TinyWIDS: a misuse-based Intrusion Detection System for IEEE 802.15.4 Wireless Sensor Networks

Authors:
Walter Tiberti, Luigi Pomante

University of L'Aquila - DISIM - Centre of Excellence DEWS

Università degli Studi de L'Aquila

Dipartimento di Ingegneria e Scienze dell'Informazione e Matematica

Introduction

- **WSN** — A *Wireless Sensor Network* (WSN) is a network composed of a high number (often heterogeneous) of small, energy- and resource constrained nodes called *motes*.
- The reference standard for WSN communication is the *IEEE 802.15.4*
- **Security** — Providing *security* features in a WSN is *challenging*:
  - Small storage available (50-100 KB of ROM, 8-12 KB of RAM)
  - Small and slow micro-controller unit (MCU) (e.g. Atmel, MSP430)
  - Vulnerable communication channels (2.4 GHz ISM bands)
- **IDS** — An *Intrusion Detection System* (IDS) is a component whose purpose is to detect any malicious attempt to compromise the security of the network
- **IDS for WSN** — WSNs have additional security issues when attacker intrusions are considered (e.g. malicious mote injection, mote-stealing, mote-tampering etc.). In order to detect and defend against classical and specific attacks, an IDS for WSN should be carefully designed while respecting the resource constraints of the WSN motes.
- **Solution** $\rightarrow$ Adopt WSN-specific lightweight solutions

WIDS

- Our approach, the *WSN Intrusion Detection System* (**WIDS**)[1], is an ad-hoc IDS solution for WSN. It exploits the *Weak-Process Models* (WPM) [1] to model WSN attacks, using only a small amount of computational resource and storage.
- A WPM can be represented as a graph in which nodes represent the **state** of the mote and the edges represent the possible **transitions**.
- Leaf nodes of the WPM represent a successfully detected attack.
- Each transition has a **score** value and a list of **observables** events.
- WIDS tracks observables, process transitions and update the list of possible states (**state trace**) in which the WSN could be. At each update interval, WIDS tries to prune all the states from the trace in which the WSN mote is unlikely to be. The remaining states are considered and, if any of them is a state marked as *dangerous*, a notification of a successful detection is sent to the application.

WIDS: Concept

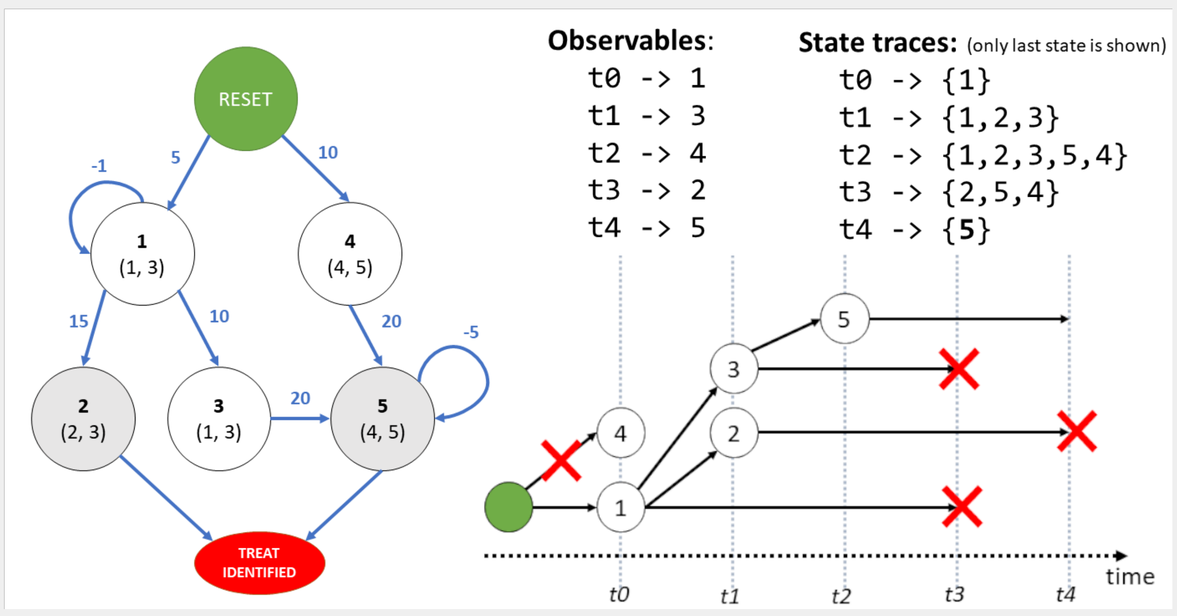

Figure 1:A sample WPM

The WPM represent an example attack model with seven states (including the starting state *Reset* and the *Detected* state). Here, the Observable creation for each step is shows along the resulting state traces and the time diagram that results in WIDS selecting the state trace ending with state 5 as dangerous.

Acknowledgments

- This research has been founded in part by the **SafeCOP project**, under the ECSEL Joint Undertaking Grant #692529

TinyWIDS: TinyOS-based implementation of WIDS

- **TinyWIDS** [3] is our current implementation of WIDS using the TinyOS [2] operating system for WSN. The architecture of TinyWIDS is shown if Figure 2:
- **Metrics**. Each Metric components keep track of a particular event or value from the hardware or from the driver layer (provided by TinyOS). An example is the `FrameReceived` metric.
- **Observables**. An Observable is a event that triggers when a chosen set of Metrics fulfil a selected *condition* e.g. `FrameReceived` Metric in a 1-second timeframe has a value above 100.
- **Observable Notifier**. The connection between Observables and their required subset of Metrics is performed by the Observable Notifier component (OBSN). The OBSN monitors the Metrics, check for the Observable triggering conditions and *"creates"* the Observable.
- **Attack Models**. Each class of attacks is modeled by a WPM. The WPM are codified in JSON format and can be trans-compiled into C headers files by the *Model Compiler*.
- **WIDS**. The WIDS component continuously collect newly created Observables into a list. The list is sampled with a selectable frequency and passed to WPMs for updating the current states. When one or more attacks are detected, WIDS notify the upper layer (e.g. the application or the component responsible of *reactions* to attacks).
- **Attacks**. For each modeled attack, a proper Attack component is provided. The Attack components receive the Observable list and update their Attack Models, eventually notifying the WIDS component upon the detection of the attack.

TinyWIDS: Software Architecture

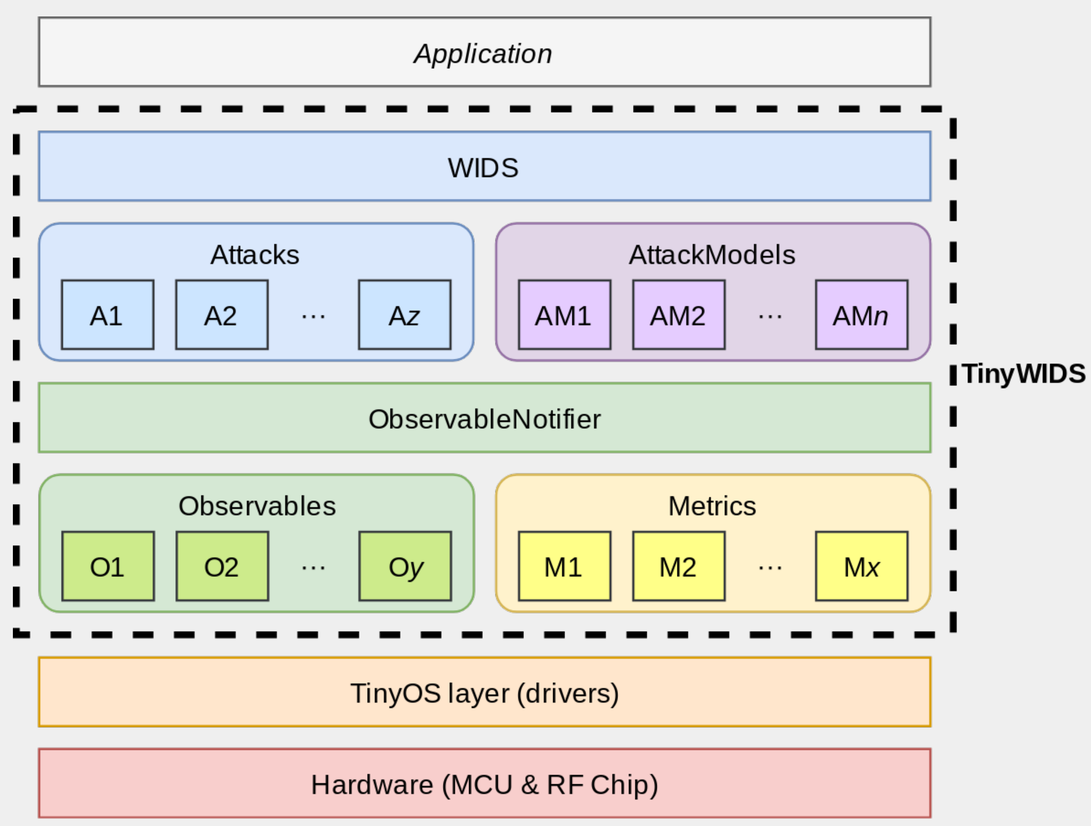

Figure 2:TinyWIDS Architecture

References

[1] S. Marchesani, L. Pomante, M. Pugliese, F. Santucci. "WINSOME: A Middleware Platform for the Provision of Secure Monitoring Services over Wireless Sensor Networks". 9th International Wireless Communications & Mobile Computing Conference (IWCMC 2013), Cagliari, July 2013.

[2] TinyOS homepage: http://webs.cs.berkeley.edu/tos/

[3] L. Bozzi, L. Di Giuseppe, L. Pomante, M. Pugliese, M. Santic, F. Santucci, W. Tiberti. TinyWIDS: a WPM-based Intrusion Detection System for TinyOS2.x/802.15.4 Wireless Sensor Networks. Fifth Workshop on Cryptography and Security in Computing Systems (CS2 2018).

Contact Information

Authors email addresses:
- walter.tiberti@graduate.univaq.it
- luigi.pomante.univaq.it