Discovery, Infiltration, and Denial of Service in a Process Control System Wireless Network

Bradley Reaves and Thomas Morris
Department of Electrical and
Computer Engineering
Mississippi State University
Starkville, MS 39762
Email: {bgr39,morris}@ece.msstate.edu

***Abstract*—Process control systems and Supervisory Control and Data Acquisition (SCADA) systems use computers to control physical processes in many critical industries including electric power generation, electric power distribution, gas pipelines, waste treatment, water distribution, and many others. Because process control systems can be spread over large distances, wired connections become infeasible. Commercially available wireless radios are often used in place of wires to connect network nodes. In this paper we provide an overview of process control systems, discuss how process control systems differ from networks commonly found in the information technology domain, demonstrate the ability to detect and infiltrate a wireless radio network used in control systems, and finally, detail a denial of service attack against a process control system. This attack denies feedback from nodes monitoring the controlled physical process and is therefore a dangerous cyber-attack.**

I. Introduction

Process control systems, also known as Supervisory Control and Data Acquisition (SCADA) systems, often use proprietary long-distance radio networks to interconnect network nodes in point-to-multipoint configurations on multi-acre industrial sites and to interconnect network nodes via multi-mile (tens of miles) point-to-point line-of-site connections. In this paper, we demonstrate a scan which discovers the existence of radio networks built with a commercially-popular proprietary radio technology. Next, we demonstrate a second scan which learns the configuration setting required to infiltrate discovered radio networks. Finally, we demonstrate a denial of service attack which breaks critical feedback control loops used by process control systems to ensure safe and proper physical process functionality.

Process control systems are pervasive computer systems which monitor and control physical processes in both utilities and private industry. Water treatment plants, electricity generation and distribution (including smart grid initiatives), petroleum refining, chemical processing, and manufacturing are examples of industrial applications which use process control systems to control critical cyber-physical processes. In recent years, the security of process control systems has come under much scrutiny. Because of the consequences of disruption or damage to process control systems, they form a substantial target for individuals or groups that would like to cause large scale and/or conspicuous damage. There have been a number of security incidents involving process control systems.

One of the most cited examples of a process control systems system security breach occurred in Maroochy, Queensland, Australia in 2000 [1]. A disgruntled contractor obtained a laptop, control system computer, and wireless radio of the same model used in a local wastewater station and was able to inject erroneous commands and data into the process control system; as a result, one million liters of wastewater was dumped into a storm water drain, which drained into nearby waterways. In this breach, the contractor was able to use insider knowledge to access the systems remotely and cause havoc on a number of occasions. The contractor used a stolen radio, but the same model (a Motorola Radius M120 [3]) used in the breach was available on eBay at the time of this writing.

In 1991 a Bellingham, Washington process control system became unresponsive and failed to open a pressure relief valve [2]. Due to this failure, the pipeline ruptured and leaked over 250,000 gallons of gasoline into two nearby streams. The leaked gasoline later ignited and created an explosion which killed three persons and injured eight others. The exact cause of this incident is unknown because of a lack of available forensic evidence detailing system activity around the time of the accident. However, in an analysis of the incident, Abrams and Weiss show that the human machine interface software used by operators to monitor and control the pipeline became unresponsive around the time of the pipeline rupture and subsequent explosion. According to Abrams and Weiss the system normally updated screen information, including pipeline pressure, every five to seven seconds, however the system did not update for over 20 minutes around the time of the incident. This left operators unaware of the exact condition of the process control system and unable to intercede to prevent the ultimate failure. In this paper, we demonstrate a denial of service attack which would create symptoms very similar to those seen in the Bellingham incident.

A number of vulnerabilities may be present in a process control system. Process control systems integrate conventional

978-1-4244-4626-1/09/$25.00 ©2009 IEEE

computing platforms running Microsoft Windows, UNIX, or Linux operating systems with application-specific computing platforms running real-time operating systems and or no operating system at all. Because of this combination of conventional and specialized computer systems, process control systems exhibit vulnerabilities commonly found in the information technology domain as well as vulnerabilities specific to their unique configurations and purpose. While process control systems can "inherit" the vulnerabilities of the conventional technologies they use, there are also domain-specific vulnerabilities. MODBUS and DNP3 are two communication protocols common in the process control system domain. Neither Modbus [7] nor DNP3 included security provisions in their respective specifications, although security has been recently added to DNP3 in IEC 62351-5. In particular, neither of the protocols specified any form of authentication or cryptography. Another area of vulnerability is in the Human-Machine Interface (HMI) application software [4] used to allow an operator to monitor and control process control systems. Furthermore, the physical network layers of process control systems, often RS-232, proprietary and standards-based wireless systems, dial-up modems, satellite, and microwave are vulnerable to intrusion and attack. This paper demonstrates vulnerabilities in a proprietary wireless system commonly used in the process control system domain.

In this paper, we investigate a model of wireless radio from a single manufacturer. The investigated radio is marketed primarily for process control systems communication. We choose not to disclose the name of the radio manufacturer or the specific model of radios tested to avoid enabling attacks. Rather, through this paper we hope to highlight the need for further research and investigation into vulnerabilties specific to these types of radios and research towards solutions which will secure this critical infrastructure. The test radio is a prominent, recently developed product of a leading process control systems/SCADA communication system provider that is much newer than the Motorola M120 that was exploited in the aformentioned Maroochy incident. The manufacturer of the tested radio promises reliable and secure encrypted communication via a "proprietary spread spectrum technology [that] prevents detection and unauthorized access." In addition to the publication of the paper, we are communicating details of this vulnerability to the radio manufacturer.

We are able to discover active radio networks, connect to these network without authorization, eavesdrop on network traffic, and inject data into the network. Furthermore, we are also able to disrupt and deny communications in the radio network. Specifically, we are able to prevent slave nodes from communicating with master nodes in the process control system. Denying this communication breaks the feedback control loop used to control the physical process and makes it impossible to monitor and control the physical process.

The body of this paper begins by discussing related topics and background on the the specific radio system we used. Next, we descibe how to discover and connect to the radio network. Then we describe how to deny service to the radio network, and the potential consequences of such an attack outside of a lab. Finally, we present a list of potential countermeasures, future work, and conclusions.

II. Background

A. Scanning

A common technique in security auditing is scanning. Scanning can be defined as "a method for discovering exploitable communication channels." [5] Examples of scanning include wardialing, wardriving, and TCP/UDP port scanning. Wardialing is the practice of using a dial-up modem to dial a number of telephone numbers to find other computer systems, while wardriving is the practice of driving or otherwise travelling through an area looking for open WiFi access points. TCP/UDP port scanning makes use of one or more protocol characteristics to connect (or partially connect) to individual ports on a given network machine to determine what services are running on a machine.

In section III-A we discuss a method for discovering process control system proprietary radio networks which is similar to the scanning techniques mentioned above. Additionally, in section III-B we discuss a scanning method used to learn a set of connection paramters to join a process control system radio network.

B. Denial of Service

A Denial of Service attack is an attack that causes a computer system to be unavailable or unresponsive to requests from other systems. Examples from the internet include ping flooding, SYN flooding, and distributed denial of service attacks all of which attempt to overwhelm a server with illegitimate requests. These requests "tie up" the server's processing availability and no legitimate requests for service get through.

In section III-C we discuss a denial of service attack which prevents slaves in a process control system proprietary radio network from communicating with master nodes in the same network. This attack is similar to the denial of service attacks mentioned above.

C. Process Control Systems

Process control systems collect data from sensors that monitor physical processes and use this data to control the process via electronic control of actuators, switches, and valves. Figure 1 shows an example process control system. On the right of Figure 1 are two Remote Terminal Units (RTU). RTU provide an interface between Master Terminal Units (MTU) and gauges, meters, switches, actuators, drives, and other industrial components which interact directly with the physical process. Process control systems often use feedback control loops to control the physical process. A feedback control loop is a system in which process decisions are made by an operator (or computer) based on data sent from the downstream sensors. Figure 2 shows an example of a feedback control system. Sensor data is collected at the physical process and transmitted to the RTU and to upstream nodes, including the MTU, and

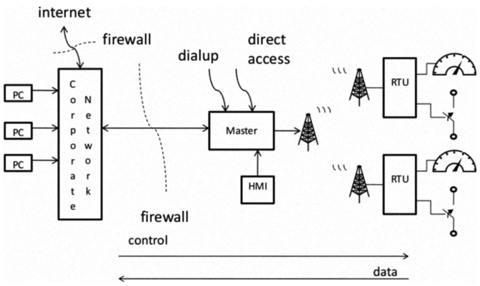

Fig. 1. Structure of an example process control systems

nodes in the corporate network. There are multiple feedback control loops in a typical process control system. First, a feedback control loop may exist at the RTU. The RTU can be programmed to react to physical events and make immediate control decisions. Second, sensor data is returned from the RTU to the MTU. The MTU can be programmed to react to sensor data and make control decisions. Third, the MTU returns sensor data to the human-machine interface (HMI). The human-machine interface presents sensor data to a human operator and allows the human to interact with the systems controlling physical process. Finally, sensor data can be sent to devices connected to the corporate network or even to devices connected via the internet. For example, in electric power distribution, electrical generation plants return sensor data to regulators which in turn send instructions to the power generator requesting specific levels of power generation. Each feedback control loop represents a vulnerable connection which must be maintained to ensure proper functionality of the process control system.

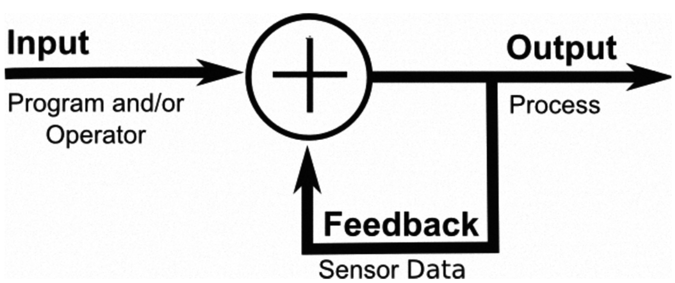

Fig. 2. Block Diagram of a Feedback Control System

In Figure 1 the MTU and RTU are connected via a wireless radio network. The connection between MTU and RTU may also be satellite, dial-up modem, microwave, or switched telephony. As mentioned above, process control systems are often connected to corporate networks (as seen on the left side of Figure 1). However, it is common for there to be no access control or firewall preventing users on the corporate network from interfacing with the process control system. Typically, only a small subset of employees at a company require access to the control system. Allowing other employees and systems on the network access makes the system vulnerable to inside attack and infection from viruses, spyware, and other malware.

Some process control systems are accessible via external connections including VPN and dial-up connections to allow control system engineers to access the process control system remotely.

While process control systems security and traditional IT security have much in common conceptually, in practice process control systems have a number of unique characteristics which add additional security challenges [6]. First, process control systems are designed to run for months or years without having to be taken offline; as such, when vulnerabilities are discovered, they often go unpatched. Second, the potential for unintended consequences related to a hardware or software change make process control systems administrators reluctant to patch or upgrade systems. Third, many process control systems depend on low-latency data from the processes they control; for this reason, many conventional security measures such as firewalls and antivirus programs cannot be used as they introduce too much latency for the data to be relevant to decision making. Finally, the feedback control loop itself is a unique characteristic. If the feedback control loop is broken the process control system can not be monitored or controlled.

D. Radio Background

Several examples of proprietary process control systems radio systems are given in Table I. All of the radios listed have similar feature sets with various levels of built-in security. Our analysis of datasheets and specifications available on the internet leads us to conclude that some or all of the radios listed in Table I are vulnerable to similar types of attacks. However, the specific details of the attack would need to be tailored to the target radio.

The radios tested for this work are used for RS-232 serial communications between two devices (presumably an MTU and RTU, but in general any RS-232 device). A pair of radios effectively replaces a null modem cable; each radio has one serial port which is connected to the end device, and the radios autonomously negotiate communications between themselves invisibly and independent of the end devices. It is difficult for an end device to distinguish being connected through a pair of wireless radios or a via an actual null modem cable, except perhaps through latency.

According to the product documentation, the tested radios use a frequency hopping spread spectrum Gaussian frequency shift keying transceiver in the 902 to 928 MHz unlicensed

TABLE I
SIMILAR RADIO PRODUCTS

Manufacturer	Model
Bentek Systems	SCADALink SMX-900
Synetcom	WISE DIN-S
GE MDS	Mercury 900
Freewave Technologies	Ranger R
Sutron	Wireless Radio Modem
Data-Linc Group	SM6000
Control Microsystems	SCADAWave Ultra KR50
Omega Communications	Modpac Plus RF Modem

industrial, scientific, and medical band. They use a proprietary, connection-oriented half-duplex slotted communication protocol that includes acknowledgment and error checking in the form of a 32-bit cyclic redundancy check (CRC).

The tested radios have 2 RS-232 serial ports. The first serial port is for communications and connects to the serial port of a PCS device. The second serial port is a diagnostics port that interfaces with vendor supplied software that gives an administrator information about the network. We make extensive use of the diagnostics port in the techniques described in this paper.

A radio network consists of a single "master" radio and one or more "slave" radios. There may also be repeaters in the network. There are two types of radio network: point-to-point and point-to-multipoint. In a point-to-point network, there is one master radio and one slave radio. In a point-to-multipoint network, there is one master radio and one or more slave radios (typically there will be more than one slave). The exploits and vulnerabilities discussed in this paper are targetted at the point-to-multpoint case. We have not yet discovered any vulnerabilities in point-to-point networks. In the remainder of this paper we will always refer to radios in a point-to-multipoint network, unless otherwise specified .

The tested radios are configurable via a built-in menu accessible from the serial port by pressing a button on the radio, or via the diagnostic port by sending a special character. The radios provide a number of parameters for configuring transmission and network characteristics. Table II lists the parameters relevent to the exploits described in this paper. A radio must be configured to be a master, slave, or repeater node of either a point-to-point or point-to-multipoint network. The RF data rate field sets the connection speed for radios in the network. The minimum and maximum packet size fields set a packet size range. This packet size governs the size of packets sent across the wireless connection and is not related to the size of packets in higher network layers.

Point-to-multipoint networks can use either serial number lists or a network identifier for access control. Each radio has a unique serial number which can not be changed. The serial number list is an access control list consisting of the serial numbers that a slave should connect to. The default setting for a point-to-multipoint network is to use a serial number list. The serial number is programmed or hard-wired into the radio at the factory making it difficult to spoof. For radios to connect either each slave must have the master's serial number in its serial number list or every radio must have the same network identifier. The network identifier is a 12-bit field. Setting the network identifier to a certain value configures the radio to use the serial number list for access control. The network identifier and call book features are more than a security feature, as they also allow colocation of mulitple radio networks. Radios with different network identifiers or which do not list the master in the slaves' serial number lists will never connect.

The radios use frequency hopping spread spectrum modulation; that is, the radios change transmission frequency pseudorandomly around 200 times per second. There are three parameters which control the frequency hopping schedule. The frequency key sets the order of frequency hops. All connected radios in a network must share a single frequency key. The hop table size allows users to choose the number of legal hop frequencies. The hop table version allows users to choose a subset of hop frequencies. Hop table version values vary by the country where the radio is installed.

Radios will join a network only if either the network identifier or serial number list match and all other parameters listed in Table II match the same parameter settings on the master.

TABLE II
FREQUENCY AND NETWORK CHARACTERISTICS

Characteristic	Description	Range
Network Type	Specify point-to-point or point-to-multipoint connection.	Point-to-Point Point-to-Multipoint
Endpoint Type	Specify master, slave, or repeater configuration.	Master, Slave, or Repeater
Call Book	List of authorized serial numbers for connecting radios.	7-digits
Network Identifier	Number used to specify to which network a radio should belong in a multipoint network.	0-4095
RF Data Rate	Controls data rate used between two radios	0-2
Frequency Key	Allows for selection among distinct hopping patterns	0-14
Maximum packet size	Sets the maximum packet size a radio can transmit	0-9
Minimum packet size	Sets the minimum packet size a radio can transmit	0-9
Hop Table Size	Allows for selection of how many channels to hop to in the spectrum	0-36
Hop Table Version	Allows for selecting which subset of frequencies to use	0-6

The tested radio does have access control and data confidentiality security features. Access control is achieved either with the network identifier parameter or the serial number list. We show in this paper that the shared network identifier field is inadequate for access control as it can easily be learned through an exhaustive search of the network identifier parameter range. Similarly, we show that a radio can search through possible master serial numbers to discover networks configured to use the serial number list. The tested radios also have a "Dynamic Key Substitution" encryption feature. This is a non-configurable encryption, and has no effect in preventing any two radios from communicating, given that parameters from Table II match on all radios in a network. The encryption feature appears to be intended only for communication channel confidentiality and/or to prevent reception of data by other devices in the ISM band.

We have developed a set of exploits which discover point-to-multipoint networks protected by serial number list and network identifier, learn the configuration settings required to connect to point-to-multipoint radio networks as a slave device, and deny communication from all slave nodes in point-to-multipoint networks.

III. Discovery, Infiltration, and Denial of Service

This section outlines the discovery, infiltration, and disruption of point-to-multipoint networks consisting of our tested radio. The full process for radio network discovery, infiltration, and all attacks are detailed in the following subsections.

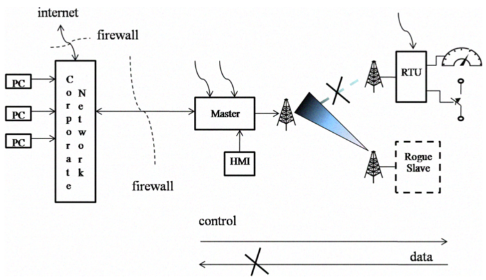

Fig. 3. Conceptual Diagram of the Attack

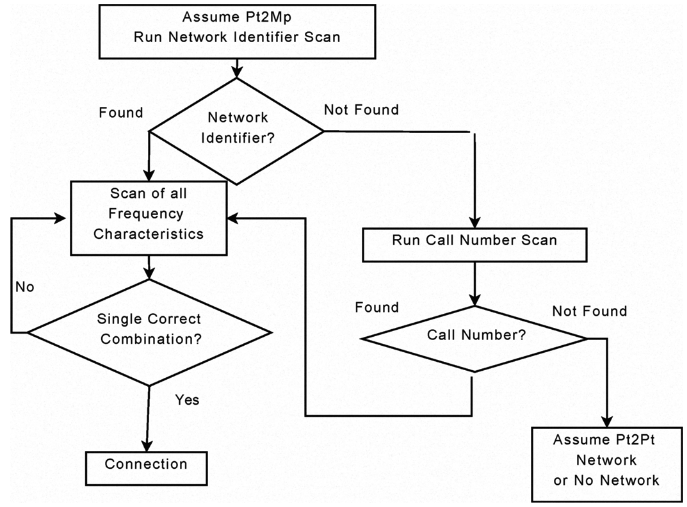

Fig. 4. The Attack Process

A. Point-to-Multipoint Wireless Network Detection

In this section we detail discovery of point-to-multipoint wireless networks consisting of the investigated radios. The flow chart in Figure 4 shows the steps required to discover and infiltrate a radio network. To discover a network, it is assumed that there is a nearby point-to-multpoint network. First, we search for point-to-multipoint networks configured to use a network identifier for access control. We perform an exhaustive search for the network identifier and RF data rate by attempting to connect to the radio network with each possible network identifier and RF data rate combination. Henceforth, we refer to this type of exhaustive search as a scan. Our experience has shown that running a the scan will always yield the correct result plus a minimal number of false positives. In our trials, all false positives are eliminated when we repeat the second scan (searching only the combinations returned as connect from the first pass) a second time. If a correct network identifier and RF data rate pair is found, a second type of scan is used to learn the remaining parameters from Table II. If a correct pair of network identifiers is not found, we then search for a point-to-multipoint network configured to use a serial number list. A scan of master serial numbers and RF data rate combinations is run; if a working master serial number/RF data rate pair is not discovered, it is assumed that there is either no network or a point-to-point network.

If a network identifier or serial number is found we have discovered an active radio network. Once a network is discovered we can proceed with a second scan which allows us to learn correct values for the remaining parameters from Table II which in turn allows us to infiltrate the network as a rogue slave device. After infiltrating a radio network we are able to execute several different attacks, each described in Section III-C . Merely discovering a process control system radio network, without learning the correct values for the remaining parameters from Table II, is a security vulnerability. Knowledge of a network's existence is a required precursor to attack. Our scan is analogous to wardriving - namely it can tell us if and how many networks exist within an area. Furthermore, we are able to inject small amounts (tens to hundreds of bytes) of bad data into a radio network without knowledge of the five remaining connection parameters from Table II.

We learned through laboratory examination of the tested radios that two radios with only matching network identifier, or correct serial number values, and matching RF data rate, will sporadically connect for brief periods regardless of the configured values for all other parameters in Table II. We exploited this characteristic to discover the existence of point-to-multipoint networks.

We believe the sporadic connections occur due to two radios, a master and a scanning slave, briefly sharing the same frequency for a short amount of time, usually less than one second, as they traverse through their respective hop sequences. During these brief windows of time the radios establish a connection and transmit and receive data. Much as network port scanners make use of partial TCP connections to determine that a service is listening on a given port, our scan uses the observed transitory incidental connections that occur to determine that a radio network is present in the area.

The scans were written using the Expect extension to the Tcl scripting language to configure the scanning radio via its diagnostic port. Expect is a tool for automating interactions with command-line and other text-based programs. Both scans work by configuring the radios via the serial port interface to a new network identifier or set of serial numbers, waiting for a time, and then examining internal registers in the radio which hold the number of disconnections and signal strength. This information is available to allow users to audit radio connection quality. This information can be used to learn of connections. If either the disconnection or signal values are non-zero then a connection has occurred. The connection may be the result of all parameters in Table II being correct, or it may be an example of the sporadic connections mentioned above. In either case, knowledge of the connection and subsequent disconnection indicates a correct network identifier and RF data rate pair or a correct serial number and RF data rate pair.

In our laboratory experiments, we were able to scan network identifier/RF data rate pairs at 6.375 seconds per combination. There are 12 288 possible network identifier/RF data rate combinations. At 6.375 seconds per combination, this yields a total search time of 21.76 hours maximum to discover a network and its associated data rate. We found experimentally by trial and-error that if we wait 5 seconds per combination there will always be at least one connection, and corresponding reported disconnection, for the correct network indentifier and RF data rate pair regardless of the value of the remaining parameters from Table II. The remaining time spent per combination is latency in the script in calculation and changing parameters.

For the serial number list case we were able to scan through serial number/RF data rate pairs at a rate of 1.7 seconds per combination. The serial numbers of the tested radios are seven digit decimal numbers, with three digits corresponding to the model of the radio. If there are at most 32 models of compatible radio (a very conservative estimate), there are a maximum of 96,000 serial number/RF data rate combinations. At 1.7 seconds per serial number, the total search time is less than 45.5 hours. Ten serial numbers can be tested at once since a call book may contain 10 serial numbers. As long as one of the serial numbers in the call book is correct, the radios will connect. We found experimentally by trial and error that if we wait 10 seconds per combination we will always have at least one connection, and corresponding reported disconnection, for the correct serial number and RF data rate pair regardless of the value of the remaining parameters from Table II. The remaining time spent per combination is latency in the script in calculation and changing parameters.

The serial number list is intended as an access control mechanism. However, we have found through laboratory experimentation with the tested radios that radios configured as masters ignore the call book entirely. This allows a rogue slave which has learned the master serial number to connect to a serial number list-protected radio network regardless of the contents of the master radio's serial number list.

B. Point-to-Multipoint Network Infiltration

Once an active point-to-multipoint radio network is discovered, it is possible to automate learning of the five remaining parameters from Table II which allow a rogue slave radio to connect to the discovered network. The five remaining parameters are frequency key, hop table size, hop table version, maximum packet size, and minimum packet size. We learn the correct value of the five remaining parameters by attemping to connect the to radio network with all combinations of the parameters. There are 539 400 legal combinations of these parameters. As with the network discovery scans described above, incorrect parameter combinations during an attempted connection may result in a sporadic connection or may result in no connection at all. Only the correct combination of all parameters from Table II will lead to a permanent network connection.

Once the correct parameter values are learned, a radio configured as a slave will connect automatically to the network. This "rogue" slave radio is then able to monitor all transmissions from the master radio to any slaves connected to the radio network. The rogue slave can also transmit information to the master. The rogue slave cannot communicate with other slaves in the same radio network. It cannot monitor transmissions from other slave radios or transmit data to them.

In our laboratory experiments, we were able to scan combinations of the five remaining parameters at a rate of 12 seconds per combination. There are 539 400 possible legal combinations. At 12 seconds per combination, this yields a total search time of approximately 75 days to learn the five remaining parameters and enable rogue slave connection to the discovered radio network. We found experimentally by trial and error that if we wait 10 seconds per combination there will always be at least one connection, and corresponding reported disconnection, for the correct parameter set. The remaining two seconds per combination is due to latency in the script. Like the scan described in the previous section, this scan also yields false positives in addition to the correct parameter values. The false positives result from the sporadic connections described above. However, we have observed that a false positive is very unlikely to sporadically connect again when rescanned. In our laboratory experiments we were always able to eliminate false postives by rescanning all combinations marked as correct from the original scan two more times. Our experimental scans returned between 0.04 and 3 percent false positives. For the worst case of 3 percent false positives rescanning just the combinations marked as correct would require an additional 2.25 days based upon 12 seconds per combination.

We analysed the false positives returned from the above scan in an attempt to learn if the false positives provide any indication of the correct configuration or if they are random in nature. We were unable to find any correlation between the false positives and the correct parameter values. Figure 5 is a histogram showing the number of correct parameters found in each combination returned by the scan as potentially correct. Of the returned combinations 14 had no correct parameters, 16 had only one correct parameter, and more than 95% of the combinations returned had less than half of the parameters correct. Figure 5 provides circumstantial evidence for our belief that the sporadic connections are caused by random collisions in the spectrum.

C. Compromised Radio Networks

Once all parameters from Table II are learned, a radio can be used to connect to the discovered radio network as a rogue slave. A rogue slave can initiate a denial of service attack, eavesdrop on all master to slave transmissions, and inject data into the network. Malformed packets, false data, and random bits are examples of data that can be sent to the master.

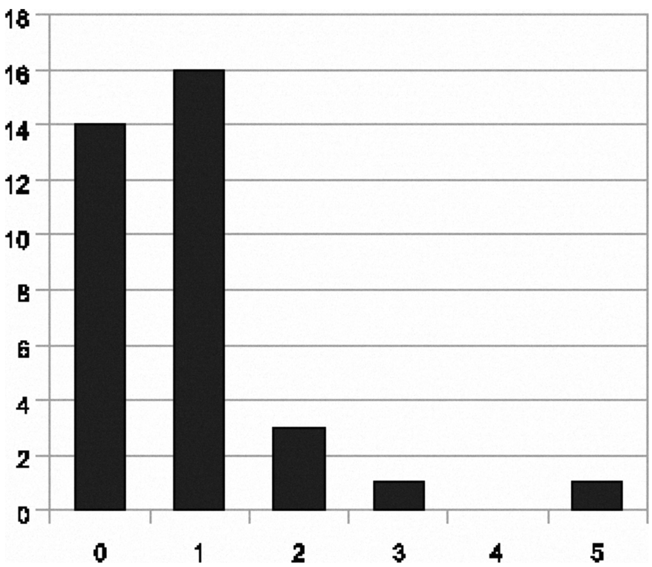

Fig. 5. Histogram of Number of Correct Parameters in False Positives

1) Half-Duplex Denial of Service Attack: We have implemented a denial of service attack which disrupts the half-duplex link between slaves and masters, making it impossible for slaves to communicate to masters. The attack does not prevent masters from sending packets to the slaves. The attack works by a rogue slave flooding the radio network with data. A rogue slave can transmit data continuously for any length of time. As long as the rogue slave is transmitting continuously, other radios in the system cannot transmit any amount of data to the master. The data transmitted by the rogue slave may be random, or it could consist of carefully crafted protocol packets (for example, Modbus) designed to imply a smoothly running process to an operator or a MTU.

The radio communicates using a half-duplex slotted protocol. Figure 6 shows graphically how the duplexing is accomplished. In each slot, the master radio is allocated a portion of the slot to transmit, and the remainder of the slot is allocated to the slaves. Slaves only transmit within this second portion of the slot. There is no arbitration when multiple slaves wish to connect. Instead, slaves monitor the bus for other transmitting slaves and do not transmit if another slave is currently transmitting. Slaves queue their transmissions until they find the bus is available. When the rogue slave wishes to transmit it will wait until the bus is available. Since ordinary control system traffic occurs in short bursts, the rogue slave easily finds an available slot to transmit. The denial of service attack works because the rogue slave transmits continuously causing other slaves in the network to always find the bus busy. When the rogue slave finally stops transmitting the other slaves in the network can begin to empty their tranmission queues. If the denial of service attack continues long enough slaves will likely overflow their transmission queues and data will be lost.

Because slaves in the network cannot transmit sensor data to the master node the feedback control loop is broken. However, since it is denied sensor data, the control commands issued by a master during this denial of service attack are based on old or false information and therefore erroneous.

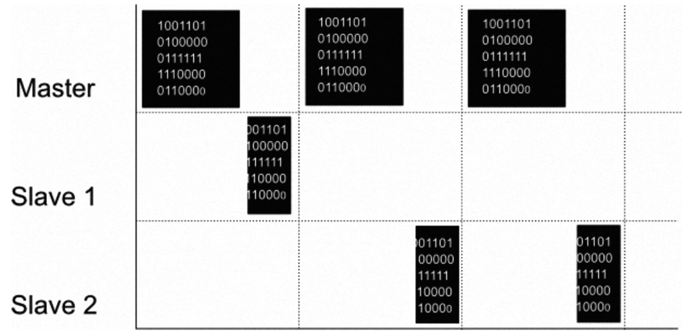

Fig. 6. Duplexing in the Radio Network

Process control systems use multiple layers of feedback control to control a physical process. Because slaves in the network cannot transmit sensor data to the master node during the denial of service attack the feedback control loops are broken, causing master nodes to make control decisions with incorrect sensor data. In the Bellingham, Washington pipeline explosion discussed in the introduction, there was a human operator who was to open a release valve if the measured pressure was at an unsafe level. In our process control systems laboratory, we were able to duplicate a similar scenario using Commercial Off-The-Shelf process control system hardware, including the wireless radio studied, and a laboratory scale pipeline. In the scenario, a pump steadily increases pipeline pressure while human machine interface (HMI) software displays pipeline pressure and allows the operator to control the pipeline procress control system by shutting down the pump and by opening a pressure relief valve. We launched the denial of service attack while the pressure was at a safe level. Once the denial of service attack started the HMI software no longer received updated transmissions from the process control system slave device monitoring pipeline pressure. As a result the pipeline pressure increased significantly beyond the safety limit while the HMI continued to display the last received pressure value. Figure 7 shows a picture of the pipeline's digital pressure gauge overlayed on a screen image captured from the human-machine interface (HMI) software screen that controlled the system ; the pipeline is at an "unsafe" level as shown by the 54.7 PSI on the digital gauge, while the HMI shows a "safe" level of 17.9 PSI.

Process control systems typically use either MODBUS or DNP3 as a network layer protocol. Although it is possible to craft packets MODBUS or DNP3 packets to send during the denial of service attack, this is not necessary. For our implementation of the described denial of service attack, we used a terminal program to send a continuous stream of random data. The transmitted data is interpreted as by the master node as MODBUS packets with bad checksums and are dropped.

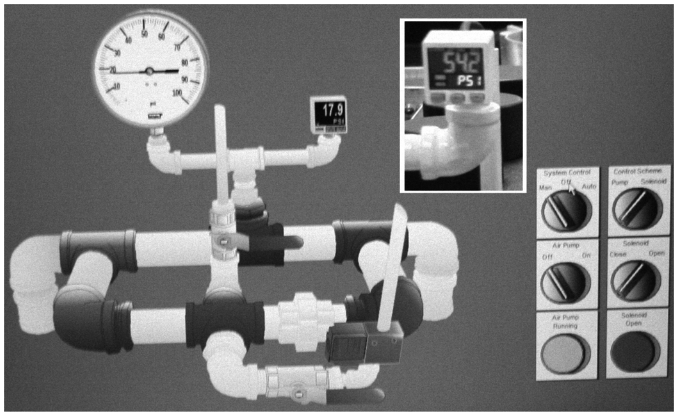

Fig. 7. HMI and Actual Pressure Gauge of Model Pipeline Under Attack

2) Other Attacks: A rogue slave connected to a radio network can monitor all transmissions from the master terminal unit in the process control system. This is possible because all transmissions from the master node are broadcast to all connected slaves. The data transmitted from master terminal unit to remote terminal unit may interest an industrial spy as it could reveal information about a company's particular process controls.

The denial of service attack discussed above injects malformed data into the radio network. A rogue slave connected to a radio network may send any type of data it desires. It may send partially or incorrectly formed network layer packets to attempt to induce protocol failures or unintended protocol responses. A rogue slave may also examine the packets sent by the MTU and respond with answering packets impersonating an RTU. Such an attack could allow a potential saboteur to send information to the MTU that would cause it to change the system in undesired or even dangerous ways. For example, opening the valves of a chemical process at the wrong time could cause a chemical spill.

IV. Potential Countermeasures and Future Work

A. Potential Countermeasures

Although the vendor will be directly notified, we list here a number of potential suggestions for the benefit of the community. To create a more secure radio system, we suggest the following countermeasures:

1) Eliminate sporadic network connections
2) Better authentication mechanism
3) More effective serial number list

First, the scans we used to find radio networks take advantage of the fact that the tested radios connect sporadically even when a serial number list serial number/RF data rate pair or network identifier/RF data rate pair is correct regardless of all other configuration values. We believe this occurs when the frequency-hopping radios temporarily have overlapping carrier frequencies. The radio protocol should not allow these sporadic connections to occur.

Second, the convenience of being able to maintain a network based upon a common characteristic, like a network identifier, is an important and useful feature. However, relying upon this single factor to provide access control is fallacial; a 12-bit integer is not strong enough to prevent a determined attacker. We recommend that an authentication method similar to Public Key Infrastructure (PKI) be used to authenticate connected radios. The new authentication method should take into account the potentially limited microprocessor resources available on the radios. If processor resources are limited, we recommend elliptic curve cryptography (ECC) in place of RSA. In fact, we recommend that all process control system components, users, and network traffic be authenticated. Authenticated radios could generate dynamic encryption keys to support data confidentiality.

Third, the serial number list feature that defines the serial numbers of radios allowed to join a radio network is a promising security feature, providing that it is difficult to spoof the serial number built into the radio. However, we found that the serial number list feature was was easy to defeat since we could easily learn the master radio's serial number and add it to our slave's call book. If both master and slave radios checked serial number lists, rather than just slaves, it would be much more difficult to infiltrate a radio network.

Although we have not yet discovered similar vulnerabilities in point-to-point networks, we have not completed enough research to definitively assert their security. Furthermore, a point-to-point network is simply ill-suited to many process control systems network needs, so recommending the use of only point-to-point networks to provide security is unsatisfactory.

B. Future Work

Presently, we do not know of a simple mechanism, such as a set of configuration parameters, to prevent the attacks documented in this paper. However, a diagnostic tool available with the tested radios is capable of detecting the presence of all radios in an area. This software could be modified to alert intrusion detection systems of a possible intrusion and initiate mitigation. Methods for detecting scans of the type discussed in this paper should be developed and integrated into intrusion detection systems.

The scans used to discover radio networks and learn the configuration values necessary to connect as a rogue slave are time consuming. While we believe the target value justifies the long waits for certain attackers, we also believe work can be done to significantly speed up the scans.

Security reviews and investigations should be conducted on all existing process control system hardware, software, and networking protocols to discover and mitigate security vulnerabilities. As new standards and equipment are developed for this industry security must be a key priority.

V. Conclusion

Process control systems control both industrial and critical infrastructure systems, which for public safety and economic security must be secure. In this paper we described methods to discover, infiltrate, and deny service to wireless networks used with process control systems (also known as SCADA control systems). The attacks documented are specific to radio networks built with popular model of radio used in the process control system industry. However, as many vendors in this domain promise radio compatibility across models, and even sometimes brands, we believe the described attacks may affect more than the specific radio tested. The radio tested relied primarily on uniqueness of radio parameters for security. Effectively, this is a form of "security by obscurity," which has repeatedly been shown to be ineffective at preventing a determined attacker. Additionally, the tested radios are well inside any available firewall or authentication mechanisms intended to protect the process control and therefore provide backdoors into even well-managed, otherwise secure process control system networks.

Acknowledgment

The authors acknowledge the support of a Summer Undergraduate Research Fellowship from the Shackouls Honors College, Mississippi State University. This work is partially funded by the Department of Homeland Security under task order # 4000055437.

References

[1] Slay, J. and Miller, M., 2008, "Lessons Learned from the Maroochy Water Breach," IFIP International Federation for Information Processing, Volume 253 , Critical Infrastructure Protection, eds. E. Goetz and S. Shenoi; (Boston: Springer), pp. 73-82.

[2] Abrams, M. and Weiss, J. "Bellingham, Washington, Control System Cyber Security Case Study." National Institute of Standards. Information Technology Laboratory. August, 2007. Sampled on January 26, 2009. http://csrc.nist.gov/groups/SMA/fisma/ics/documents/Bellingham_Case_Study_report%2020Sep071.pdf .

[3] Abrams, M. and Weiss, J. "Malicious Control System Cyber Security Attack Case Study-Maroochy Water Services, Australia," National Institute of Standards. Computer Security Resource Center. August, 2008. Sampled on June 29, 2009. http://csrc.nist.gov/groups/SMA/fisma/ics/documents/Maroochy-Water-Services-Case-Study_briefing.pdf .

[4] McGrew, R. W. and Vaughn, R. B., "Discovering vulnerabilities in control system humanmachine interface software" *Journal of Systems and Software*, Volume 82, Issue 4, pp 583-589, Apr. 2009.

[5] Lyon, G. , "The Art of Port Scanning", *Phrack Magazine* Volume 7, article 11, Issue 51. September,1997.

[6] Ning, C., Wang, J., and Xinghuo Yu, "SCADA System Security: Complexity, History and New Developments," *IEEE International Conference of Industrial Informatics*, Daejeon, South Korea, July, 2008.

[7] Grzelak, D. "SCADA Penetration Testing: Hacking Modbus Enabled Devices, " *Ruxcon*, Sydney, Australia, November 2008. Sampled on June 29, 2009. http://www.ruxcon.org.au/files/2008/SIFT-Ruxcon2008-SCADA-Hacking-Modbus-Enabled-Devices.pdf