Virtual Time Consistency in Smart Grid Test-beds

David M. Nicol
Franklin W. Woeltge Professor of ECE
Director, Information Trust Institute
What purpose for a Test-bed?

Create a model of a physical system that

- Captures salient features of interest
- Can be observed through controlled presentation of inputs or boundary conditions
Test-bed Behavior

We want the test-bed to “act like” the system it models

- Present some input
Test-bed Behavior

We want the test-bed to “act like” the system it models

• Input presented to some test-bed component
Test-bed Behavior

We want the test-bed to “act like” the system it models

• Initiates some sequence of actions and inputs/outputs
Test-bed Behavior

We want the test-bed to “act like” the system it models

• Initiates some sequence of actions and inputs/outputs
Test-bed Behavior

We want the test-bed to “act like” the system it models

- Initiates some sequence of actions and inputs/outputs
Test-bed Behavior

We want the test-bed to “act like” the system it models

• Initiates some sequence of actions and inputs/outputs
Test-bed Behavior

We want the test-bed to “act like” the system it models

• Initiates some sequence of actions and inputs/outputs
Test-bed Behavior

We want the test-bed to “act like” the system it models

• Output observed

One wants the components to interact in the same way, with the same input/outputs, as in the field, to get the same output
Test-bed Behavior

We want the test-bed to “act like” the system it models

• Output observed

As in the field, sequencing in the test-bed is governed by real-time delays
Test-bed Behavior

We want the test-bed to “act like” the system it models
• Output observed

As in the field, sequencing in the test-bed is governed by real-time delays
Our View of a Smart Grid Test-bed

**Devices**
- Meters, relays, PMUs, data aggregators, adaptive multi-channel source, etc.

**Software**
- Control station, data historian, authentication servers, etc.

**Power System Simulators**
- Hardware assisted (e.g., RTDS, Opal-RT)
- Software only, PowerWorld, GridLab-D, OpenDSS, PSAT, etc.

**Device Emulation**
- Xen, QEMU, LXC

**Network Emulation**
- Emulab, CORE, Deter

**Device/Network Simulation**
- ns-3, S3F, OmNet++, etc.
Keeping it all coordinated...

*Time* matters a lot

Normal emulation execution is *best effort*

Suppose in the modeled system 3 devices all send messages to the same router *at the same time*

- The VMM sends them when it gets around to it...
Keeping it all coordinated...

*Time* matters a lot

Normal emulation execution is *best effort*

Suppose in the modeled system 3 devices all send messages to the same router *at the same time*

• The VMM sends them when it gets around to it...
Keeping it all coordinated...

_Time_ matters a lot

Normal emulation execution is *best effort*

Suppose in the modeled system 3 devices all send messages to the same router _at the same time_

- The VMM sends them when it gets around to it...
Keeping it all coordinated...

*Time* matters a lot

Normal emulation execution is *best effort*

Suppose in the modeled system 3 devices all send messages to the same router *at the same time*

- The VMM sends them when it gets around to it...
Keeping it all coordinated...

*Time* matters a lot

Normal emulation execution is *best effort*

Suppose in the modeled system 3 devices all send messages to the same router *at the same time*

- The VMM sends them when it gets around to it...
Keeping it all coordinated...

*Time* matters a lot

Normal emulation execution is *best effort*

Suppose in the modeled system 3 devices all send messages to the same router *at the same time*

- The VMM sends them when it gets around to it...
Keeping it all coordinated...

*Time* matters a lot

Normal emulation execution is *best effort*

Suppose in the modeled system 3 devices all send messages to the same router *at the same time*

- The VMM sends them when it gets around to it...
Keeping it all coordinated...

*Time* matters a lot

Normal emulation execution is *best effort*

Suppose in the modeled system 3 devices all send messages to the same router *at the same time*

- The VMM sends them when it gets around to it...
Keeping it all coordinated...

*Time* matters a lot

Normal emulation execution is *best effort*

Suppose in the modeled system 3 devices all send messages to the same router *at the same time*

- The VMM sends them when it gets around to it...
Keeping it all coordinated...

*Time* matters a lot

Normal emulation execution is *best effort*

Suppose in the modeled system 3 devices all send messages to the same router *at the same time*

• But if synchronized in *virtual time*
Keeping it all coordinated...

*Time* matters a lot

Normal emulation execution is *best effort*

Suppose in the modeled system 3 devices all send messages to the same router *at the same time*

- But if synchronized in **virtual time**
Keeping it all coordinated...

*Time* matters a lot

Normal emulation execution is *best effort*

Suppose in the modeled system 3 devices all send messages to the same router *at the same time*

- But if synchronized in **virtual time**
Keeping it all coordinated...

*Time* matters a lot

Normal emulation execution is *best effort*

Suppose in the modeled system 3 devices all send messages to the same router *at the same time*

• But if synchronized in **virtual time**
Keeping it all coordinated...

*Time* matters a lot

Normal emulation execution is *best effort*

Suppose in the modeled system 3 devices all send messages to the same router *at the same time*

• But if synchronized in **virtual time**
Keeping it all coordinated...

*Time* matters a lot

Normal emulation execution is *best effort*

Suppose in the modeled system 3 devices all send messages to the same router *at the same time*

• But if synchronized in *virtual time*
Keeping it all coordinated...

*Time* matters a lot

Normal emulation execution is *best effort*

Suppose in the modeled system 3 devices all send messages to the same router *at the same time*

• But if synchronized in *virtual time*
Keeping it all coordinated...

*Time* matters a lot

Normal emulation execution is *best effort*

Suppose in the modeled system 3 devices all send messages to the same router *at the same time*

- But if synchronized in **virtual time**
Keeping it all coordinated...

*Time* matters a lot

Normal emulation execution is *best effort*

Suppose in the modeled system 3 devices all send messages to the same router *at the same time*

- But if synchronized in *virtual time*
Keeping it all coordinated...

*Time* matters a lot

Normal emulation execution is *best effort*

Suppose in the modeled system 3 devices all send messages to the same router *at the same time*

- But if synchronized in **virtual time**, **dispatch is concurrent**

![Diagram](image)
Test-bed Behavior

We want the test-bed to “act like” the system it models

- Output observed

As in the field, sequencing in the test-bed is governed by real-time delays
Test-bed Behavior

We want the test-bed to “act like” the system it models

• Output observed

To sequence as in the field, we have to sequence with respect to Virtual Time
Imagine the Possibilities....

With a test-bed embedded in virtual time, you can

- **model larger systems** on smaller test-beds
  - More simulation, fewer devices

- **Mask latencies in test-bed federation**
  - Run xN slower, turns 50ms real delay into a $50/N$ ms virtual delay
  - Synchronization
Virtual Time Sequencing

Every component action needs a **time-stamp**
Every component action needs a **time delay**

Virtual time management framework
- Schedules component actions in virtual time
- Manages inter-component input/output
Virtual Time Sequencing

Every component action needs a time-stamp
Every component action needs a time delay
Virtual time management framework
• Schedules component actions in virtual time
• Manages inter-component input/output

Sometimes known as Discrete-Event Simulation

Questions
• How to embed emulation in VT?
• How to embed power system flow simulation in VT?
• How to embed device execution in VT?
• How to coordinate it all?
Embedding Emulation in Virtual Time

Requires

• calls to clock return *virtual* time
  – Based on measured execution and time dilation factor

• Scheduling
  – Advance virtual machines concurrent w.r.t. virtual time

Examples

• Versions of Xen just shift off clock bits
  – TDF of 2,4,8, etc.
  – Ordinary VM scheduling

• Timekeeper scales time by TDF
  – Advances all LXCs paced by slowest container
Network Simulation/Emulation Coordination

Requires synchronization

• Virtual machines and simulators need to advance at the same rate
  – Best effort interactions within that

• Fine grained synchronization
  – Ensure that no simulator or VM receives a time-stamped communication “in its past”
  – Achieved using synchronization protocols from parallel discrete-event simulation
    • S3F and Timekeeper
Power System Flow Simulators and Virtual Time

Requirements

• Export state, with virtual time-stamps
• Pause/Restart, or scaled release execution
• Buffer state
Device Execution and Virtual Time

No “always works” methodologies

• But some important special cases
  – PMU
Device Execution and Virtual Time

Possible approaches

• Tinker with the input
Device Execution and Virtual Time

Possible approaches

• Tinker with the input
• Tinker with the output
Assembling Pieces of the Puzzle

Heavy-weight VM farm (OS images)

Lightweight Linux Protocol stacks (TimeKeeper)

Combined Physical / Virtual Network S3F

Testing Devices

Smart Grid Appliances

Power Simulation

SG Device Emulation

SG Device Simulation

Network topologies
Layer-1/2/3 Wireline and wireless

Traffic generation
PC device emulation

High speed protocol traffic generation
Fuzzers
Examples
Example Use of a Smart Grid Testbed

DDoS Attack Using C12.22 Trace Service in AMI

C12.22 Trace Service

Request: 26 26+R1 26+R1+R2 26+R1+R2+R3

Response: 26+R1+R2+R3+R4

• Amplification
  – Increased volume of traffic
• Reflection
  – Spoofed source address

Components used
• Meter emulation
• Meter simulation

• Access point simulation
• Zigbee wireless simulation

Might have included power simulation but wasn’t needed
Experiment

- 4x4 blocks, 448 meters
- 5 attackers
- Victim: the single egress point (meter gateway)
- ZigBee wireless network, 1 Mb/s bandwidth
- Normal traffic: 100-byte packet per 10 second
- Attacking traffic: 200 times faster, 15-30 hops
Experimental Results

B1. $r_c$ - channel contention (normal)

C1. $r_l$ - packet loss (normal)

B2. $r_c$ - channel contention (attacking)

C2. $r_l$ - packet loss (attacking)
Example: Attack on Situational Awareness

View of power system state seen at Test-bed Control station
- Its view derived from network simulation
Example: Attack on Situational Awareness

Control Station polls system devices for status and state.
Example: Attack on Situational Awareness

Control Station polls system devices for status and state.
Example: Attack on Situational Awareness

Control Station polls system devices for status and state.
Example: Attack on Situational Awareness

Control Station polls system devices for status and state.
Example: Attack on Situational Awareness

Control Station polls system devices for status and state.
Example: Attack on Situational Awareness

Control Station polls system devices for status and state
Example: Attack on Situational Awareness

1. Holes in FW permit outbound http connection to site with hidden malware
Example: Attack on Situational Awareness

2. HMI monitor infected, maps PCS, IDs devices
Example: Attack on Situational Awareness

Power System
- OSI Control Station
- Data Aggregator
- RTU
- relay
- AMS

Test-bed Interface
- Control Station
- Substation
- Substation

Data Converter

3. Network map returned to malware master
Example: Attack on Situational Awareness

Test-bed
Power System
OSI Control Station
Data Aggregator
RTU
relay
AMS

Control Station
Substation
Substation
Substation
Substation

INTERNET

3. Malware master returns infection program to compromised host
3. Compromised host infects others, creates botnet
Example: Attack on Situational Awareness

4. Attack on relay causes line to open
Flow topology changes
Example: Attack on Situational Awareness

5. Simultaneous launch of bandwidth consumption attack by botnet blinds control station
Conclusion

Virtual time consistency in Smart Grid test-beds
• Allows greater flexibility in what can be studied

Integration of device/communication simulation + emulation well underway

Integration of power flow simulation with device/communication simulation has been accomplished
• Flow simulator needs special hooks

Integration of devices with virtual time is a Work in Progress