Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE
Corso di Laurea in Informatica

Analisi di vulnerabilità e Penetration Test su Architetture SCADA

Relatore:
Chiar.mo Prof.
MARCO PRANDINI

Presentata da:
SIMONE FERRAGUTI

Correlatori:
Dr. DAVIDE BERARDI
Dr. LORENZO RINIERI
Dr. AMIR AL SADI

Sessione III
2021/2022

Indice

Elenco delle figure

Elenco dei Codici

Capitolo 1

Introduzione

In questa tesi si è analizzato il comportamento di una infrastruttura informatica industriale, simulando un ambiente di lavoro reale, dove fosse presente uno sviluppatore intento a configurare quest'ultima e venisse presa di mira da un attacco informatico. Verranno trattati alcuni dei principali protocolli di comunicazione utilizzati, come *Modbus* e *Common Industrial Protocol (CIP)*. I livelli di sicurezza informatica dei Programmable Logic Controllers (PLC) possono essere suddivisi in due categorie principali: sicurezza fisica e sicurezza logica [5, 6].

- La **sicurezza a livello fisico** riguarda la protezione dei PLC stessi e dei suoi ambienti di funzionamento da accessi non autorizzati, danni accidentali e dalle interferenze esterne. Ciò può essere ottenuto attraverso la protezione fisica del dispositivo, l'utilizzo di controlli di accesso e l'utilizzo di protocolli di comunicazione sicuri.

- La **sicurezza a livello logico** riguarda la protezione del software e dei dati all'interno del PLC. Ciò include la protezione contro l'hacking, la modifica non autorizzata del software, la raccolta non autorizzata di dati e la sicurezza dei dati in transito. Ciò può essere ottenuto attraverso la crittografia, l'autenticazione e la sicurezza dei dati in transito.

In generale, i PLC possono essere vulnerabili ad attacchi informatici a causa della loro connessione a reti aperte, alla loro natura di dispositivi industriali che nel tempo hanno avuto la necessità di interfacciarsi con le apparecchiature IoT. I vecchi sistemi PLC spesso non hanno funzioni di sicurezza integrate e possono essere vulnerabili ad attacchi ed exploit noti. Gli attacchi possono causare interruzioni del sistema, la modifica di parametri di processo o la raccolta di informazioni sensibili. Dare una semplice risposta alla domanda: "Come rendere sicuri questi dispositivi?" non è facile, poiché dipende da molti fattori. Per cercare di prevenire questi problemi, è importante prima di tutto mantenere i PLC aggiornati con le ultime patch rilasciate dai produttori. Inoltre, utilizzare sistemi di monitoraggio, di gestione degli eventi di sicurezza, sistemi di crittografia e di autenticazione implementati appositamente per il contesto industriale. In questo trattato, in sezione 4, verranno mostrate quali sono le metodologie che un attaccante si potrebbe trovare ad poter eseguire in un ambiente Supervisory Control and Data Acquisition (SCADA). Successivamente, in sezione 5, verrà analizzato nel dettaglio come è stato sviluppato l'ambiente di lavoro. Infine, in sezione 6, dopo una preliminare fase di profiling votata ad apprendere quante più informazioni sul sistema in analisi, verranno eseguiti una serie di attacchi ed exploit. I risultati e i dati carpiti saranno analizzati in sezione 7.

Capitolo 2

Il contesto industriale

2.1 Cosa sono i PLC

I PLC (Programmable Logic Controllers) sono dispositivi elettronici utilizzati per automatizzare processi industriali e di produzione. Essi sono in grado di controllare e monitorare l'attuazione di una serie di comandi predefiniti, attraverso l'utilizzo di un software di programmazione specifico. I PLC sono utilizzati in una vasta gamma di settori, tra cui l'automazione industriale, l'automazione dei processi, l'automazione dei trasporti, l'automazione dei sistemi di sicurezza e molto altro ancora.

Il loro funzionamento si basa sull'elaborazione di segnali elettrici in ingresso e in uscita, attraverso l'utilizzo di circuiti elettronici integrati e di un microprocessore. I PLC possono essere programmati per svolgere una vasta gamma di funzioni, tra cui la gestione dei motori, la regolazione dei processi, la gestione degli allarmi e la raccolta di dati. Essi possono essere facilmente programmati e modificati per soddisfare le esigenze specifiche di un determinato processo o sistema, e possono anche essere integrati con altri dispositivi di automazione, come sensori e attuatori.

2.2 Casi d'uso

Proprio a causa della loro versatilità vengono utilizzati in una vasta gamma di contesti industriali e di produzione, certuni dei quali considerati ad alto rischio[5], alcuni esempi sono:

1. Sistemi di controllo di processo in ambito chimico e petrolchimico: gestione dei flussi di materie prime, la regolazione della temperatura e la gestione della pressione.

2. Sistemi di controllo di processo in ambito minerario: controllo e monitoraggio dell'estrazione e lavorazione delle materie prime, tra cui la gestione dei motori, la regolazione dei processi e la gestione degli allarmi.

3. Sistemi di sicurezza in ambito nucleare: gestione dei sistemi di raffreddamento, la regolazione dei processi e la gestione degli allarmi.

4. Sistemi di controllo di processo in ambito automobilistico ed aerospaziale: gestione dei motori, la regolazione dei processi e la gestione degli allarmi.

In questi contesti, i PLC sono utilizzati per garantire la sicurezza del processo industriale e per prevenire incidenti che potrebbero causare danni alle attrezzature, alle persone o all'ambiente circostante.

2.3 Pericoli

In un contesto industriale o di produzione, un attacco informatico ai PLC potrebbe causare una serie di conseguenze negative, tra cui:

1. Interruzione del processo: la compromissione dei sistemi di controllo potrebbe portare all'interruzione del processo industriale o di produzione, causando potenzialmente danni alle attrezzature, rallentamenti nella produzione e perdite economiche.

2. Modifica dei parametri di processo: la modifica non autorizzata dei parametri di processo potrebbe causare la produzione di prodotti difettosi o la generazione di sostanze pericolose per l'uomo e per l'ambiente.

3. Danni alle attrezzature: la compromissione dei sistemi di controllo potrebbe portare al danneggiamento di attrezzature e alla necessità di riparazioni costose.

4. Perdita di dati: la compromissione dei sistemi di archiviazione dei dati avrebbe come risultato la perdita di dati importanti o il furto di informazioni sensibili.

5. Rischio per la sicurezza delle persone: in alcuni casi, un attacco informatico potrebbe causare la compromissione dei sistemi di sicurezza, mettendo a rischio la sicurezza delle persone.

2.4 Vulnerabilità

Attualmente, i PLC presentano ancora molti dei problemi che li hanno contraddistinti nel tempo. La manutenzione dei PLC può essere complessa e richiedere competenze specialistiche, rendendo difficile la riparazione o la sostituzione dei componenti difettosi. Data la loro natura industriale, i produttori sono numerosi e questo ha portato nel tempo difficoltà di interfacciamento, rendendo complessa l'interoperabilità, anche a causa della volontà delle aziende di mantenere il codice **Closed Source**. Seppure nel tempo possa essere stata scoperta una vulnerabilità dovuta ad una criticità del sistema, si è tendenzialmente preferito rilasciare patch piuttosto che ripensare l'architettura, proprio a causa delle difficoltà di interoperabilità e manutenzione. Un'ulteriore esempio di criticità è la possibilità di essere vulnerabili alla modifica non autorizzata del software, alla raccolta non autorizzata di dati e alla sicurezza dei dati in transito. La causa principale di tali problematiche sono le insicurezze intrinseche dei protocolli di comunicazione utilizzati, che rendono necessario l'utilizzo di protocolli di sicurezza specifici.

Capitolo 3

SCADA

Supervisory Control and Data Acquisition, nel lessico dei controlli automatici, indica un sistema distribuito per il monitoraggio e la supervisione di sistemi fisici. La figura 3.1 mostra un esempio di infrastruttura SCADA dove una porzione di sistema è adibita al monitoraggio, una sezione è adibita al controllo ed infine un'ultima sezione che ha lo scopo di permettere a degli operatori umani di interagire con il sistema nella sua totalità (Human Machine Interface (HMI)). Le debolezze che caratterizzano questa tipologia di sistema sono:

- La maggior parte dei protocolli sono derivati dal vecchio standard di comunicazione Seriale RS-485 [7].
- Assenza di:
  - Autenticazione (Nella trasmissione dei messaggi non viene eseguito nessun controllo della fonte)
  - Integrità (Non inteso come correzione di errori. Non è possibile capire se i messaggi siano stati modificati all'interno di una comunicazione fra due dispositivi, quindi non è presente un controllo di manomissione)
  - Confidenzialità (Messaggi trasmessi in plain-text)

Possiamo dividere il sistema SCADA in 2 *layer*: il *layer* client, che è in pratica il lato in cui gli operatori interagiscono e *layer* "data server", dove avvengono la maggior parte delle attività di controllo dei dati. I server dati comunicano con i dispositivi attraverso controllori, come ad esempio PLC a cui sono collegati direttamente attraverso la rete oppure Fieldbus.

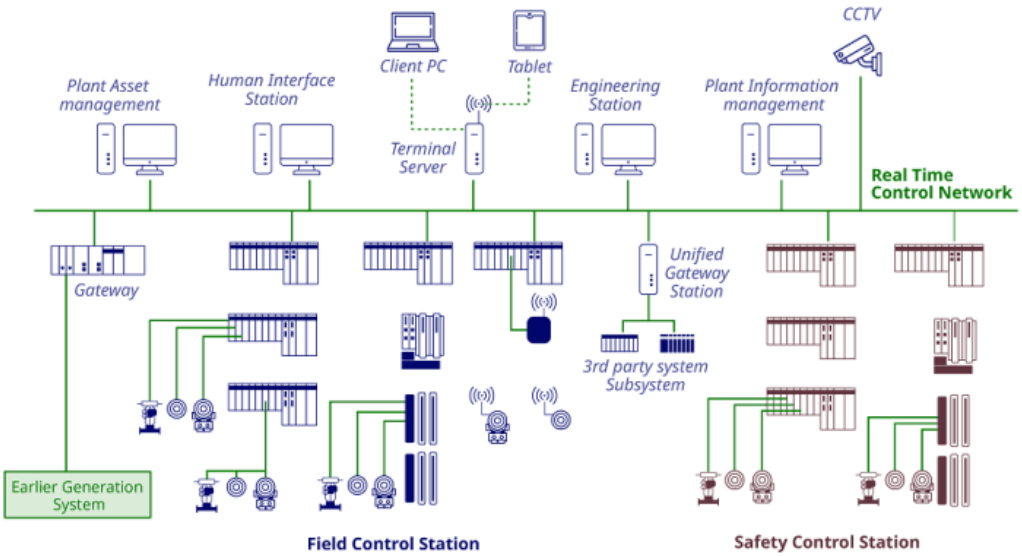

Figura 3.1: Esempio di implementazione di una rete SCADA [1].

3.1 Comunicazione

La comunicazione server-client e client-server è in generale una comunicazione basata ad eventi, ed utilizza principalmente il protocollo TCP/IP. L'applicativo **client** sottoscrive (imposta, modifica, legge) un parametro che è "esposto" dall'applicativo **server**, che successivamente comunica la risposta al **client**. Nonostante la comunicazione sia ad eventi, data l'assenza di un meccanismo publish/subscribe a livello di rete, le letture e scritture di parametri tra **client** e **server** devono essere effettuate ciclicamente tramite un'interazione a polling. Nel tempo i protocolli di comunicazione utilizzati dagli Industrial Control System (ICS) sono stati vari, dato lo sviluppo della

tecnologia, ma senza mai effettuare un cambiamento veramente strutturale. Prendendo come esempio il più classico tra i protocolli di comunicazione: *Modbus*, ha avuto una evoluzione nel tempo, ma la struttura di base è rimasta invariata. Basti pensare che seppure le aziende hanno implementato nuovi standard di comunicazione per gli ICS, come ad esempio CIP, la maggioranza dei dispositivi supporta ancora a pieno il protocollo *Modbus* [8]. In questa tesi, in particolare, verranno trattati i protocolli Modbus ed Ethernet/IP.

3.2 Modbus

È il protocollo SCADA più popolare: molto semplice, basato su reti seriali, comandi codificati in funzioni all'interno del codice ed è "Open Source"(non sono richieste licenze). Generalmente tutti i dispositivi industriali hanno protocolli proprietari di comunicazione, ma in ogni caso supportano in aggiunta questa tipologia di comunicazione. È diventato uno standard de facto nella comunicazione di tipo industriale ed è attualmente uno dei protocolli di connessione più diffusi al mondo fra i dispositivi elettronici industriali. Modbus si posiziona a livello 7 nella pila ISO/OSI [9], definendo la formattazione dei messaggi detta *framing* e la modalità di trasmissione dei dati e delle funzioni di controllo. La comunicazione avviene tramite il paradigma **client-server**. Il protocollo definisce un **Protocol Data Unit (PDU)** che non dipende dal sottostante layer di comunicazione. L'Application Data Unit (ADU) introduce campi addizionali per l'indirizzamento e il controllo dell'errore. Esistono diverse versioni del protocollo ModBus, ASCII e RTU operano su reti implementate su tecnologie seriali RS232, RS422 e RS485, mentre la versione TCP/IP opera su reti che supportano la suite TCP/IP.

3.2.1 Modbus Seriale

ModBus seriale definisce la struttura dei messaggi che vengono spediti e che i dispositivi, distinti in master (client) e in slave (server), interpretano indipendentemente dalla tipologia di rete (client-server) su cui lavorano. Il

protocollo ModBus seriale definisce due tipi elementari di dato: *discrete input* e *input register*. Il tipo di dato discreto rappresenta un valore (bit) per indirizzare gli output (coils) di un PLC, mentre il dipo di dato input register rappresenta valori interi a 16 bit. Molto spesso viene utilizzata una coppia di valori di tipo registro per rappresentare float a 32 bit – attenzione in questo caso l'ordinamento è *little endian*. In generale questo tipo di dato, ad esclusione dei float a 32 bit, viene trasferito con ordinamento *big endian*. Per la parte seriale esistono due diverse modalità di formattazione dei messaggi:

- **Modbus RTU** che fornisce una rappresentazione binaria – compattezza dei dati.

- **Modbus ASCII** versione più *Human-Readable* – i dati (indirizzi, funzioni, eccetera) sono leggibili da parte dell'utente poiché vengono codificati in esadecimale.

In entrambi i casi viene utilizzata una comunicazione seriale. RTU fa seguire a comandi, funzioni e dati un checksum di tipo Cyclic Redundancy Check (CRC) mentre per l'ASCII viene generato un checksum di tipo Longitudinal Redundancy Checksum (LRC). Nodi RTU non possono comunicare con nodi ASCII e viceversa.

Ci sono da sottolineare alcune limitazioni del protocollo ModBus seriale, utilizzando un paradigma master-slave non c'è modo per un dispositivo di forzare un invio di dati/eccezione. Il nodo master deve continuamente (pull) interrogare gli slave per conoscere eventuali modifiche nei dati. L'indirizzamento Modbus permette il collegamento di un massimo di 254 dispositivi, fattore che limita il numero massimo di dispositivi connessi ad una stazione master.

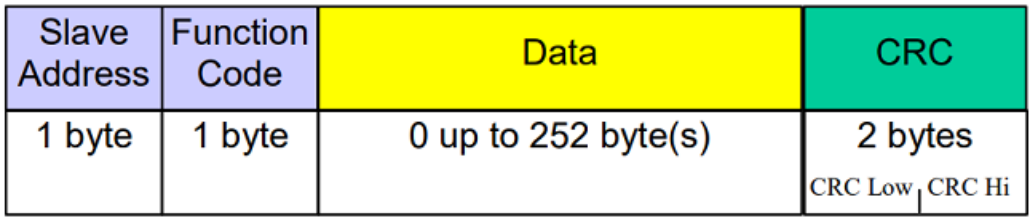

Figura 3.2: Modbus RTU Frame [2].

In riferimento alla figura 3.2.1 il frame Modbus RTU è composto da:

- Slave ID: Indirizzo univoco da 1 a 247.
- Function Code: Comunica ai dispositivi Slave che comando eseguire.
- Data: Dipende dalla funzione comando utilizzata.
- Cyclic Redundancy Check (CRC): Controllo utilizzato per la rivelazione di errori. Non corregge o identifica errori controllando l'integrità del messaggio, ma semplicemente si limita a rilevare se ce ne sono stati nella trasmissione.

3.2.2 Modbus Frame su TCP

Nel tempo Modbus è stato aggiornato per essere compatibile con le reti TCP/IP, aggiungendo un frame TCP in testa al frame Modbus, il che consente comunicazioni su reti internet/intranet. Il protocollo Modbus TCP/IP utilizza una codifica binaria dei dati e implementa un meccanismo di rilevamento errori TCP/IP, a differenza del Modbus seriale. La versione TCP/IP è orientata alle connessioni e permette di eseguirle in modo concorrente sullo stesso slave o su più dispositivi. Anche Modbus TCP/IP utilizza il paradigma master-slave. Il frame del messaggio Modbus TCP/IP è differente rispetto al Modbus seriale.

Il **Modbus Application Protocol (MAP) Header** identifica l'ADU e, a differenza del modello seriale, questo è identificato da un byte "unit

identifier" che corrisponde all'indirizzo dello slave del protocollo RTU. Questo è utile per avere un router/gateway con un singolo indirizzo IP che può supportare più unità indipendenti. Le richieste nel ModBus TCP/IP vengono costruite di modo che il dispositivo ricevente possa verificare la fine del messaggio. Per fare questo nel MAP Header vengono inserite informazioni aggiuntive in modo da permettere al ricevente di identificare i limiti del messaggio, molto utile se il messaggio è stato spacchettato in più pacchetti.

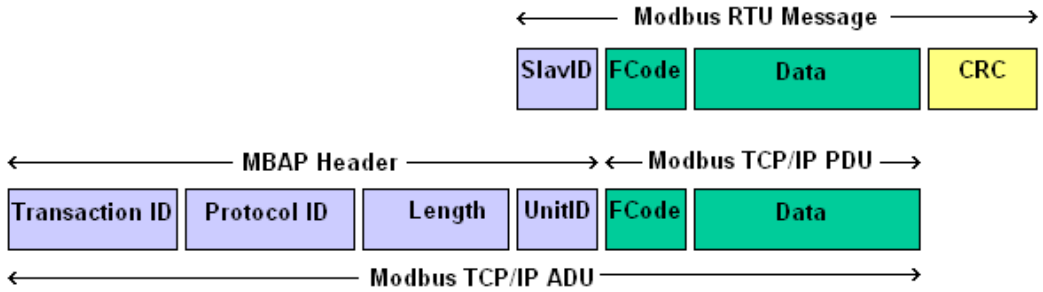

Figura 3.3: Modbus TCP Frame [3].

In riferimento alla figura 3.2.2 il frame Modbus TCP è composto da:

- Transaction ID: un'identificativo di transazione per tenere traccia di tutte le transazioni che vengono effettuate. Se l'host effettua una transazione viene assegnato un numero casuale all'ID transizione in modo che quado vengano ricevute una serie di risposte, è possibile interpretare a quale query ogni risposta fa riferimento.

- Protocol ID: in origine Modbus su TCP è stato pensato per poter interagire con altri protocolli, quindi questo identificativo fa riferimento al protocollo con cui Modbus si deve interfacciare. Ad oggi non è utilizzato se non in implementazioni particolari.

- Length: lunghezza del Frame successivo.

Dopo Lenght abbiamo semplicemente un Frame Modbus standard. È stato mantenuto lo slaveID per poter dare la possibilità, ai dispositivi compatibili, di convertire messaggi trasmessi su reti TCP/IP e trasmetterli su reti seriali. In questa maniera i dispositivi di nuova generazione hanno mantenuto la possibilità di comunicare con gli apparecchi che comunicano solamente attraverso la rete seriale. Nella comunicazione TCP/IP quel byte viene semplicemente ignorato.

3.2.3 CIP - Ethernet/IP

Il Common Industrial Protocol è uno standard per le applicazioni nell'industria dell'automazione. Ethernet/IP è una parte del protocollo CIP. Quest'ultimo definisce la struttura e la specifica del trasferimento dei messaggi. CIP implementato su Controller Area Network (CAN) definisce DeviceNet. CIP implementato su Ethernet definisce Ethernet/IP. Precedentemente conosciuto come *Control and Information Protocol*, CIP comprende una suite completa di messaggi e servizi per la comunicazione fra ICS[10, 11].

CIP utilizza un design object oriented per fornire ad Ethernet/IP profili di servizi e dispositivi necessari al controllo applicativo real-time in modo da promuovere un'implementazione consistente nelle funzioni dell'automazione per prodotti provenienti da ecosistemi diversi. Ethernet/IP è un protocollo che definisce caratteristiche e funzioni per i livelli 5, 6 e 7 all'interno del modello ISO/OSI [9]. Tutti i dispositivi dentro ad una rete Ethernet/IP trasmettono i dati come una serie di valori, chiamati *attributes*, che possono essere raggruppati insieme ad altre informazioni all'interno di set di attributi chiamati *objects*. Esistono *objects applications* che contengono informazioni per specifici dispositivi. Per esempio, un *EtherNet/IP drive device* potrebbe possedere un oggetto "motore". Tutti i dispositivi Ethernet/IP che supportano specifici strumenti hanno tutti lo stesso set di *EtherNet/IP application object*.

Sono presenti due tipologie di messaggi trasferiti tra un *EtherNet/IP scanner device* (apre la connessione e inizia il trasferimento dei dati) ed un

EtherNet/IP adapter devices (produce dati in risposta allo scanner). Questi messaggi possono essere messaggi espliciti (asincroni, alla necessità) e messaggi I\O (informazioni che devono essere trasferite in continuazione). In aggiunta, adatta elementi chiave dello standard Ethernet e servizi alla struttura del modello ad oggetti CIP, come ad esempio l'User Datagram Protocol (UDP), che Ethernet/IP utilizza per il trasporto di **messaggi I\O**.

Capitolo 4

Scenari di attacco

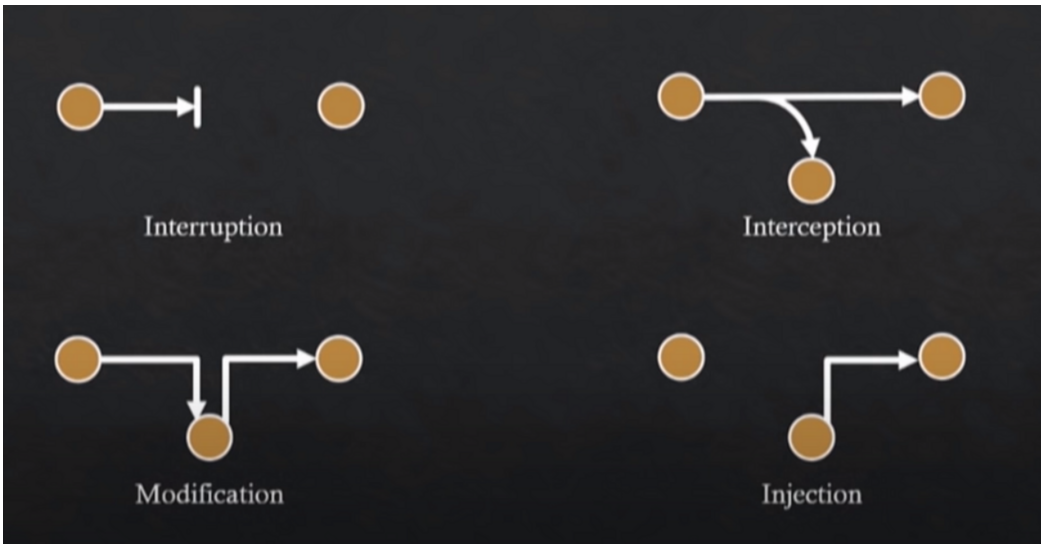

Figura 4.1: Scenari di Attacco [4].

Analizzando i comportamenti di un ambiente SCADA e le varie vulnerabilità che sono state scoperte nel tempo, le tipologie di attacco possono essere classificate principalmente in quattro gruppi. È possibile bloccare, intercettare e leggere il contenuto dei messaggi presenti nella rete, dato che non c'è nessuna codifica e nessuna confidenzialità. A conseguenza di ciò è anche possibile modificarne il contenuto e reindirizzarli. Prima di procedere con

alcuni esempi di attacchi perpetrabili è doveroso fare una premessa: quando si parla di codice malevolo o illecito non ci si riferisce solamente a codice che interrompe il normale funzionamento di un sistema disabilitandolo o rendendolo inefficace in vari modi. Esiste anche la possibilità che i dati vengano riconosciuti come leciti dagli ICS, ma che portino a risultati che non erano stati previsti dal progettista del sistema SCADA.

4.1 Interruption ed Injection

Un Attacco classico è quello del Denial of Service (DoS), che nell'ambiente SCADA può venir sviluppato, ad esempio, in due maniere: la prima, più classica, impossibilitando i vari ICS a poter comunicare fra di loro e quindi bloccando effettivamente tutti i processi in esecuzione dai vari dispositivi. Come seconda possibilità, l'attaccante mira a disabilitare funzioni specifiche dei processi di controllo, rendendo di fatto questo attacco più pericoloso del precedente. Per far intuire la pericolosità di quest'ultimo basti pensare, come esempio, ad una situazione in cui l'ICS in questione sia adibito al controllo delle paratie di una diga e che venga inibita la lettura dei sensori che verificano se queste ultime siano aperte o chiuse, potenzialmente aumentando la probabilità di causare ingenti danni.

4.2 Interception

L'attacco che mira ad intercettare informazioni scambiate all'interno dell'ambiente SCADA è reso possibile a causa dall'assenza di confidenzialità presente nei protocolli utilizzati per la comunicazione. Un attaccante collegato all'interno della stessa rete in cui comunicano i dispositivi di controllo può permettersi di eseguire uno *Sniffing* della intranet e leggere in chiaro le informazioni che si stanno scambiando i diversi dispositivi. Questo attacco è più semplice da eseguire rispetto alle altre tipologie, poiché non è necessario

intraprendere azioni all'interno della rete e ci si limita a carpire i dati in modo passivo.

4.3 Modification

Questa tipologia di attacco combina la fase di intercettazione ed interruzione della comunicazione fra dispositivi ed infine inietta codice malevolo all'interno dei target dell'attacco. Complicato poiché l'attaccante non solo deve intercettare i dati, ma deve prevenire che questi vengano recapitati al destinatario lecito della comunicazione e successivamente fingersi il trasmittente e inviare i dati illeciti al destinatario. La pericolosità di questo attacco sta nel fatto che chi riceve i dati crede di stare comunicando con un dispositivo *Trusted*, ma non ha modo di verificane la veridicità. Può essere perpetrato attraverso: hardware fisico, montando un dispositivo elettronico fisicamente in mezzo i dispositivi obiettivo dell'attacco, oppure tramite software. Nel caso di attacco tramite software l'attaccante in qualche maniera deve intercettare i dati del trasmittente, rallentare la consegna di quest'ultimi, modificarli, e trasmettere i dati illeciti prima che al destinatario vengano consegnati quelli leciti. In questa maniera anche se, successivamente, venissero consegnati i dati leciti verrebbero targati come *out-of-date* e quindi ignorati [12].

Capitolo 5

Ambiente di lavoro

Nelle prossime simulazioni si è presupposto di essere in uno scenario in cui l'attaccante si sia già infiltrato nella stessa rete in cui sono presenti i sistemi di controllo. Questa possibilità, per esempio, potrebbe verificarsi a causa di un'*Insider threat*, di un dispositivo compromesso oppure a causa di una configurazione errata all'interno della rete che l'ha resa vulnerabile. Conoscendo le fragilità dei protocolli [12] si è cercato di simulare un caso reale di utilizzo impiegando un PLC della nota marca **Rockwell Automation - Allen Bradley** modello **Micro850** collegato ad una macchina Windows contenente l'IDE proprietario del dispositivo che si sta analizzando. Durante le prove effettuate si è resa necessaria la progettazione e creazione di un circuito elettronico (figura 5.1), che forzasse un "hard reset" del dispositivo data la natura invasiva dei test, per poter ripristinare il funzionamento del PLC successivamente a certe tipologie di attacco. Alcuni test in particolare hanno causato un danneggiamento del software di funzionamento interno di quest'ultimo, per cui il riavvio del dispositivo non è stata più condizione sufficiente al ripristino e si è reso necessario ricaricare il programma tramite Integrated Development Environment (IDE) per poterlo riportare ad uno stato funzionante.

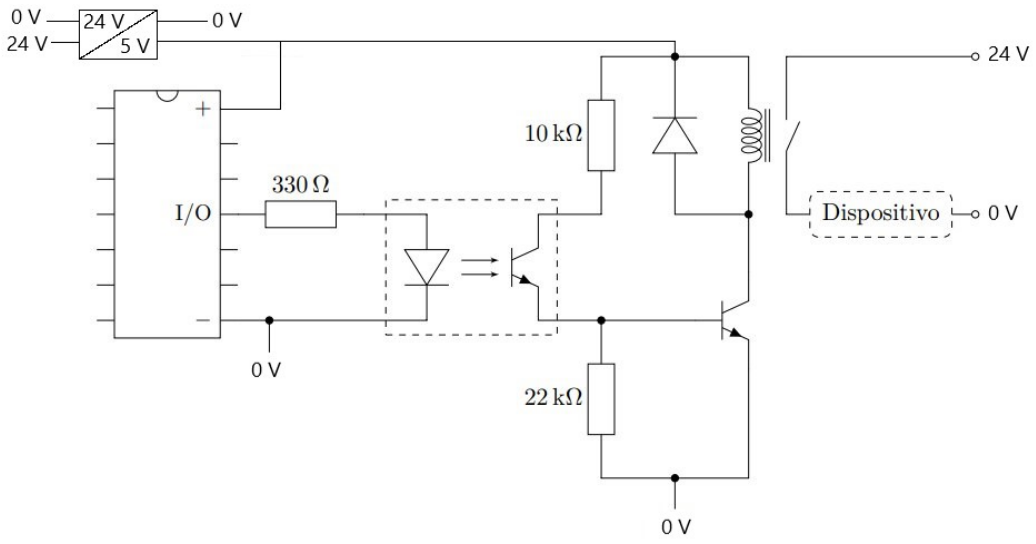

Figura 5.1: Circuito di controllo potenza.

In figura 5.1 è mostrato il circuito elettronico utilizzato per riavviare il PLC. Dato che l'alimentazione dei dispositivi industriali normalmente è 24VDC si è utilizzato un regolatore di tensione per abbassarla a 5VDC, utilizzata poi per alimentare il Controllore che comanda l'apertura e chiusura del relè e il circuito stesso. Siccome l'assorbimento massimo di un'uscita del Controllore utilizzato (in questo caso un *Raspberry pi*) è 16mA, è stato necessario includere un transistor NPN standard per fungere da protezione e allo stesso tempo controllare il relè. Come ulteriore protezione si è inserito un optoisolatore, poiché nel momento in cui il relè viene eccitato e poi diseccitato avviene una inversione di tensione ai capi dello stesso che potrebbe causare il danneggiamento dei GPIO del Controllore.

Capitolo 6

Attacco

Il lavoro è stato eseguito sulla base del modello *Black Box*, cioè si è cercato il più possibile di avvicinarsi ad una situazione ideale in cui un attaccante non conoscesse l'architettura dell'infrastruttura e quindi dovesse apprendere quante più informazioni possibili riguardo all'obiettivo su cui stesse eseguendo l'attacco.

6.1 Information Gathering

Dopo aver eseguito una scansione completa della rete ed individuato gli indirizzi dei dispositivi potenzialmente a rischio, si è proceduto con l'utilizzo del tool **enip-stack-detector**[13] con l'obiettivo di identificare possibili vulnerabilità. L'output relativo, riportato nella sezione di codice 6.1, mostra informazioni piuttosto utili, riguardanti: modello, produttore, vulnerabilità riscontrate e protocolli di comunicazione attualmente in uso sul dispositivo.

Codice 6.1: enip-stack-detector

```
==============Device================
[!] 192.168.1.11: 2080-LC50-48QWB (vendor:1 type: 14, v8.11)
================Tests===============
[ X ]: ENIP Register Session Number Sequential (Used value: 0x1)
[ X ]: ENIP Register Session Number Sequential (Used value: 0x10)
[ X ]: ENIP Register Session Number Sequential (Used value: 0x100)
[ X ]: ENIP Register Session Number Sequential (Used value: 0x1000)
```

```
[ X ]: ENIP Register Session Number Sequential (Used value: 0x10000)
[ X ]: ENIP Can Register Session with Bad Options (Used value: 1)
[ X ]: ENIP Can Register Session with Bad Length (Used value: 3)
[ V ]: ENIP Is List Targets Supported
[ V ]: ENIP List Services Protocol Version is 1
[ X ]: ENIP List Services Name is "Communications␣\x00" (with space)
[ X ]: ENIP List Services Name is "Communications\x00" (with single null (bug))
[ V ]: ENIP List Services Name is "Communications\x00\x00" (with nulls)
[ X ]: ENIP List Services Name is "COMMUNICATIONS\x00\x00" (upper with nulls)
[ V ]: ENIP List Services Name Capability Flags Reserved Bit Are Empty
[ V ]: CIP Forward Open is supported
[ X ]: CIP Forward Open allows multiple requests for connection id 0
[ X ]: CIP Forward Open is O2T Sequential by 1
[ V ]: CIP Forward Open is T2O zero
[ X ]: CIP Forward Open can open with bad connection flags
==============Results==============
[!] EtherNet/IP & CIP Stack: Rockwell LC 20/50 (sig: '0000000110010110010')
====================================
```

Delle informazioni ottenute si può estrapolare marca e modello del PLC che stiamo analizzando, ma ancora più interessante è la tipologia di protocollo usata per comunicare. Sfruttando un altro strumento, **libplctag**[14], entriamo in possesso della lista di variabili che sono state dichiarate all'interno del programma, i nomi e le quantità di ingressi ed uscite che sono presenti sul dispositivo, come possiamo notare nall'estratto di codice 6.2. Come informazione aggiuntiva apprendiamo anche che, connessi al dispositivo, ci sono anche svariati moduli aggiuntivi direttamente collegati.

Codice 6.2: libplctag

```
Tag "_IO_EM_DO_00" (00c1) BOOL: Boolean value.
tag string = "protocol=ab-eip&gateway=192.168.1.11&plc=Micro800&
    elem_size=1&elem_count=1&name=_IO_EM_DO_00"

Tag "_IO_X1_DO_00" (00c1) BOOL: Boolean value.
tag string = "protocol=ab-eip&gateway=192.168.1.11&plc=Micro800&
    elem_size=1&elem_count=1&name=_IO_X1_DO_00"

Tag "_IO_X1_ST_00" (00c1) BOOL: Boolean value.
tag string = "protocol=ab-eip&gateway=192.168.1.11&plc=Micro800&
    elem_size=1&elem_count=1&name=_IO_X1_ST_00"

Tag "_IO_X2_AI_00" (00c3) INT: Signed 16-bit integer value.
tag string = "protocol=ab-eip&gateway=192.168.1.11&plc=Micro800&
    elem_size=2&elem_count=1&name=_IO_X2_AI_00"
```

```
Tag "COUNT" (00c4) DINT: Signed 32-bit integer value.
tag string = "protocol=ab-eip&gateway=192.168.1.11&plc=Micro800&
    elem_size=4&elem_count=1&name=COUNT"

Tag "LIFE_TIME" (00c4) DINT: Signed 32-bit integer value.
tag string = "protocol=ab-eip&gateway=192.168.1.11&plc=Micro800&
    elem_size=4&elem_count=1&name=LIFE_TIME"

Tag "TEST_VALUE_02" (00da) Counted character sting with 1 byte per
    character and 1 byte length indicator.
tag string = "protocol=ab-eip&gateway=192.168.1.11&plc=Micro800&
    elem_size=80&elem_count=1&name=TEST_VALUE_02"
```

Individuando i protocolli di comunicazione in utilizzo, si hanno ora disposizione una serie di attacchi noti [12, 15, 16] che si possono effettuare.

6.2 DoS injection attack

Come primo approccio, conoscendo i nomi delle variabili inizializzate nel dispositivo, è possibile inviare comandi al PLC arbitrariamente in modo da settare quest'ultime a nostro piacimento[16], come mostrato nella sezione di codice 5.3.

Codice 6.3: invio di comandi arbitrari

```
##Scrivere un valore in un tag
>>>plc.write('dint_tag', 100)
Tag(tag='dint_tag', value=100, type='DINT', error=None)

##Scrivere molteplici valori tag differenti
plc.write(('tag_1', 1), ('tag_2', True), ('tag_3', 1.234))
[Tag(tag='tag_1', value=1, type='INT', error=None), Tag(tag='tag_2',
     value=True, type='BOOL', error=None), ...]

##Scrivere tag con valori Stringa
plc.write(('short_string_tag', 'Test␣Write'))
Tag(tag='short_string_tag', value='Test␣Write', type='STRING20',
    error=None)
```

La pericolosità che comporta il poter eseguire questi comandi è data dalla possibilità di inviare un numero elevato di istruzioni arbitrarie al PLC modificando valori di determinate variabili o nel caso peggiore settando specifiche uscite fisiche. Se inviati ad intervalli continui, questi comandi possono

comportare un Denial of Service (DoS). Citando anche l'esempio fatto nella sezione 4.1, possiamo fare un altro esempio pratico avendo a disposizione queste nuove informazioni. Se valutassimo l'attacco in un ambiente in cui il PLC si trovasse a comandare dispositivi come ad esempio le resistenze di una caldaia e l'attaccante iniettasse in lettura un valore di temperatura errato, le resistenze potrebbero continuare a scaldare l'acqua presente nel serbatoio oltre valori critici, causando nel peggiore delle ipotesi l'esplosione della stessa [16].

6.3 IDE sniffing

Analizzando il traffico di rete, durante il normale funzionamento del PLC, si è tentato di carpire informazioni scambiate tra quest'ultimo ed un dispositivo in cui vi fosse installato l'IDE proprietario. Con gli strumenti sfruttati fino ad ora si è stati in grado di legge e scrivere variabili a cui, normalmente, anche il programmatore ha accesso liberamente durante la creazione del software di controllo, ma all'interno della comunicazione fra IDE e PLC vengono scambiate anche altre informazioni. Per esempio, vengono scambiati valori associati a variabili di sistema, che dovrebbero essere nascosti ai dispositivi presenti nella rete. In figura 6.1 viene mostrato l'IDE di programmazione del PLC. Si noti come le variabli $\_\_SYSVA\_$ sono private e riservate al sistema, quindi non modificabili, mentre al di sotto di quest'ultime, sono definite le variabili globali che possono essere definite dal programmatore.

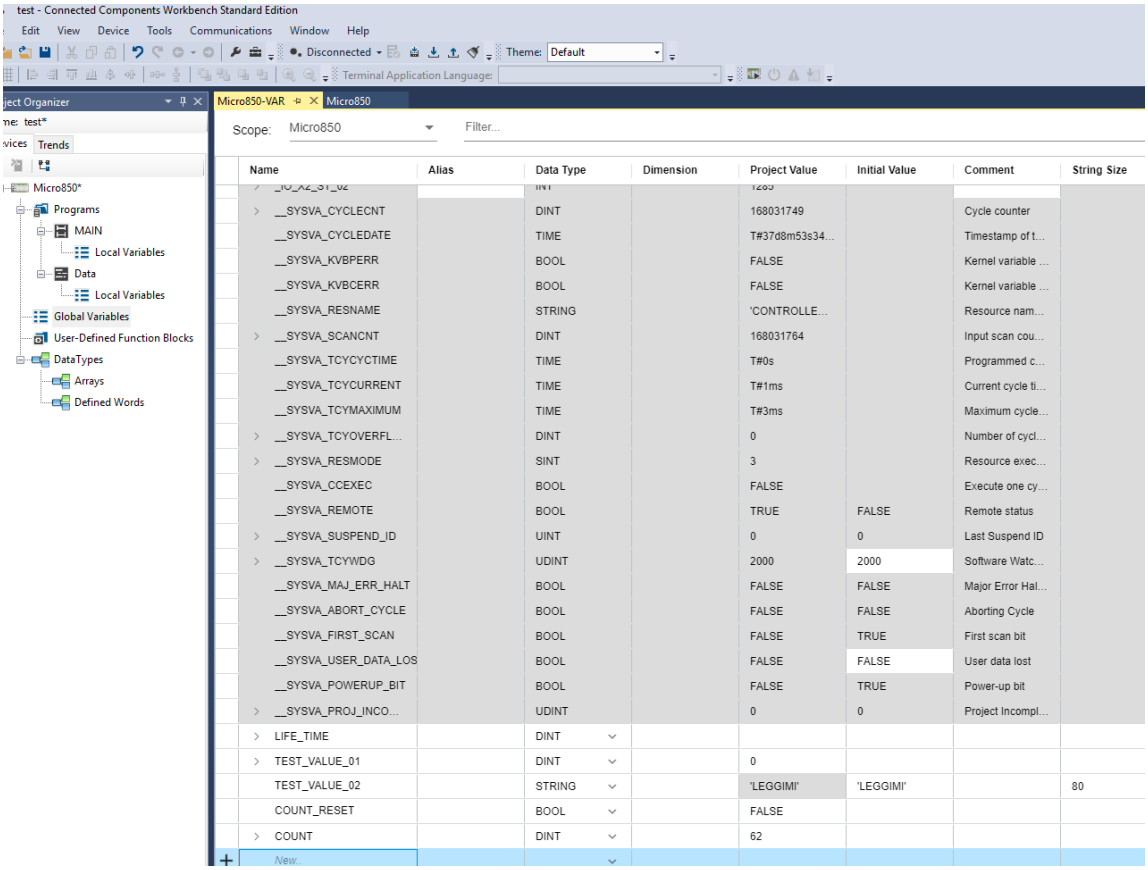

Figura 6.1: Variabili globali.

Sfruttando la vulnerabilità di cui si è parlato nella sezione 4.2, si è cercato di catturare il traffico scambiato fra i dispositivi. Guardando la figura 6.2, si può notare che fra i pacchetti scambiati si trova una stringa in chiaro. Controllando la figura 6.3 e la figura 6.1, tale stringa trova riscontro tra le variabili globali private che non dovrebbero potersi leggere a causa della loro natura protetta. Questo conferma come tutto il traffico sia scambiato in chiaro, senza nessuna confidenzialità, confermando le criticità già citate in precedenza (sezioni 3 e 4.2).

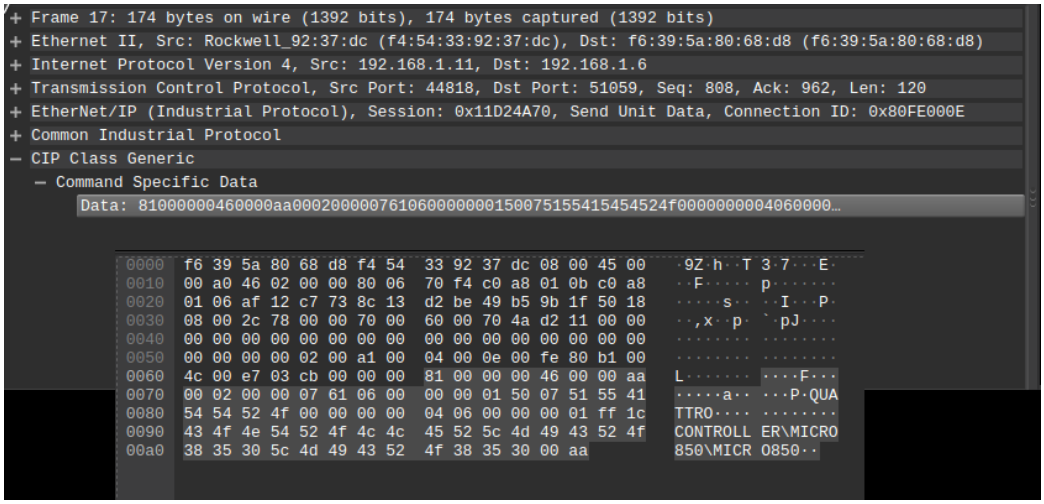

Figura 6.2: Traffico analizzato.

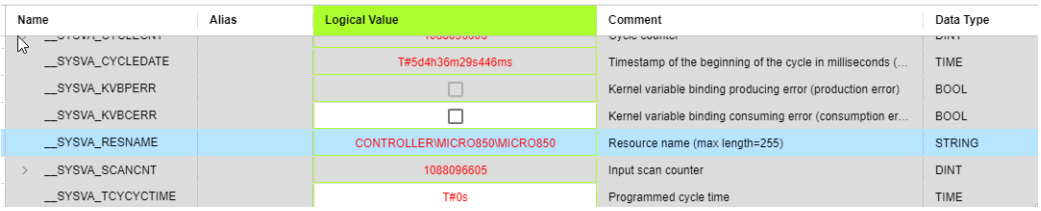

Figura 6.3: Variabili protette.

Utilizzando il programma **Wireshark**[1], si riesce a spacchettare e leggere tutto il traffico incorporato nel protocollo Ethernet/IP-CIP, ma a differenza delle stringhe non è possibile interpretalo completamente poiché quest'ultimo è impacchettato a sua volta all'interno della codifica proprietaria PCCC. Sfruttando quindi l'utilizzo di un parser in grado di interpretare PCCC si è in grado di leggere tutto ciò che viene scambiato all'interno della rete tra IDE e PLC. Come conseguenza di ciò, l'attacco dimostrato nell'articolo: "SCADA network forensics of the PCCC protocol"[15] sfrutta proprio questa vulnerabilità per poter eseguire un attacco di tipo Forensic Analysis. Quest'ultimo

[1] `https://www.wireshark.org/`

utilizza la vulnerabilità per analizzare il processo di trasferimento del programma di controllo sul PLC con lo scopo di estrarre artefatti digitali e presentarli in una forma *Human-Readable*. La valutazione dei risultati ottenuti è utile per identificare trasferimenti di logica di controllo al PLC, estrarli e memorizzarli in file sul disco in modo da compararli con artefatti template che possiede l'attaccante.

6.4 Exploit

Analizzando nello specifico il PLC che è sotto esame in questa tesi, conoscendo le insicurezze che affliggono questa tipologia di apparecchi, tramite lo strumento **enip-exploiter**[17] è facile notare che siano diversi gli attacchi disponibili per questa tipologia di dispositivi. Le vulnerabilità esplorate da questo strumento colpiscono in particolare la famiglia di PLC *MicroLogix*. I firmware affetti da queste problematiche arrivano fino all'ultima versione (FRN21.05) dove infine l'azienda produttrice ha introdotto una nuova modalità per la CPU chiamata "*Enhanced Password Security*". Quando questa modalità è abilitata ed il dispositivo è protetto da password, molti di questi exploit smettono di funzionare poiché questa modalità richiede l'autenticazione con password per ogni lettura/scrittura su memoria protetta. Detto questo, è importante ricordare che quest'ultima modalità non è abilitata di default, il programmatore deve abilitarla manualmente, quindi questo significa che il PLC potrebbe rimanere insicuro anche con l'ultima versione del firmware caricata. La lista degli attacchi effettuabili è la seguente:

- Accensione e spegnimento remoto del PLC.
- Lettura della password di protezione.
- Sovrascrittura della password di protezione (anche se quest'ultima è criptata).
- Modifica dell'IP del dispositivo da remoto.

- Cancellazione della memoria del dispositivo.
- Riavvio remoto forzato.
- Abilitazione forzata di tutti i protocolli di comunicazione (Il PLC può comunicare ultizzando protocolli diversi, anche quest'ultimi con le loro vulnerabilità. Con l'ultima versione del firmware, di default vengono disabilitati per sicurezza. Questo attacco li abilita tutti).
- Forzatura dello stato di FAULT.
- Blocco del dispositivo utilizzando un pacchetto Modbus corrotto (Non più perpetrabile con l'avvento dell'ultima versione del firmware).

Capitolo 7

Conclusioni

In questo trattato si è studiato il comportamento di un ambiente SCADA sottoposto ad un attacco informatico. I produttori di ICS stanno ricercando nuovi standard per contrastare le vulnerabilità di sicurezza degli apparati industriali che continuino a supportare dispositivi legacy, nativamente insicuri. Al contrario, le aziende propongono protocolli proprietari invece di cercare di uniformare l'insieme di apparecchiature che definiscono gli ambienti SCADA.

Nella prima fase ci si è concentrati sulla raccolta delle informazioni scansionando la rete ed individuando i dispositivi potenzialmente vulnerabili ad un attacco. I dati raccolti hanno mostrato che, una volta individuato il bersaglio, è possibile carpire numerose informazioni dai dispositivi PLC, tra cui il nome del produttore, modello, vulnerabilità di cui è possibilmente affetto e protocolli di comunicazione attualmente in utilizzo. Sfruttando le conoscenze acquisite, è stato possibile profilare numero e nomi di variabili globali, uscite ed ingressi in uso sul dispositivo.

Grazie a queste informazioni è stato possibile procedere ad eseguire un attacco di tipo DoS su un PLC Micro 850 della marca Rockwell Allen Bradley, inviando pacchetti arbitrariamente in modo da settare le variabili precedentemente identificate su valori non previsti. Prima di tutto, questo permette di disabilitare le comunicazioni fra i dispositivi presenti nella rete. In secondo luogo, eseguendolo mirando a specifiche aree di memoria nel PLC, consente

di creare malfunzionamenti puntuali.

Ipotizzando di essersi trovati in un ambiente in cui fosse stato presente un programmatore intento a configurare il dispositivo sotto analisi, si è proceduto con l'esecuzione di un attacco in cui si è cercato di intercettare i dati scambiati tra il PLC ed il programmatore. Questo ha permesso di scoprire che la comunicazione è priva di riservatezza, riuscendo a carpire informazioni che dovrebbero essere confidenziali.

Con a disposizione molte informazioni riguardanti il sistema, si è proceduto con l'esecuzione di una serie di exploit volti a evidenziare il numero di vulnerabilità che affliggono il PLC in esame.

Successivamente, sono stati implementati attacchi che permettono di abilitare forzatamente protocolli di comunicazione non nativamente abilitati sui PLC. L'attivazione di questi protocolli di comunicazione è condizione necessaria affinché altri exploit possano essere perpetrati.

Questo dimostra che, se anche un sistema non è vulnerabile a causa delle impostazioni di default imposte dal produttore o dalle configurazioni settate dal programmatore, è comunque possibile renderlo un sistema insicuro. Come visto, è possibile sfruttare tali attacchi per abilitare molteplici protocolli di comunicazione che possono rimanere attivi nello stesso momento.

In conclusione, si può affermare che le problematiche relative alla sicurezza delle comunicazioni in un ambiente SCADA dipendono non solo da configurazioni errate del sistema o da vulnerabilità specifiche del protocollo, ma anche da errori presenti all'interno del firmware che permettono di abilitare una comunicazione non confidenziale.

Acronimi

ADU Application Data Unit. 9, 11

CAN Controller Area Network. 13

CIP Common Industrial Protocol. 1, 9, 13, 14, 26

CRC Cyclic Redundancy Check. 10, 11

DoS Denial of Service. 16, 24, 29

HMI Human Machine Interface. 7

ICS Industrial Control System. 8, 9, 13, 16, 29

IDE Integrated Development Environment. 19, 24, 26

LRC Longitudinal Redundancy Checksum. 10

MAP Modbus Application Protocol. 11, 12

PDU Protocol Data Unit. 9

PLC Programmable Logic Controllers. 1–5, 8, 19, 20, 22–24, 26, 27, 29, 30

SCADA Supervisory Control and Data Acquisition. 2, 7, 9, 15, 16, 29

UDP User Datagram Protocol. 14

Termini speciali

Black Box Modello in cui il sistema in esame è una scatola nera, ovvero non è noto a priori né ciò che contiene né come si comporta. È possibile studiarne il comportamento esclusivamente analizzando le risposte che esso produce a fronte delle sollecitazioni che riceve. 21

DeviceNet DeviceNet è un protocollo di rete utilizzato nel settore dell'automazione per interconnettere i dispositivi di controllo per lo scambio di dati. 13

Ethernet/IP Protocollo di comunicazione che combina la tecnologia Ethernet tradizionale al protocollo industrial application layer, destinato all'automazione industriale. 13, 14, 26

Fieldbus Sistema di distribuzione supportato da una rete locale che garantisce la comunicazione tra i dispositivi collegati. 8

Forensic Analysis processo di rilevamento, investigazione e documentazione delle ragioni, processo e conseguenze di un attacco informatico o di una violazione contro leggi statali o di organizzazioni. 26

Insider threat Persona legata all'azienda che sfrutta i propri privilegi di accesso per compromettere sistemi e informazioni sensibili. 19

PCCC Programmable Controller Communication Commands - Protocollo di comunicazione proprietario Allen Bradley's. 26

Sniffing Attività di intercettazione passiva dei dati che transitano in una rete informatica: può essere svolta sia per scopi legittimi(analisi ed individuazione di problemi di comunicazione o di tentativi di intrusione), sia per scopi illeciti(intercettazione fraudolenta di password o altre informazioni sensibili). 16

Windows en-us_windows_11_business_editions_updated_jan_2022_x64. 19

Bibliografia

[1] automazione plc.it. Scada-network. [Online]. Available: https://automazione-plc.it/scada-comunicazione.html

[2] O. 10. Modbus rtu. [Online]. Available: https://ozeki.hu/p_5854-modbus-rtu.html

[3] S. Modbus. Modbus tcp. [Online]. Available: https://www.simplymodbus.ca/TCP.html

[4] A. Thiago. Scenari di attacco. [Online]. Available: https://media.defcon.org/DEF%20CON%2026/DEF%20CON%2026%20presentations/DEFCON-26-Thiago-Alves-Hacking-PLCs-and-Causing-Havoc-on-Critical-Infrastructures-Updated.pdf

[5] M. Alanazi, A. Mahmood, and M. J. M. Chowdhury, "Scada vulnerabilities and attacks: A review of the state-of-the-art and open issues," *Computers & Security*, vol. 125, p. 103028, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404822004205

[6] C.-W. Ten, C.-C. Liu, and G. Manimaran, "Vulnerability assessment of cybersecurity for scada systems," *IEEE Transactions on Power Systems*, vol. 23, no. 4, pp. 1836–1846, 2008.

[7] J. Axelson and N. Dierksheide, "Power Systems." *IEEE*, pp. 368–374, 1986.

[8] A. Daneels and W. Salter, "What is scada?" *International Conference on Accelerator and Large Experimental Physics Control Systems*, 1999.

[9] V. Schiffer, "Common industrial protocol (cip™) and the family of cip networks," in *Industrial Communication Technology Handbook*, 2nd ed. CRC Press, 2015, pp. 9–1–9–100.

[10] V. Schiffer, D. Vangompel, and R. Voss, "The common industrial protocol (cip) and the family of cip networks," *Milwaukee, Wisconsin, USA, ODVA*, 2006.

[11] P. Brooks, *Ethernet/IP-industrial protocol*. IEEE, 2001, vol. 2.

[12] P. Huitsing, R. Chandia, M. Papa, and S. Shenoi, "Attack taxonomies for the modbus protocols," *International Journal of Critical Infrastructure Protection*, vol. 1, pp. 37–44, 2008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S187454820800005X

[13] C. Company. enip-stack-detector. [Online]. Available: https://github.com/claroty/enip-stack-detector

[14] libplctag team. libplctag. [Online]. Available: https://github.com/libplctag/libplctag

[15] Senthivel, Saranyan, I. Ahmed, and V. Roussev, "SCADA network forensics of the PCCC protocol." *Digital Investigation*, pp. S57–S65, 2017.

[16] T. Alves. (2018) Thiagoralves/defcon26: Tools demonstrated at def con 26 talk "hacking plcs and causing havoc on critical infrastructures". [Online]. Available: https://github.com/thiagoralves/defcon26

[17] A. Thiago. (2020, Apr) Thiagoralves/ethersploit-ip: Exploiting allen-bradley e/ip plcs. [Online]. Available: https://github.com/thiagoralves/EtherSploit-IP