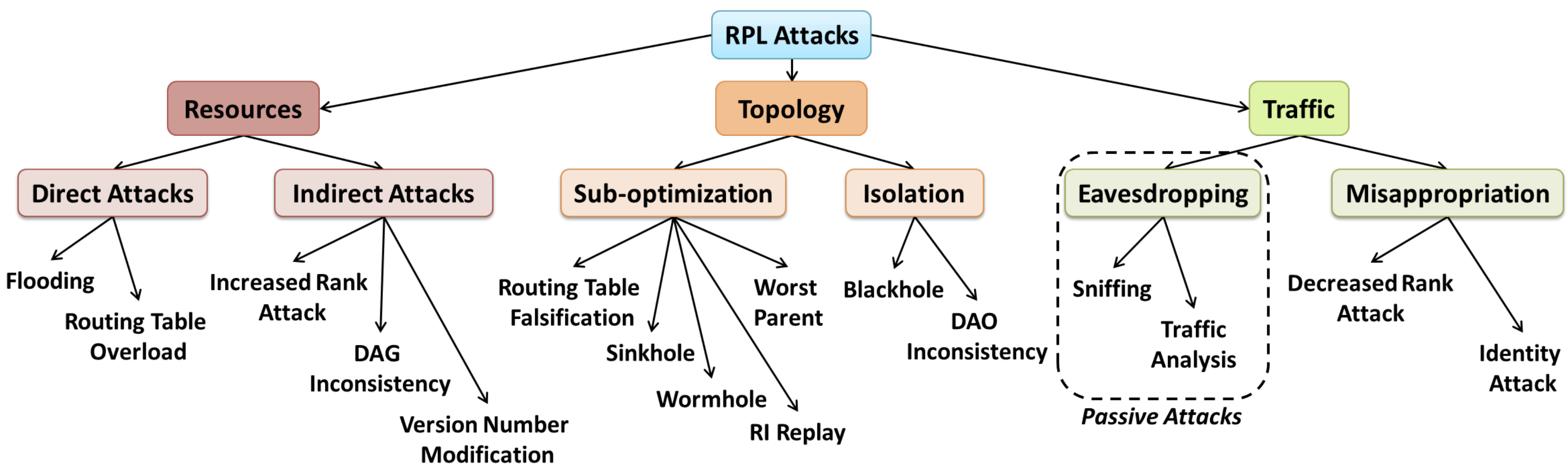
DODAG Version Number Attack Tutorial

Resource Attacks

- Resource attacks are one category of security attacks on the RPL protocol, as shown in the taxonomy below
  - Their purpose is the exhaustion of node or network resources, e.g., via an overload on power consumption, memory, etc.



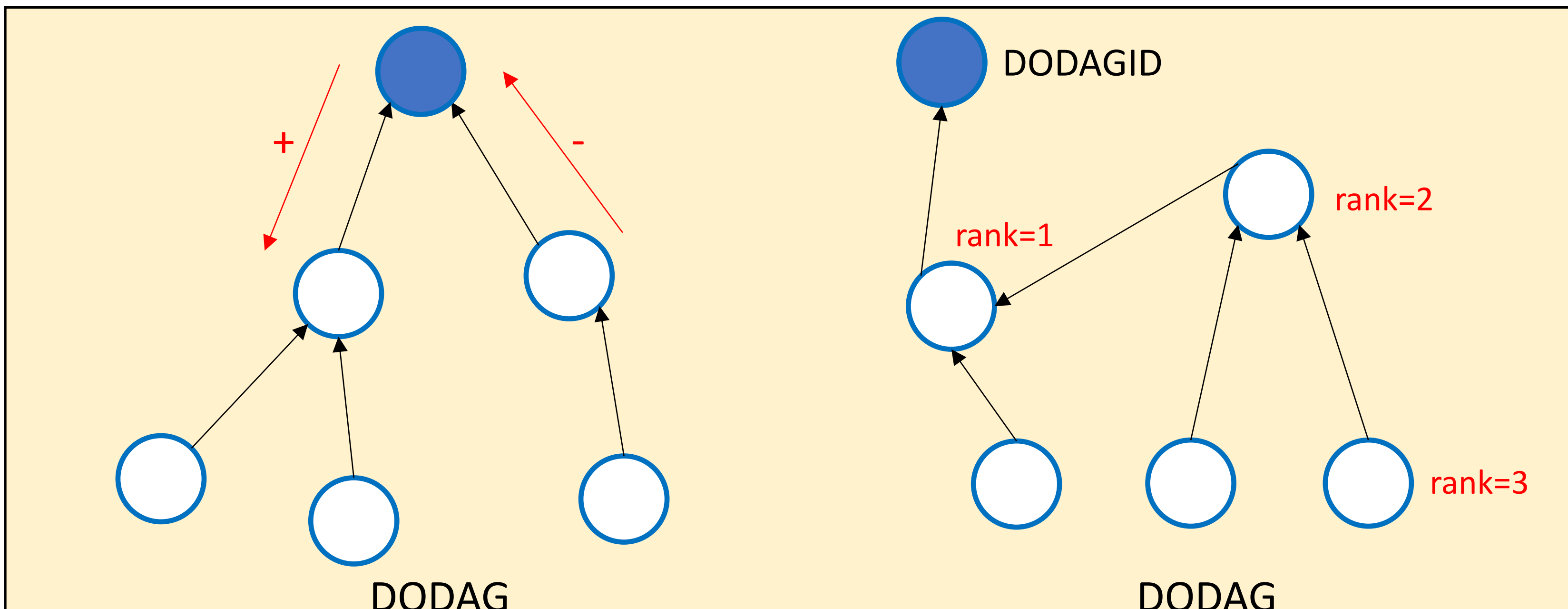
Source: https://hal.inria.fr/hal-01207859/document

Resources Attacks (cont.)

- Resources attacks can be done by forcing legitimate nodes to perform unnecessary actions to increase their use of resources
  - Have impact on the availability of the network by congesting available links or by incapacitating nodes, thus may also influence the lifetime of the network
- Indirect attacks are one category of resource attacks in which the malicious node provokes the other nodes to generate the overload
  - The DODAG version number attack is an example in this category that we shall address in more detail next

Review of the DODAG Version Number and Rank Mechanism

- To identify and maintain a network topology, RPL uses the DODAG Version Number and Rank mechanism
  - DODAG Version is a specific DODAG iteration with a given id, and the Version Number is a sequential counter incremented by the DODAG root
  - DODAG Rank defines the node's individual position relative to other nodes with respect to a DODAG root



DODAG Version Number Attack

- In RPL, the version number parameter is used as a global repair indicator, and should only be altered by the root of the DODAG to signal the need for topology reconstruction
  - However, there is no security mechanism to protect this parameter from malicious modifications
- An attacker can change the version number by illegitimately increasing the value of this field of DIO messages when it forwards them to its neighbors
  - Such an attack is called DODAG version number attack, and it causes an unnecessary rebuilding of the whole DODAG graph, thus wasting nodes' resources

DODAG Version Number Attack Simulation

DODAG Version Attack Simulation

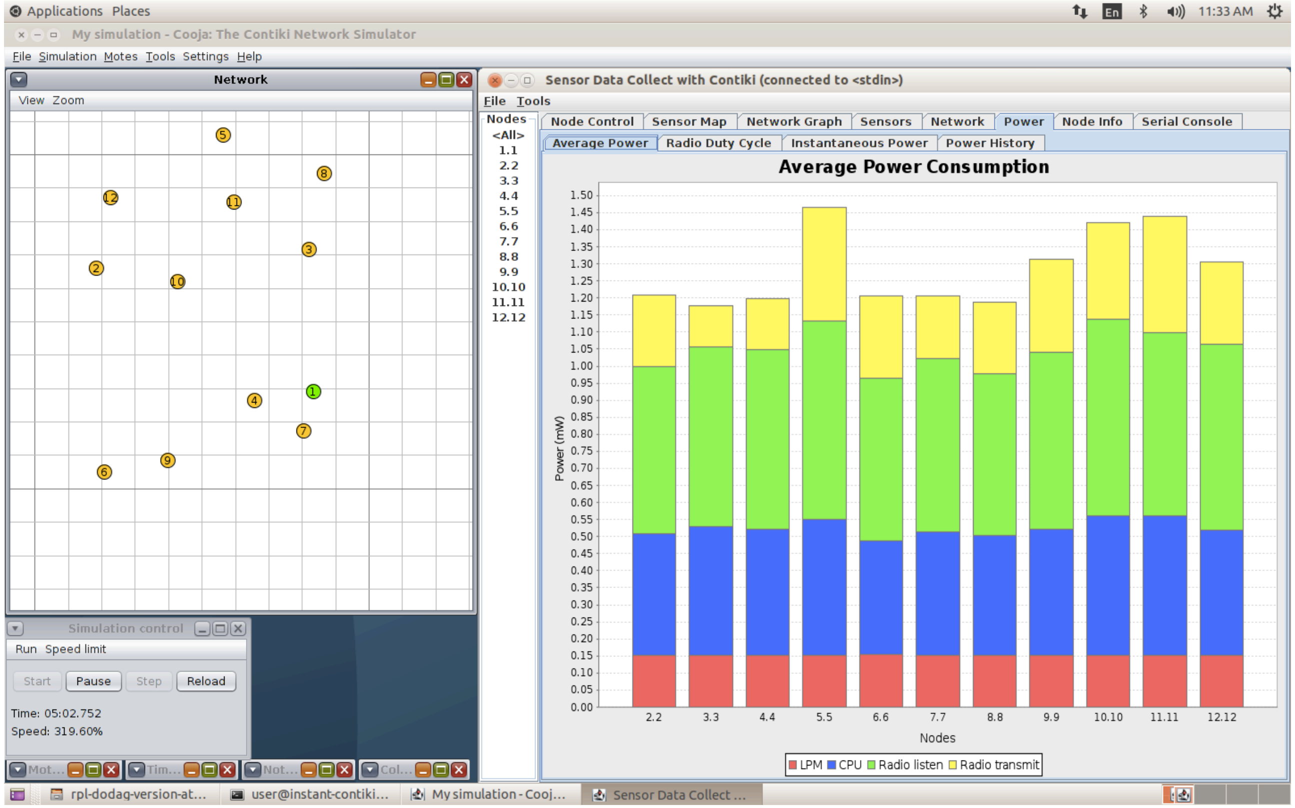
- Open the desired simulation in Cooja by selecting the corresponding scenario via the IoTrain-Sim interface
  - We recommend that you first select the “Reference Scenario Simulation” entry to view the reference scenario
- Alternatively, the simulations can be opened manually as follows
  - In Cooja, select the menu File > Open simulation > Browse…
  - Go to the folder “iotrain-sim/database/security_training/dodag_version_attack/simulation/”
  - Select “dodag_attack-reference.csc” for the reference scenario, and click “Open”

DODAG Version Attack Simulation (cont.)

- Simulation and data collection procedure
  1. In the CollectView window, click on the “Start Collect” button, then click on the “Send command to nodes” button
  2. In the Simulation control window of Cooja, click on the “Start” button to begin the simulation
  3. Wait for at least two minutes of simulation time
  4. Back in the CollectView window, go to the Power tab and see the Average Power plot for the scenario
- Follow the same procedure to perform the attack simulation and compare the results
  - The attack scenario can be opened via the menu “DODAG Version Attack Simulation” in IoTrain-Sim, or directly in Cooja via the file “dodag_attack-simulation.csc”
  - You may need to wait for more than five minutes of simulation time to get statistics for all the nodes

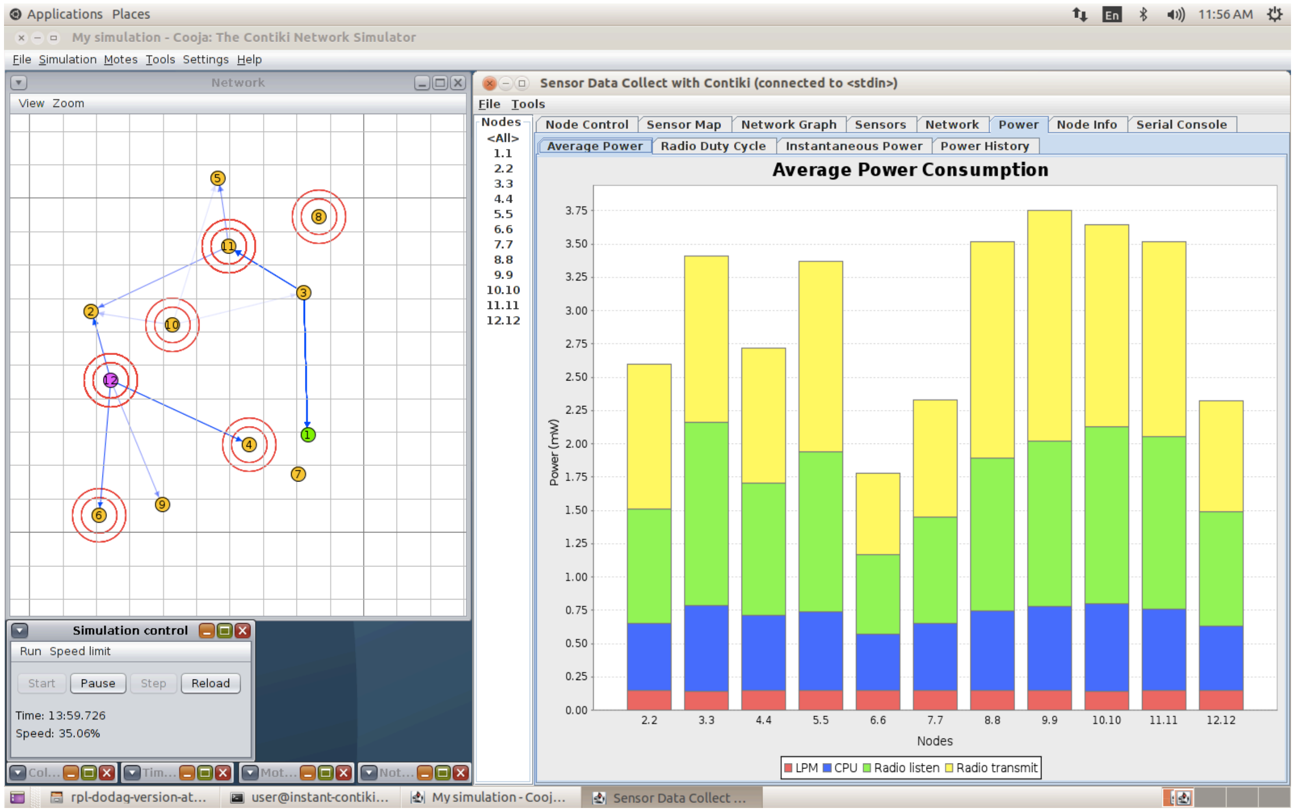
Reference Scenario and Results

- Node 1 (green color) is a SINK node that acts as a border router
- The other nodes are sender nodes that act as normal sensors



Attack Scenario and Results

- Node 1 and the nodes in yellow color have the same roles as before
- Node 12 became a malicious node performing a DODAG version attack, with effects on most of the other nodes



Discussion

- Reference scenario
  - All the sender nodes (nodes 2 to 12) have nearly the same average power consumption, which is at a low level of around 1.2 mW
- Attack simulation
  - Because of the continuous global repair needed to reconstruct the network topology, the power consumption of all the nodes increases, typically by a factor of 2 to 3, reaching even values of 3.75 mW

DODAG Version Number Attack Implementation

Implementation Overview

- To implement the DODAG version number attack, some changes are necessary to the normal source code for the RPL implementation in Contiki
- The file to be modified is located in the directory "contiki/core/net/rpl/"
  - rpl_icmp6.c, which manages the input and output for RPL control messages

Changes to rpl-icmp6.c

- The file “rpl-icmp6.c” includes a function that constructs the DAG object, and one of the stored data items is the DAG version number
- To implement the DODAG version attack, one can increment internally the DAG version variable
  - This will cause the protocol to continuously try to recompute the network topology

```
/* DAG Information Object */
pos = 0;

buffer = UIP_ICMP_PAYLOAD;
buffer[pos++] = instance->instance_id;
buffer[pos++] = dag->version++;//added '++' after 'version', in order to increment version and provoke global repair
```

DODAG version number attack implementation by incrementing the DAG version number for each DAG packet

Exercises

- After making the suggested modifications in a copy of the Contiki source code, compile the files and assign the resulting malicious firmware to one of the motes in the reference scenario
- We suggest you use node 12 first as malicious one, as in our example, then change the malicious node to another one and see how the results change
  - You can also use multiple malicious nodes and compare the simulation results