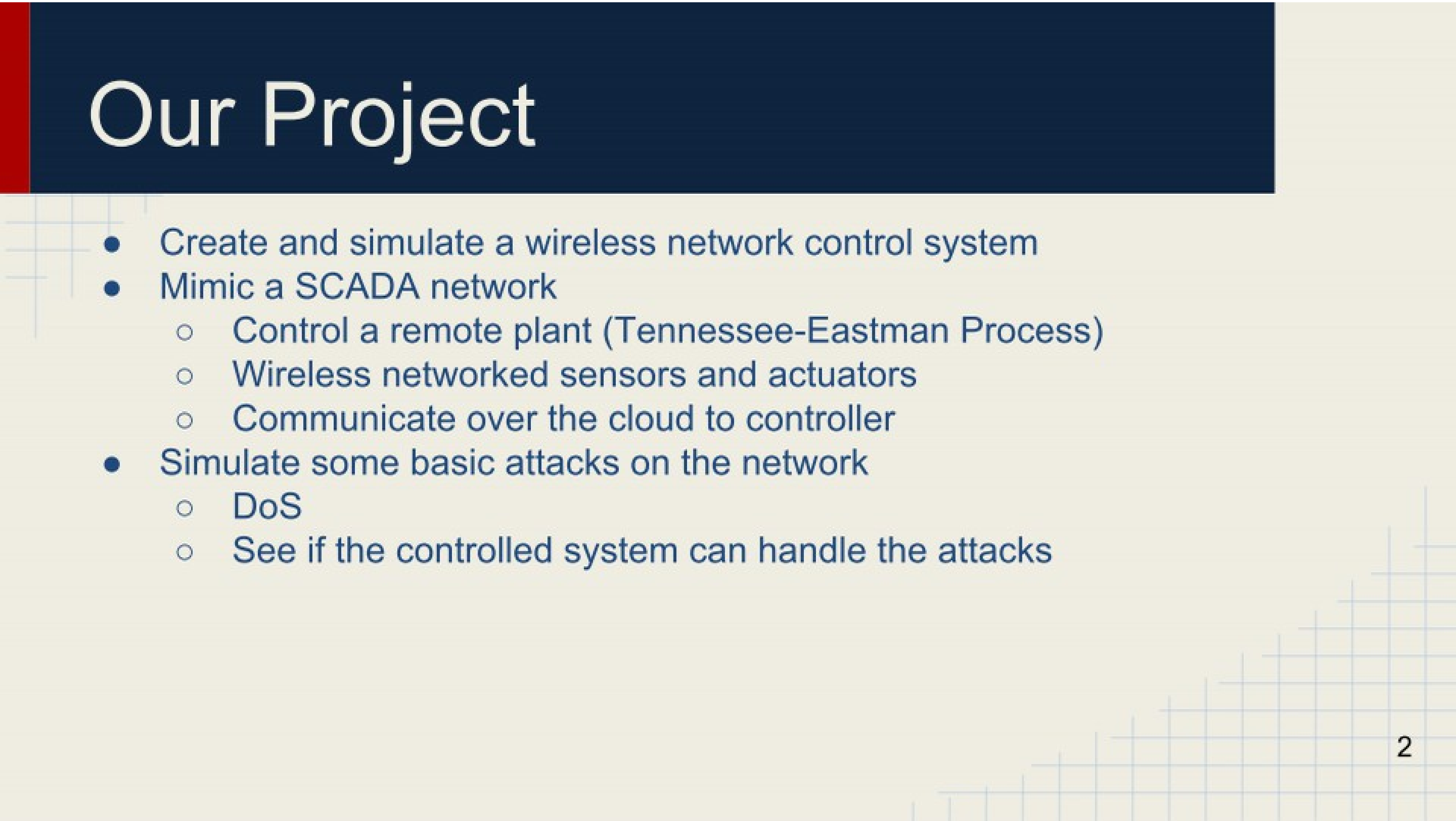
[1] Scheuer, Florian, Karl-Peter Fuchs, and Hannes Federrath. "A safety-preserving mix zone for VANETs." *Trust, Privacy and Security in Digital Business* (2011)

[2] Palanisamy, Balaji et al. "Location privacy with road network mix-zones." *Mobile Ad-hoc and Sensor Networks (MSN), 2012 Eighth International Conference* on 14 Dec. 2012

# Example Project II

- Project from Spring 2014: “Cross-layer Mischief in Networked Control Systems”

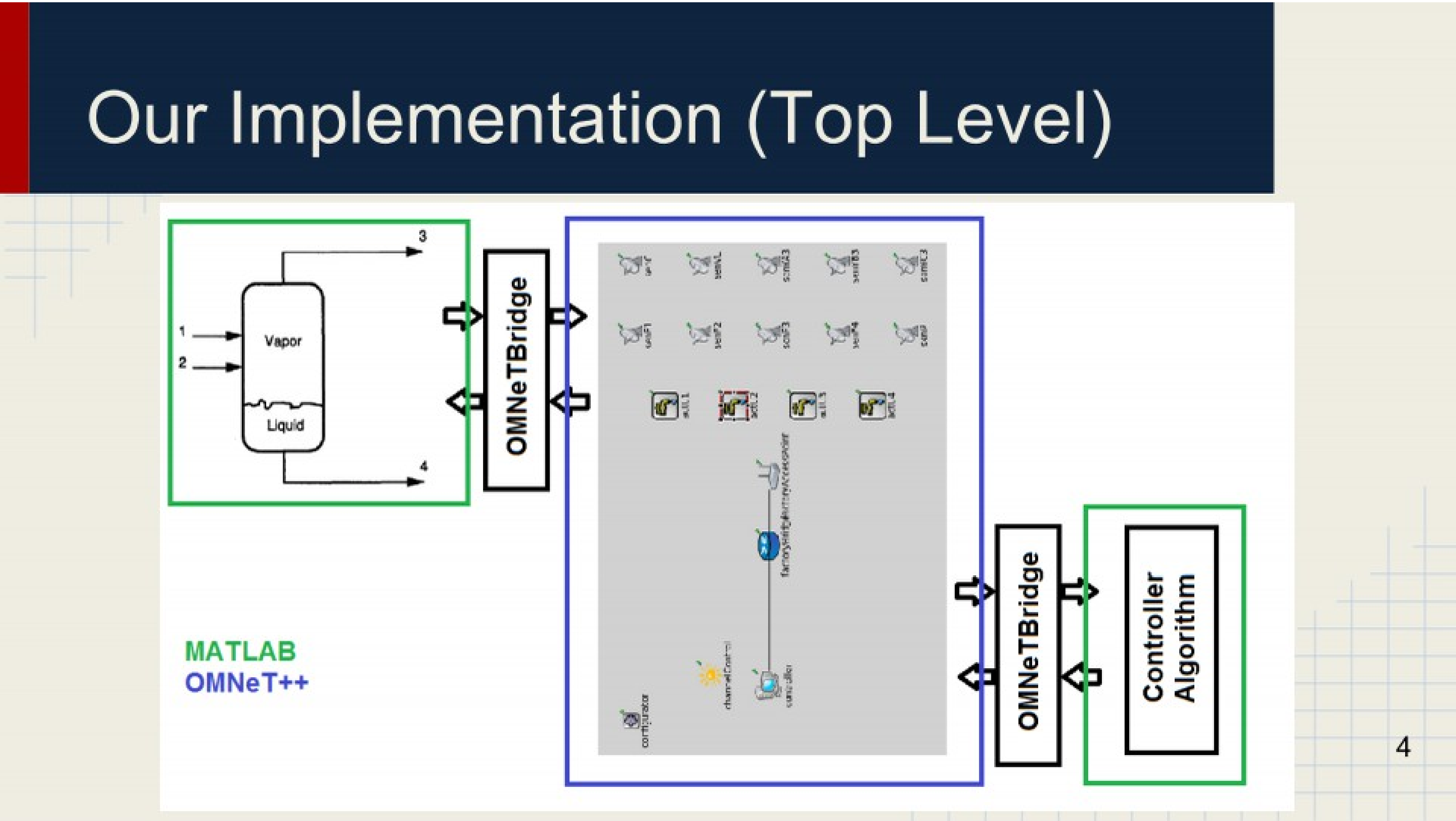
# Example Project II



## Our Project

- Create and simulate a wireless network control system
- Mimic a SCADA network
  - Control a remote plant (Tennessee-Eastman Process)
  - Wireless networked sensors and actuators
  - Communicate over the cloud to controller
- Simulate some basic attacks on the network
  - DoS
  - See if the controlled system can handle the attacks

# Example Project II



## Our Implementation (Top Level)

# Create-Your-Own Project

- Teams are free to come up with their own project idea, as long as the project is approved by me prior to the first presentation
  - Topics must be appropriate for the course
  - Project scope must be appropriate for one semester

# Projects Available

- I may have a few projects available that would be mentored by my senior PhD students.
  - If so, I'll make announcements and provide relevant information via Blackboard

# Timeline & Some Tips

- We highly recommend starting early and not working according to the presentation and report timeline
  - Don't wait until right before the deadline to make progress!
  - Try to get started right away, get some preliminary results, figure out if project scope needs to change, get a good working demo early, then iterate and improve

# HW/SW Resources

- Don't be afraid of hardware
  - We have some fun stuff you can experiment with
  - We will consider reasonable requests for additional hw

- Take advantage of existing projects / tools
  - OMNET++ has a lot of add-on packages that can be very useful, and other tools exist as well (don't limit yourself to OMNET++ just because we say so)
  - Many researchers publish their tools, simulation kits, etc. so you may not need to start from scratch
  - We will consider reasonable requests for sw licensing

# Questions about Projects?

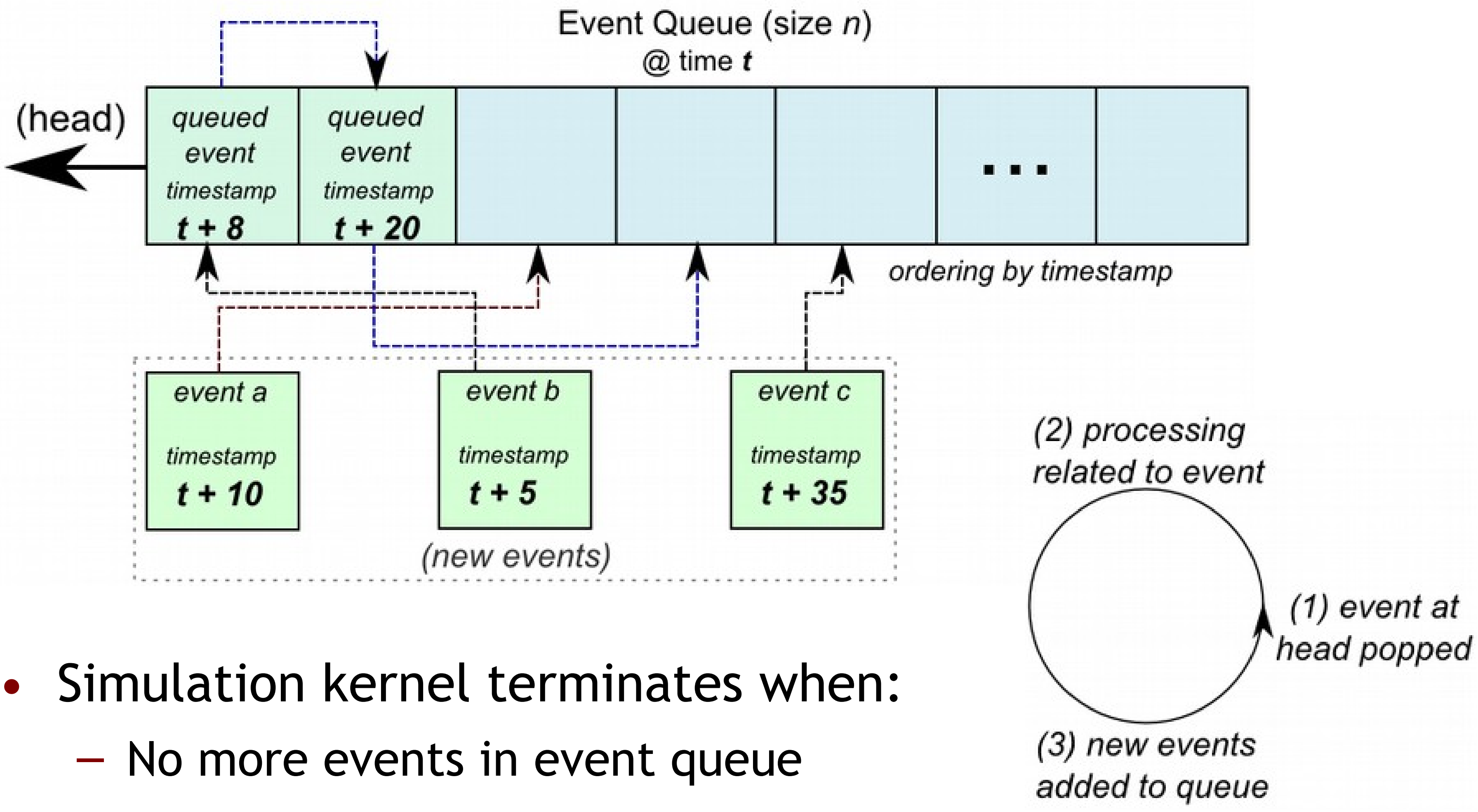
# Intro to OMNET++/INET

# What is OMNeT++?

- It's a discrete event simulator that provides a base for “network” simulation
  - Communication networks
  - Queuing networks
  - Digital logic networks
  - '...' networks

- Two components
  - Event-driven simulation kernel
  - Utility classes
    - Implementations of common functionality for network simulations
      - Math functions
      - Statistics
      - Physical network characteristics helper classes
      - ...

# Simulation Kernel


Event Queue (size *n*)
@ time ***t***
(head)
queued event timestamp ***t + 8***
queued event timestamp ***t + 20***
. . .
ordering by timestamp
event a timestamp ***t + 10***
event b timestamp ***t + 5***
event c timestamp ***t + 35***
(new events)
(2) processing related to event
(1) event at head popped
(3) new events added to queue

- Simulation kernel terminates when:
  - No more events in event queue
  - Termination condition reached
  - User terminates

# INET

- Communication networks simulation package for OMNeT++
  - Provides models for many wired/wireless networking protocols
  - These models build upon each other to create simulation models of communication nodes, and networks
  - Gives OMNeT++ communication networks support without us having to write our own protocols

- Easy to install – comes bundled with OMNET++

# Simulation Models

- A simulation model consists of *modules*, which are grouped/connected together.
  - Modules that are grouped together are themselves modules
  - Provides a module hierarchy

- In OMNeT++, a simulation model is also called a *network*
  - A network (simulation model) is itself a module

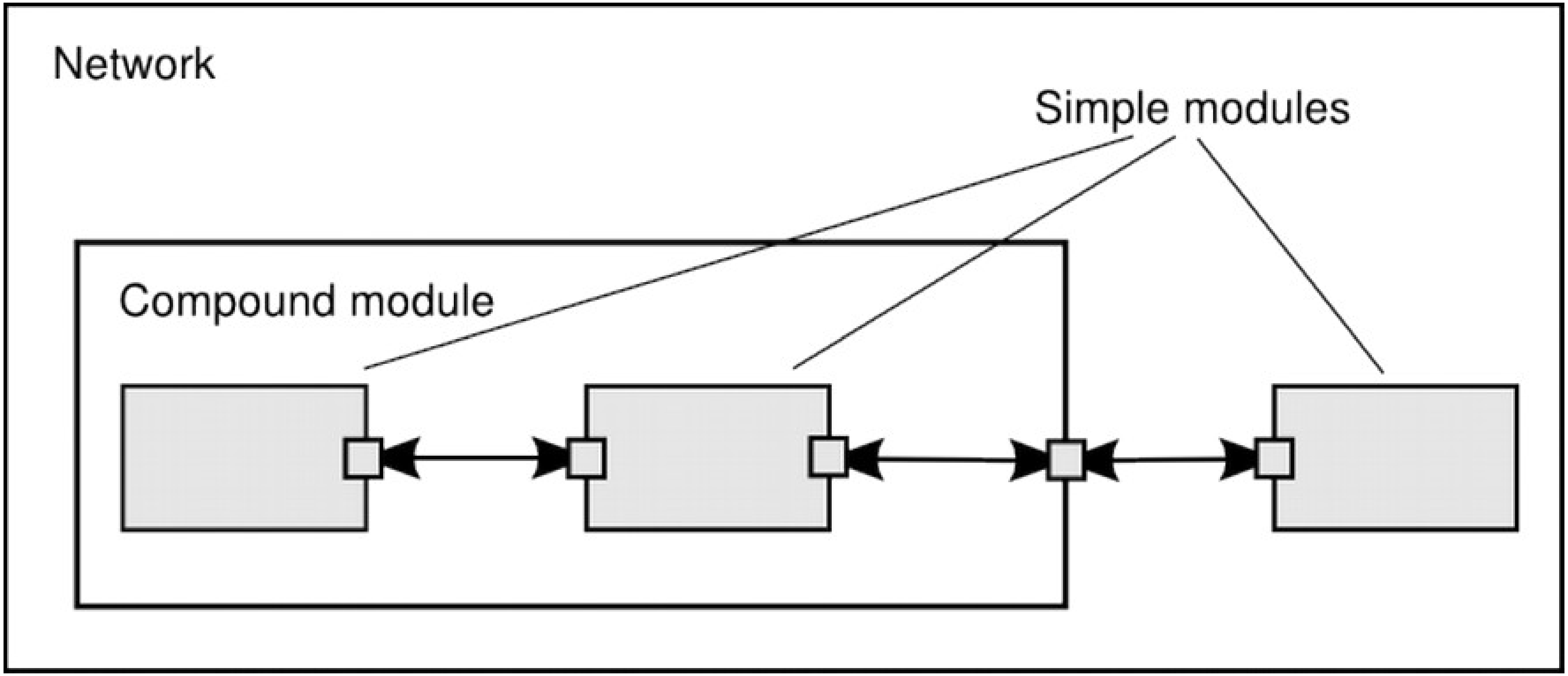
# Module Types

- What goes into a simulation model?
  - It all starts with *Simple Modules*
    - Base building blocks
    - Declared using the NED language
    - Backed by C++ classes which define their behavior
    - Defines parameters to pass to (C++) implementation
  - Simple modules group together to form *Compound Modules*
    - Declared using the NED language
    - Defines parameters to pass to simple modules

# Simulation Models - Gates

- *Gates* allow for message passing
  - *Messages* pass between gates using *connections*
    - Two gates can be directly linked via connection
      - Think wired communication network
    - Connections can also be used to directly pass a message to an unlinked gate
      - Think wireless communication network
  - Connections can be defined and reused
    - Called *channels*

# Simulation Models



- http://www.omnetpp.org/doc/omnetpp/manual/usman.html#sec101

# Statistics Collection

- Output vectors
  - Time-series data
  - Stuff that gets recorded during a simulation

- Output scalars
  - Aggregate stuff recorded at the end of a simulation
  - Mean of something, std dev of something, …

# Statistics Collection

- Declaring Statistics
  - In NED:
    - @statistic[stat_name](properties)
      - stat_name = variable emitted from the C++ class
      - properties = what to record, and in which form (scalar, vector)
    - @statistic[received_pkt](record=sum,vector?)
      - received_pkt is a variable emitted each time a packet is received
      - We are recording to a scalar the total packets received
      - We are recording to a vector each time a packet is received
      - Note the '?' - this means its optional

# Statistics Collection

- Emitting variables (signals)
  - http://omnetpp.org/doc/omnetpp/manual/usman.html#sec193
  - Register the signal by name
    - registerSignal(“stat_name”)
      - stat_name must match that given in the NED declaration
      - Function returns an id for the signal
  - Emit the signal when appropriate
    - emit(signal_id, value)
      - signal_id = id of signal (mapped to stat_name)
        - *simsignal_t signal_id = registerSignal("stat_name")*

# Example Time

inet/examples/wireless/hosttohost/

# January 21:
Physical Layer Threats; Jamming