Supervisory Parameter Adjustment for Distribution Energy Storage (SPADES) DOE CESER - CEDS Program

Subtask 3.2 - Red Team Attack Tests
Attack KPIs (Qualitative)

Tier 1
- Power Delivery Disruption
- Instability (Oscillation)
- Voltage Imbalance
- Substation power factor

Tier 2
- Equipment useful life degradation
- Power Quality degradation (poor power factor or over/undervoltage conditions)
Attacks Associated to Tier 1 KPIs

• Power Delivery Disruption:
  • DER disconnection based on IEEE 1547
  • Line overloading (decrease in DER output or increase in load)
  • Transformer overloading

• Voltage oscillation:
  • Aggressive settings of volt-var, volt-watt curves and interaction with voltage regulators
  • Quick connect/disconnect of DER’s and Loads or change of setpoints
  • Topology reconfiguration
    • Between feeders or within feeders
  • Repeated operation enable/disable regulator, capbanks or tap changes

• Voltage Imbalance:
  • Connect/disconnect single phase DERs/loads to create imbalance

• Substation Power Factor:
  • Connect/disconnect loads, DERs, capbanks
Attack Tests
Attack Tests Implementation

• Testing of all envisioned attacks have been performed in OpenDSS
  • Validation of attacks and their consequences
• Some of those attacks have also been implemented as prototypes in PyCIGAR
  • Useful for definition of best ways of integrating attacks to the framework
• Functionalities included in PyCIGAR
  • Flexibility to define start and end times independently for multiple devices and multiple attacks of same device
  • Definition of multiple types of attacks to each device
  • Implementation of classes corresponding to hacked devices/controllers
  • Additional input file for defining attacks that may include complex coordination of multiple devices in multiple points in time
  • Wrapper of input parser for processing of attack inputs and integrating them to the simulation
    • Also employed for processing computer network device information and creation of NetJSON representation.
Topology Reconfiguration Attack

• Attack Scenario:
  • Over/under voltage type PDD attack
  • Step 1: Open normally closed (NC) line or sectionalizing switch
  • Step 2: Close NO switch for radial topology reconfiguration (intra or inter-feeder)
  • Step 3: Repeat 1-2 to cause voltage oscillations (optional)

• Simulation Model (OpenDSS)
  • IEEE 37 bus

• Attack implementation in PyCIGAR
  ✓ Implementation of Hacked controllers/devices
    ▪ (Hacked switch controller)
  ✓ Attack parameters: Change topology for defined time
  ✓ Results: Allow the change of topology based on predefined topologies in PyCIGAR
Tests Performed
• Close 741-720 and Open 702-713

Test Results (explanation, outcome of tests)
• Feeder is longer after reconfiguration and experiences lower voltages towards the end of the feeder
Tests Performed
Modify basic topology of IEEE 3 network by utilizing a hacked switch
• Close Switch S2 and open Switch S1
Attack is executed twice

Test Results (explanation, outcome of tests)
• Load S703 is connected to Transformer
  ➢ Voltage of Load S703 increases
Load/DER Disconnect Attack

- Attack scenario
  - Voltage Imbalance type attack
  - Selective load shedding/increase on a single phase to worsen phase imbalance
  - Repeated actions could also cause oscillations (optional)

- Simulation Model (OpenDSS)
  - IEEE 37 bus

- Attack implementation in PyCIGAR
  - Implementation of Hacked controllers/devices
    - (Hacked Load Device and Hacked load controller)
  - Downscale load in IEEE 3 bus system
  - Attack parameters: scaling of load according to provided scaling factor for a defined time
  - Result: Allow the scale loads at any node in PyCIGAR

To be done:
- Discuss and implement phase specific downscaling
- Test implementation for single phase downscaling
Load Disconnection Attack - Voltage Imbalance

Tests Performed
- Open S701a, S714a, S738a

Test Results
- Disconnecting several large single-phase loads worsens imbalance between red and blue phases. Combining with topology reconfiguration also degrades the voltage.
Load/DER Disconnect Attack – Load scaling in PyCIGAR

Tests Performed
• Downscale the load of Load S701 in the IEEE 3 network according to the attack input
• Attack input is a scaling factor of 0.9 in the first attack and 0 in the second attack

Test Results
➢ Voltage on node S701 increases depending on the amount of down scaling.
➢ Second attack represents a load drop.

Next steps:
▪ Explore Load scaling for different phases

![Original Simulation (IEEE 3 with no initial scaling factors applied and 500 simulation steps)](image1)

![Simulation with Downscaling attack on node S701. (IEEE 3 with no initial scaling factors applied and 500 simulation steps)](image2)
Regulator Attack

- Attack scenario
  - Over/under voltage type PDD attack
  - Disable regulator, or reverse delay settings for multiple regulators
  - Repeated actions could also cause oscillations (optional)

- Simulation Model (OpenDSS)
  - IEEE 37 bus, IEEE 123 bus

- Attack implementation in PyCIGAR
  To be done:
  - Extend regulator class
  - Implement hacked component classes
  - Extend red team parser
  - Define attack parameters
  - Test attacks
Regulator Attack (Disable)

Tests Performed
• Disable regulator

Test Results (explanation, outcome of tests)
• Disabled regulator brings down the voltages for the entire feeder (IEEE 37 Bus)

- With regulator
- Regulator disabled
Regulator Attack (Reverse time delay)

Tests Performed
• Change the delay setting of substation LTC to act after line regulators

Test Results (explanation, outcome of tests)
• Reverse delay, i.e., substation LTC has a longer delay than line regulators raise voltages

• With regulator
• Reverse delay on regulator
Regulator Attack (Reverse power)

Tests Performed
• Reverse power flow through regulator (RevRegTest.dss)

Test Results (explanation, outcome of tests)
• Reverse power through regulator pushes up voltages if reversible setting for regulator is not enabled

• With regulator, reversible=yes, revneutral=yes

• Reverse power through regulator, no reverse regulator setting
Regulator Attack (Reverse power)

Tests Performed
- Reverse power flow through regulator (RevRegTest.dss)

Test Results (explanation, outcome of tests)
- Reverse power through regulator changes voltages drastically if reversible=yes for regulator, but revneutral=no
- With regulator, reversible=yes, revneutral=no
- With regulator, reversible=yes, revneutral=no + reverse delay
Capbank Attack

- Attack scenario
  - Over/under voltage type PDD attack
  - Disable regulator, or reverse delay settings for multiple regulators
  - Repeated actions could also cause oscillations (optional)

- Simulation Model (OpenDSS)
  - IEEE 123 bus

Attack implementation in PyCIGAR
To be done:
- Identify and define or extend the necessary classes
- Implement hacked component classes
- Extend red team parser
- Define attack parameters
- Test attacks
Capacitor Attack (Disable)

Tests Performed
• Disable capbank – can modify capcontrols as well, IEEE 123 bus

Test Results (explanation, outcome of tests)
• Disabling capbank reduces feeder voltage

- With capbank
- Capacitor disabled
Energy Storage Attack

- Attack scenario
  - Manipulate active/reactive power setpoints

- Simulation Model (OpenDSS)
  - TBD

- Attack implementation in PyCIGAR
  - Implementation OF hacked classes
  - Extended red team parser

To be done:
- Define attack parameters and scenarios
- Integrate attack specifics in hacked classes
- Test attacks
Attack Costs/Budget
NetJSON to overlay Computer Network information

- Features
  - Configuration of devices
  - Monitoring data
  - Network topology
  - Routing information
- Adds the ability to define the IP network communication paths and firewall/access rules
NetJSON Generation

Network Specific Devices and Connections (csv)

Power System Devices (PyCIGAR)

Input Parser

NetJSON Output
### Sample Input files

- `device_a, device_b, property_dict`
- `inverter_s701a, network_switch1, {"type": "wired"}`
- `network_switch1, network_ids1, {"type": "wired"}`
- `network_ids1, network_firewall1, {"type": "wired"}`
- `network_firewall1, controlcenter, {"type": "wired"}`
- `inverter_s702a, network_switch2, {"type": "wired"}`
- `network_switch2, network_firewall2, {"type": "wired"}`
- `network_firewall2, controlcenter, {"type": "wired"}`
- `inverter_s703a, network_switch3, {"type": "wired"}`
- `network_switch3, network_firewall3, {"type": "wired"}`
- `network_firewall3, controlcenter, {"type": "wired"}`
- `ntp_clock1, network_switch1, {"type": "wired"}`
- `hackme_wifi, network_switch1, {"type": "wireless"}`

### Resulting NetJSON (Output)

```json
{
  "type": "NetworkGraph",
  "label": "Devices",
  "protocol": "static",
  "version": null,
  "metric": null,
  "nodes": [
    {
      "id": "inverter_s701a",
      "properties": {
        "type": "pv_device"
      }
    },
    {
      "id": "inverter_s702a",
      "properties": {
        "type": "pv_device"
      }
    },
    {
      "id": "inverter_s703a",
      "properties": {
        "type": "pv_device"
      }
    },
    {
      "id": "controlcenter",
      "properties": {
        "type": "control_center"
      }
    },
    {
      "id": "network_switch1",
      "properties": {
        "type": "switch"
      }
    },
    {
      "id": "network_ids1",
      "properties": {
        "type": "ids"
      }
    },
    {
      "id": "network_firewall1",
      "properties": {
        "type": "firewall"
      }
    },
    {
      "id": "network_switch2",
      "properties": {
        "type": "switch"
      }
    },
    {
      "id": "network_firewall2",
      "properties": {
        "type": "firewall"
      }
    },
    {
      "id": "network_switch3",
      "properties": {
        "type": "switch"
      }
    },
    {
      "id": "network_firewall3",
      "properties": {
        "type": "firewall"
      }
    },
    {
      "id": "ntp_clock1",
      "properties": {"type": "ntp_clock"}
    },
    {
      "id": "hackme_wifi",
      "properties": {"type": "wifi_router"}
    }
  ],
  "links": [
    {
      "source": "inverter_s701a",
      "target": "network_switch1",
      "properties": {
        "type": "wired"
      }
    },
    {
      "source": "network_switch1",
      "target": "network_ids1",
      "properties": {
        "type": "wired"
      }
    },
    {
      "source": "network_ids1",
      "target": "network_firewall1",
      "properties": {
        "type": "wired"
      }
    },
    {
      "source": "network_firewall1",
      "target": "controlcenter",
      "properties": {
        "type": "wired"
      }
    },
    {
      "source": "inverter_s702a",
      "target": "network_switch2",
      "properties": {
        "type": "wired"
      }
    },
    {
      "source": "network_switch2",
      "target": "network_firewall2",
      "properties": {
        "type": "wired"
      }
    },
    {
      "source": "network_firewall2",
      "target": "controlcenter",
      "properties": {
        "type": "wired"
      }
    },
    {
      "source": "inverter_s703a",
      "target": "network_switch3",
      "properties": {
        "type": "wired"
      }
    },
    {
      "source": "network_switch3",
      "target": "network_firewall3",
      "properties": {
        "type": "wired"
      }
    },
    {
      "source": "network_firewall3",
      "target": "controlcenter",
      "properties": {
        "type": "wired"
      }
    },
    {
      "source": "ntp_clock1",
      "target": "network_switch1",
      "properties": {
        "type": "wired"
      }
    },
    {
      "source": "hackme_wifi",
      "target": "network_switch1",
      "properties": {
        "type": "wireless"
      }
    }
  ]
}
```
Sample NetJSON Node Graph

- Node graph created based on the sample input file
- Contains devices and link
- In practical applications, device properties will contain information such as:
  - Attack costs
  - Communication paths
  - Firewall rules
Contact

Bruno Paes Leao
Siemens Corporation, Technology
755 College Road East
Princeton, NJ 08540
USA

E-Mail: bruno.leao@siemens.com

Web: https://www.siemens.com/research