Simulaciones Software para el Estudio de Amenazas contra Sistemas SCADA

Joaquin Garcia-Alfaro
Telecom SudParis
CNRS Samovar
UMR 5157, Evry, France
Email: joaquin.garcia-alfaro@acm.org

Cristina Romero-Tris
Universitat Rovira i Virgili
Avd Països Catalans, 26
43007 Tarragona
Email: cristina.romero@urv.cat

Jose Rubio-Hernan
Telecom SudParis
CNRS Samovar
UMR 5157, Evry, France
Email: jose.rubio_hernan@telecom-sudparis.eu

***Resumen*—El objetivo de las tecnologías SCADA (acrónimo de *Supervisory Control And Data Acquisition*), es proporcionar control remoto para la supervisión de infraestructuras críticas. Ataques contra tales sistemas suponen un riesgo importante. Nuestro interés en la temática es poder investigar mejoras en la seguridad de los sistemas SCADA, usando abstracciones a nivel de software, herramientas de simulación, dispositivos físicos y trazas de datos a partir de sistemas reales. Este artículo presenta, de manera general, algunas construcciones básicas de lo que son las tecnologías SCADA y sus componentes. Introduce, también, características generales de algunos simuladores open source disponibles. Por último, detalla limitaciones y mejoras potenciales, orientadas a completar el estudio de técnicas de detección de anomalías a nivel de señales físicas entre los componentes de sistemas SCADA.**

***Palabras clave*—Seguridad TIC, Detección de Intrusiones, Sistemas Críticos, Simulación por Ordenador, Sistemas SCADA.**

I. Introducción

SCADA es el acrónimo de Supervisory Control And Data Acquisition (Supervisión, Control y Adquisición de Datos). Los sistemas SCADA son grandes infraestructuras utilizadas para recoger y almacenar datos a distancia y en tiempo real. Estos sistemas se usan normalmente en la industria y en arquitecturas críticas, controlando procesos químicos, físicos o de transporte. Algunos ejemplos de sistemas SCADA son el suministro de agua, la generación y distribución de energía eléctrica o de gas.

Debido a su naturaleza crítica, una vulnerabilidad en la seguridad de un sistema SCADA podría tener graves consecuencias si fuera detectada por un atacante. Por esta razón, es necesario analizar los posibles ataques y estudiar las contramedidas existentes de cualquier sistema SCADA. El problema es que estos análisis no pueden realizarse sobre sistemas reales ya que el coste de reproducir los componentes SCADA es demasiado elevado, y no se puede asumir el riesgo de realizar experimentos en sistemas reales en funcionamiento. Por consiguiente, es necesario utilizar modelos teóricos y herramientas que permitan simular sistemas SCADA, posibles ataques, y contramedidas. Nuestra propuesta pasa por proponer una virtualización de sistemas SCADA para poder reproducir ataques cibernético en un entorno académico.

En este artículo, revisamos elementos tradicionales de una arquitectura SCADA, proponemos una arquitectura de estudio concreta y revisamos una solución existente para poder simular por ordenador, los distintos elementos de la arquitectura de estudio propuesta. A continuación, proponemos una extensión para poder investigar técnicas de detección de anomalías entre los componentes de las capas inferiores de la arquitectura de estudio. A ese nivel, la mayor parte de técnicas de detección requieren un tratamiento a nivel de las señales intercambiadas por dispositivos tales como sensores y actuadores. La mayor parte de las funciones en la solución de simulación estudiada se limitan a simular ataques contra dispositivos de capas superiores, tales como terminales remotos e interfaces intermedias. Nuestra extensión permite poder integrar funcionalidad adicional, a partir de otras plataformas de simulación, mediante el uso de librerías dinámicas compartidas. Como resultado final, esperamos poder poner en práctica técnicas de co-simulación, sin importar la naturaleza de los dispositivos evaluados (tanto reales como virtuales).

Organización del artículo: Las Secciones II y III definen conceptos básicos asociados con tecnologías SCADA de uso general. Las Secciones IV y V presentan una arquitectura SCADA de ejemplo, y nuestra metodología propuesta para evaluar amenazas y contramedidas. La Sección VI finaliza con las conclusiones del artículo.

II. Arquitectura de un Sistema SCADA

Presentamos en esta sección los elementos de una arquitectura SCADA típica. La bibliografía utilizada se basa en [1], [2], [3]. Asumimos que una arquitectura SCADA se compone principalmente de los siguientes elementos (representados, a modo de ejemplo, en la figura 1:

- Interfaces de usuario-máquina (en inglés, Human Machine Interfaces -HMIs-)
- Unidades de estación maestra (en inglés, Master Terminal Units -MTUs-)
- Unidades de estación remota (en inglés, Remote Terminal Units -RTUs-)

ISBN: 978-84-9717-323-0

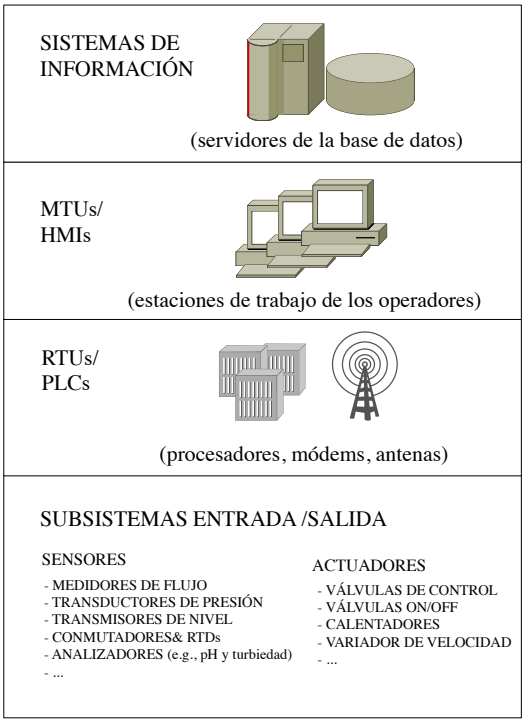

Figura 1: Elementos de un scenario SCADA de ejemplo

- Controladores lógicos programables (en inglés, Programmable Logic Controllers -PLCs-)
- Sensores y actuadores

II-A. MTUs y HMIs

Los MTUs de un sistema SCADA se localizan en el centro de control de la organización. Sirven para dar acceso a la gestión de las comunicaciones, la recogida de datos (generada por los RTUs), el almacenamiento de datos, y el control de sensores y actuadores conectados a los RTUs. La interfaz para los administradores del sistema es proporcionada por los HMIs.

II-B. RTUs

Los RTUs son unidades aisladas de adquisición de datos y control. Normalmente se trata de dispositivos basados en un microprocesador que controla y supervisa los componentes industriales de manera remota. Tienen dos tipos de funciones: (1) controlar y recoger datos de los equipos de proceso remotos, y (2) enviar los datos recogidos a una estación maestra de supervisión. Los RTUs modernos también pueden comunicarse entre ellos (ya sea con cable o de forma inalámbrica)

II-C. PLCs

Los PLCs son pequeñas máquinas de cómputo con un microprocesador. Las principales diferencias con respecto a los RTUs son el tamaño y la capacidad. Un RTU tiene un mayor número de entradas y salidas que un PLC, y mayor poder de proceso (e.g., para post-procesar los datos recogidos antes de generar las alertas para el MTU a través del HMI). Por su parte, los PLCs son frecuentemente representados como sensores con capacidad de comunicación. Los PLCs tienen dos ventajas principales respecto a los RTUs comercializados: (1) son dispositivos de uso general, permitiendo una gran variedad de funciones, y (2) son físicamente compactos, i.e., requieren menos espacio que otras alternativas.

II-D. Sensores y Actuadores

Los sensores son dispositivos de captación de medidas relacionadas con fenómenos físicos, respondiendo a algún estímulo físico. Este estímulo se transforma en una señal eléctrica, que a su vez se transforma y se almacena como datos. Los sensores pueden considerarse como el punto de entrada de un sistema SCADA. Sus datos se envían a capas superiores a través de RTUs y/o PLCs. Los actuadores son dispositivos de control, encargados de gestionar dispositivos externos. Los actuadores pueden considerarse el punto de salida de un sistema SCADA, recibiendo órdenes de RTUs y/o PLCs.

III. Protocolos de Comunicación SCADA

Los sistemas SCADA pueden usar una gran variedad de protocolos y de patrones de comunicación. A continuación se muestra un resumen de protocolos de ejemplo, respecto a los elementos descritos anteriormente.

III-A. HMI/MTUs ↔ RTUs/PLCs

La comunicación entre el centro de control (compuesto por servidores MTU/HMI) y los dispositivos remotos puede ser o no guiada. Las comunicaciones guiadas se realizan por canales eléctricos, redes de teléfono públicas (e.g., un módem de acceso telefónico o una línea alquilada) y las WANs de la organización. Las comunicaciones no guiadas se realizan por canales radio, satélite y redes inalámbricas (e.g., WPAN, WLAN, WMAM, and WWAN).

Se considera que los protocolos que se usan entre el centro de control y los dispositvos remotos son protocolos tradicionales (e.g., protocolos basados en TCP/IP), a través de redes estándares cableadas o inalámbricas (e.g., GPRS, UMTs, LTE) También es posible el uso de VPNs y de circuitos dedicados. En el centro de control, se pueden utilizar también protocolos dedicados de tipo OPC (siglas de Object Linking and Embedding (OLE) for Process Control).

III-B. RTU/PLC ↔ Sensores/Actuadores

La comunicación puede ser guiada (por cable) o no guiada (e.g., inalámbrica, basada en tecnologías como por ejemplo wifi, bluetooth, y zigbee). Protocolos de ejemplo a este nivel son Modbus (e.g., Modbus RTU y Modbus ASCII), PROFINET, DNP3 (DNP3/AGA 1.2 cifrado), EtherCAt, Fieldbus, protocolos OPC (OPC AppID y OPC UA). Remitimos al lector a consultar [1], [2], [3] para más información sobre dichos protocolos.

IV. Escenario SCADA de Ejemplo

Consideramos la arquitectura SCADA siguiente: distribución de energía, distribución de agua, y tratamiento de residuos. La figura 2 muestra una abstracción del sistema considerado.

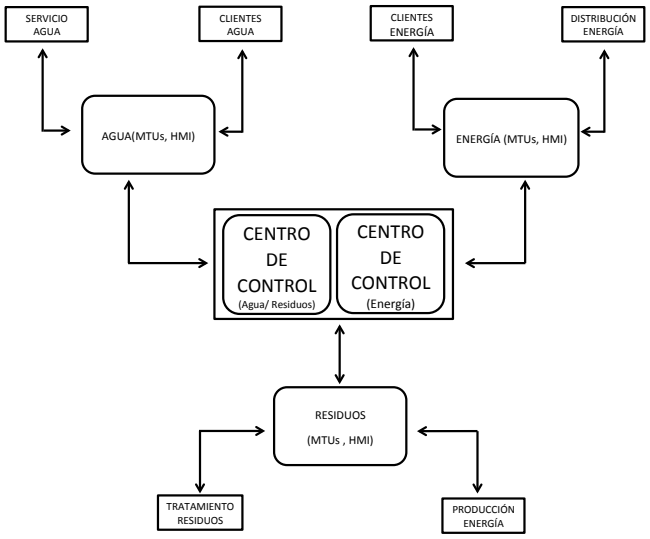

Figura 2: Escenario de ejemplo

IV-0a. Distribución de Energía: El sistema está dividido en tres capas: (a) alta tensión, (b) media tensión, y (c) baja tensión. El centro de control supervisa y gestiona tan sólo las capas (a) y (b). La capa (c) es gestionada a partir de sistemas tradicionales de tipo *hot line*.

La infraestructura asociada al centro de control cuenta con un sistema WAM (siglas en inglés de wide area measurement) que permite recoger, intercambiar y procesar los datos. El sistema WAM se complementa con una infraestructura basada en GPS para localizar los puntos finales (extremos) del sistema y un sistema operacional para gestionar los cortes de energía y la resolución de incidentes en los puntos finales.

A parte de bases de datos, asumimos aquí también elementos tipo MTUs centrales, para la gestión de áreas de alta y media tensión; puntos de suministro (e.g., energía hidráulica, solar, eólica); sub-estaciones; y puntos de transformación (eg., alta-a-media y media-a-baja) en RTUS, sensores de voltaje y corriente, y actuadores.

IV-0b. Distribución de Agua y Tratamiento de Residuos: El sistema está compuesto por un MTU central y varias sub-estaciones MTU. Cada sub-estación gestiona varios RTUs directamente conectados a sensores y actuadores. Los componentes remotes se dividen en dos capas: procesamiento (agua o residuos) y producción de energía. Los sensores (puntos de entrada del sistema) están conectados a la primera capa, proporcionando medidas de presión, temperatura, flujo y posición. Los actuadores, de carácter servomotor, están conectados al sistema de distribución de energía.

IV-0c. Protocolos de Comunicación: La comunicación (por cable o inalámbrica) desde los MTUs hasta los centros de control, así como desde los RTUs a los MTUs, utiliza VPNs a través de redes públicas y privadas (i.e., redes conmutadas públicas o líneas alquiladas para propósitos de supervisión). Asumimos que los protocolos se basan en TCP/IP. También se asume que la comunicación entre RTUs, PLCs, sensores y actuadores se realiza a través de enlaces inalámbricos o físicos. Los protocolos usados se basan, en general, en Modbus (por ejemplo, Modbus RTU o ASCII por comunicación serie o sobre TCP/IP) y DNP3 (por ejemplo, DNP3 AGA 1.2 cifrado). Remitimos al lector a consultar [1], [2], [3] para más información sobre dichos protocolos.

IV-A. Posibles Ataques a la Seguridad del Sistema

Suponemos que los objetivos del atacante son poner en riesgo la integridad y la disponibilidad del sistema dscrito. Los ataques más simples pueden ser basados en *eavesdropping*, *replay* (o ataque de reinyección), e *impersonation* (o ataque de suplantación de identidad). Ataques más complejos pueden ser iniciados como spam, phishing, e inyección de datos en puertos tipo USB. No consideramos ataques a gran escala (e.g., ataques similares a stuxnet, bien preparados, y con el apoyo técnico y financiero de grandes organizaciones). Como acciones del adversario, asumimos intercepción y modificación de paquetes, control de tráfico, inyección de comandos falsos, etc. Algunos informaciones adicionales son listadas a continuación.

- Posibles ataques a HMI/MTUs: problemas de seguridad de las tecnologías de la información y la comunicación tradicionales.
- Posibles ataques a PLCs: ataques lógicos (e.g., reescribir áreas de la memoria del PLC) y ataques físicos (e.g., apagar dispositivos de entrada/salida de manera remota).
- Posibles ataques a los puntos finales: amenazas a las comunicaciones inalámbicas, incluyendo servidores de agujero negro (blackholes), gusanos (wormholes), clonación (cloning) y suplantación de identidad (impersonation).
- Puntos de entrada: infiltración no detectada a través de dispositivos infectados (e.g., memorias USB), equipos corporativos usados de forma equivocada, protocolos vulnerables en las capas más bajas, etc.
- Problemas de seguridad en protocolos Modbus y DNP3: spoofing de mensajes en modo broadcast, ataques de replay en las respuestas a la base, control directo de esclavos, escáner de red, reconocimiento pasivo, retardo de la respuesta e intrusión.

Por último, consideramos como contramedida principal la reconfiguración del sistema (e.g., deshabilitar servicios, cortar conexiones, redirigir conexiones, bloquear aplicaciones, bajar la prioridad a los mensajes, etc.).

V. Simulación del Escenario Mediante SCADASim

Nuestra propuesta se basa en la simulación del escenario presentado en la sección IV mediante SCADASim [8], una librería para la co-simulación de entornos SCADA. A su vez,

SCADASim se basa en la plataforma de creación de simulaciones OMNeT++ [6] (disponible en http://www.omnetpp.org/). A continuación, presentamos de manera general OMNeT++ y SCADASim.

OMNeT++ es una plataforma *open source* para la creación de simuladores de eventos discretos. OMNeT++ es ampliamente utilizado a nivel académico para la simulación de redes y nuevos protocolos. La simulaciones OMNeT++ se desarrollan principalmente a partir de dos tipos de lenguaje. En primer lugar, la lógica de la simulación se desarrolla en lenguaje C++. En segundo lugar, la descripción de topologías mediante un lenguaje propio de OMNeT++ denominado NED (NEtwork Description). NED es utilizado para ensamblar componentes individuales en nuevos componentes y modelos.

Adicionalmente, OMNeT++ dispone de un gran número de librerias externas para modelar un gran número de tecnologías y protocolos, así como la creación de componentes basados en multiprocesadores y sistemas paralelos o distribuidos. Véase, por ejemplo, las librerías INET (http://inet.omnetpp.org/) para la simulación de protocolos basados en UDP, TCP, SCTP, IP, IPv6, Ethernet, PPP, 802.11, MPLS, OSPF, y muchos otros; VEINS (http://veins.car2x.org/) para la simulación de redes vehiculares; CASTALIA (http://castalia.research.nicta.com.au), para la simulación de redes inalambricas de sensores; etc. OMNeT++ permite cubrir el vacío entre herramientas orientadas a la investigación académica (tipo NS-2 y NS-3) con la potencia y facilidad de uso de herramientas comerciales de alto coste como OPNeT de Riverbed Technology (http://www.opnet.com/).

SCADASim es un proyecto *open source* disponible en http://github.com/caxqueiroz/scadasim. SCADASim permite co-simulación. Es decir, permite la coexistencia entre dispositivos simulados y dispositivos reales. SCADASim también ofrece la posibilidad de simular la ejecución de ataques contra los dispositivos de la simulación (virtuales o reales). Para ello, SCADASim tiene implementado un módulo llamado SSProxy, que actúa de enlance entre los componentes reales y los simulados. El objetivo de este módulo es recibir peticiones de una IP externa y transmitirlas a los componentes internos de la simulación. Del mismo modo, este módulo también enviaría mensajes al exterior generados por componentes internos. Para más información sobre SCADASim, remitimos al lector a las siguientes publicaciones indicadas en [8], [9].

La principal limitación actual de SCADASim es el tratamiento de la capa física de un sistema SCADA simulado por software. Por ejemplo, la funcionalidad existente para la incorporación de procesadores digitales para el tratamiento de señales es muy limitado. Esta limitación supone que el estudio de amenazas hacia las capas más bajas de la arquitectura SCADA (en especial en lo que respecta a las comunicaciones entre sensores, actuadores y PLCs), es extremadamente limitada. De hecho, esta limitación dificulta la incorporación de técnicas relevantes basada en detección de anomalías a nivel de tratamiento de señales. A modo de ejemplo, resumimos en la siguiente sección dos trabajos dentro de dicha categoria, que consideramos relevantes para el estudio de seguridad propuesto en la sección IV.

V-A. Detectores de Mo et al.

Para la detección de determinados ataques contra sistemas SCADA a nivel de PLCs, sensores y actuadores, se puede añadir a las medidas clásicas de control del flujo de datos, otro control que consiste en la autenticación de la señal que llega al sistema. Para el control y la verificación de la señal que llega al sistema, además del control y detección mediante estimación de fallos utilizado en los sistemas clásicos que permite enmarcar la señal de llegada dentro de unos patrones y verificar que la señal se sitúa dentro de los margenes (método que se puede utilizar para detectar, por ejemplo, un ataque de denegación de servicio), existen técnicas basadas en la incorporación de ruido aleatorio. Dicho ruido es introducido en la señal, para su posterior autenticación. Esta incorporación permitirá más adelante verificar que son señales válidas. En la figura 3 mostramos un detector de ejemplo propuesto por Mo *et al.* en [13], [14]. Este detector permite tratar ataques de *replay* que pueden ser transparentes a otras contramedidas. Su principal limitación es que disminuye el rendimiento del sistema, ya que debe analizar y autenticar cada señal.

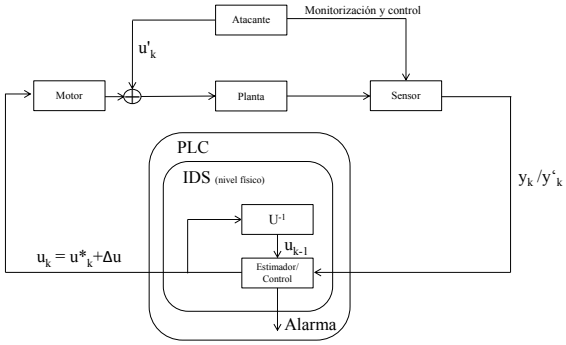

Figura 3: Sistema de detección

En la figura 3, y_k es la respuesta real del sensor, e y'_k es la respuesta forzada por el atacante al realizar un ataque de *replay*. u_k es la salida del PLC hacia el motor, y u'_k es la señal que el atacante envía la PLC como respuesta del motor. La salida del PLC puede representarse como $u_k = u^*_k + \Delta u_k$ siendo u^*_k la respuesta de control del IDS (sistema de deteción de intrusos) de nivel físico del PLC, y Δu_k el ruido aleatorio añadido en la mejora.

Una mejora en el uso de este sistema la podemos ver en [15], el sistema esta representado en la figura 4, donde se utilizan una serie de *juegos estocásticos* que permiten crear una politica de combinación entre un coste óptimo pero un sistema inseguro y un coste alto y un sistema seguro, permitiendo así mejorar el rendimiento del sistema.

V-B. Integración de los Detectores de Mo et al. en el Escenario de Ejemplo mediante SCADASim y MATLAB

En este apartado, mostramos una solución (en curso) para tratar las limitaciones de SCADASim reportadas en los apartados anteriores de esta sección. Nuestra propuesta se

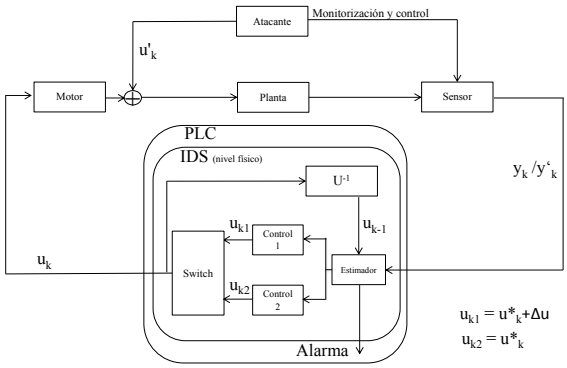

Figura 4: Detector de Mo *et al.* optimizado

basa en la incorporación de nueva funcionalidad para el procesamiento de señales digitales mediante la incorporación de librerías dinámicas mediante compilación compartida de código OMNeT++ y código MATLAB [12]. La figura 5 representa la arquitectura de nuestra propuesta. Los módulos OMNeT++ están representados en forma de cajas en color blanco. Los modulos SCADASim en cajas de color gris claro. Los módulos MATLAB en cajas de color gris oscuro. Nótese que las aplicaciones y las capas de enlace y transporte se han implementado como componentes SCADASim y OMNeT++. Las capas físicas se han implementado como librerías dinámicas MATLAB. Estás librerías son llamadas en tiempo de ejecución por las distintas instancias de OMNeT++. La capa de aplicación simplemente crea y recibe mensajes. La capa de transporte y la capa de enlace se limitan a tratar y verificar los mensajes. Por último, las funciones MATLAB son utilizadas para modular y demodular los mensajes mediante PSK (desplazamiento de fase, del inglés *Phase Shift Keying*), así como para implementar las propuestas de detección de anomalías especificadas en los trabajos de Mo et al. [13], [14].

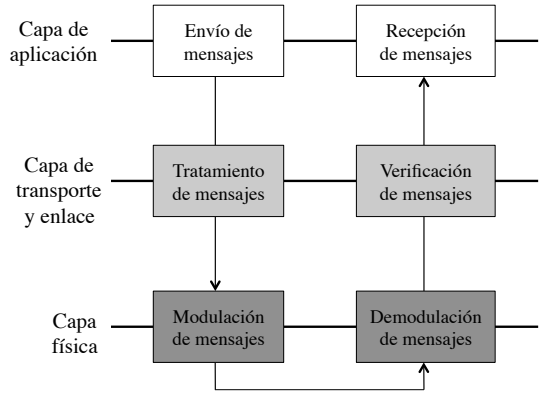

Figura 5: Arquitectura de nuestra propuesta de virtualización. Los módulos OMNeT++ están representados en forma de cajas en color blanco. Los modulos SCADASim en cajas de color gris claro. Los módulos MATLAB en cajas de color gris oscuro.

VI. Conclusiones

El objetivo de las tecnologías SCADA (Supervisory Control and Data Acquisition) es proporcionar control remoto que permita supervisar y monitorizar infraestructuras críticas e industriales, como la distribución de energía y agua. Ataques a este tipo de sistema tendrían consecuencias muy graves, y por ello es necesario herramientas que permitan detectarlos y analizar las contramedidas necesarias. En este artículo, hemos revisado una propuesta para simular ataques realizados contra sistemas SCADA, llamada SCADASim [8], [9]. A partir de la descripción de una infraestructura SCADA de ejemplo para controlar suministro de energía, hemos propuesto posibles ataques contra su seguridad, y simulados mediante SCADASim. Hemos identificado también algunas limitaciones en la versión actual de SCADASim, que impide la simulación completa de protocolos previstos en la arquitectura de ejemplo, así como la incorporación de técnicas de detección de anomalías mediante tratamiento de señales. Por ello, hemos reportado una extensión en curso sobre SCADASim, mediante la incorporación de compilación de librerías dinámicas compartidas con otras plataformas de simulación más adecuadas para el tratamiento de señales, como es el caso de MATLAB [12]. Nuestro trabajo futuro supone completar la extensión reportada en este artículo. En paralelo, proponemos trabajar otras extensiones sobre SCADASim que faciliten mayor facilidad para realizar estudios de co-simulación con componentes reales.

Referencias

[1] S. Sosik, “SCADA Systems in Wastewater Treatment,” *Process-Logic*, Whitepaper, 2007.

[2] K. Stouffer, J. Falco, K. Kent, “Guide to Industrial Control Systems (ICS) Security Recommendations of the National Institute of Standards and Technology,” *NIST Special Publication*, vol. 800, 2008.

[3] M. Shahraeini, M. H. Javidi, “SCADA Systems in Wastewater Treatment,” *Wide Area Measurement Systems. Advanced Topics in Measurements*, Chapter 15, pages 303-322, 2012.

[4] P. Manadhata, “An Attack Surface Metric,” *PhD thesis, School of Computer Science, Carnegie Mellon University*, 2008.

[5] G. Gonzalez-Ganadillo, “Optimization of Cost-based Threat Response for Security Information and Event Management Systems,” *PhD thesis, Paris VI University*, 2013.

[6] OMNeT++ Network Simulation Framework. Available On-Line. http://www.omnetpp.org/

[7] The NS-3 discrete-event network simulator. Available On-Line. http://www.nsnam.org/

[8] C. Queiroz, A. Mahmood, Z. Tari, “SCADASim – A Framework for Building SCADA Simulations,” *IEEE TRANSACTIONS ON SMART GRID*, 2(4):589–597, 2011.

[9] C. Queiroz, “A Holistic Approach for Measuring the Survivability of SCADA Systems,” PhD Thesis, College of Science, Engineering and Health, RMIT University, August 2012.

[10] M. Roesch, “Snort: Lightweight Intrusion Detection for Networks,”*13th USENIX Conference on Large Installation Systems Administration Conference (LISA-99)*, pages 229–238, 1999.

[11] Quickdraw SCADA IDS Signatures. Available On-Line. http://www.digitalbond.com/tools/quickdraw

[12] Guide, “MATLAB User’s”. The mathworks. Inc., Natick, MA. (1998).

[13] Mo, Yilin, and Bruno Sinopoli. “Secure control against replay attacks.” 47th Annual Allerton Conference on Communication, Control, and Computing, pp. 911–918, 2009.

[14] Mo, Y., Kim, T. H., Brancik, K., Dickinson, D., Lee, H., Perrig, A., and Sinopoli, B. “Cyber-physical security of a smart grid infrastructure.” Proceedings of the IEEE, 100(1), pp. 195–209, 2012.

[15] F. Miao, M. Pajic and G. J. Pappas. “Stochastic Game Approach for Replay Attack Detection.”, University of Pennsylvania, Philadelphia, PA, USA. 2013.