DoS su registro (fino a 2 persone, 2 punti)

INFO:

- Modbus permette R/W su coils e holding registers (vedi slides laboratorio);
- per DoS su registro si intende, una scrittura continua che renda difficoltoso l'utilizzo del registro e di conseguenza peggiori il lavoro della PLC, andando spesso ad intaccare anche il processo fisico controllato dalla PLC.

SPECIFICHE:

- Devo poter selezionare le PLC (tramite ip e porta) sul quale fare questo attacco; eventualmente per attaccare più PLC in contemporanea.
- Devo poter selezionare la coils o l'holding register sul quale voglio fare DoS, possono essere anche più di uno per tipo e combinazioni.
- Devo poter impostare una condizione (trigger) su una coil o più coil su quando far partire l'attacco e devo poter decidere se far partire manualmente l'attacco (Esempio, Se coil QX0.0 è ON allora l'attacco inizia, altrimenti aspetto),
- Devo scegliere se mandare un solo pacchetto e con quale valore al suo interno.

Chattering registro (fino a 2 persone, 2 punti)

INFO:

- Per chattering si intende la rapida successione di comandi alternati con il fine di danneggiare fisicamente un dispositivo fisico (ed esempio, una pompa) e peggiorare le performance del processo fisico nella sua interezza (ad esempio, allungando il ciclo di svuotamento di una vasca).
- Il chattering viene eseguito leggendo il valore in un dato momento di un registro ed inviando il valore alternati per tot secondi (Ad esempio, si tiene a 1 un coils per un secondo e poi si mette a 0, e così via).

SPECIFICHE:

- Devo poter selezionare le PLC (tramite ip e porta) sul quale fare questo attacco, possono essere anche più PLC in contemporanea.
- Devo poter selezionare la coil o l'holding register sul quale voglio fare DoS, possono essere anche più di uno per tipo e combinazioni.
- Devo poter impostare una condizione su una coils o più coisl su quando far partire l'attacco e devo poter decidere se far partire manualmente l'attacco (ad esempio, se coil QX0.0 è a 1 allora l'attacco inizia, altrimenti aspetto)
- Devo definire il range dell'attacco, impostando una condizione su uno o più input register (ad esempio, il chattering parte se il livello riportato in IW0 è tra 50 e 60; l'attacco si disattiva quando si esce da questo intervallo).
- Devo poter impostare l'intervallo tra l'invio di due comandi successivi di chattering in secondi (ad esempio, ogni secondo, ogni due secondi, etc.)
- Devo poter regolare il tempo di svuotamento della vasca tramite un parametro percentuale, come allungare al 125%, 150%, 175%, 200%. (Esempio, 3 secondi aperta e 1 chiusa, rallenta lo svuotamento del 25%, 2 secondi aperta e 2 secondi chiusa, rallenta lo svuotamento del 50% ecc…)

MITM

INFO:

- Per la cattura dei valori dei registri verrà fornito uno script di esempio, basato su ray module (per threading e distribuzione del carico)
- Per arp spoofing è possibile utilizzare tool preesistenti come *arpfox* o *arpspoof* o simili
- Per Implementazione di un server modbus (https://readthedocs.org/projects/pymodbustcp/downloads/pdf/latest/ pag 16 e 17)

SPECIFICHE:

- Devo poter selezionare le PLC (tramite ip) sul quale fare questo attacco, possono essere anche più PLC in contemporanea
- L'arp poisoning deve avvenire in un thread separato in modo da poter permettere l'utilizzo concorrente del tool di attacco
- Devo poter impostare un tempo di durata di questo attacco
    - PARTE A (3/4 persone, 4 punti + lode)
        - Devo poter registrare l'evoluzione dei valori nei registri (coils, input, discrete input e holding) del sistema attaccato per un periodo di tempo adeguato (ad esempio, 5/10 minuti). Questo file verrà fornito dal dott. Lucchese.
        - Devo poter selezionare una condizione sotto la quale far partire la cattura dei valori dei registri, in modo da sincronizzare l'attacco MITM con l'evoluzione del sistema sotto attacco.
        - Devo poter creare uno slave modbus che a seguito dell'arp poisoning riceverà le richieste dall'HMI e dovrà rispondere con i dati dei valori catturati e riportati nel file di cattura.
        - Devo poter scegliere quando far partire lo slave modbus, se manualmente o se sotto le medesime condizioni in cui si trovava il sistema al momento della cattura.
    - PARTE B (fino a 3 persone, 3 punti + lode)
        - Devo poter impostare il forward dei pacchetti anche dopo che l'arp poisoning è avvenuto, in modo da non creare disruption nel sistema (ad esempio, in una rete dove host A comunica con host B, una volta che l'attaccante C inizia il processo di arp poisoning, i pacchetti diretti da A verso B ora andranno verso C. B non riceverà più alcun pacchetto, quindi deve essere previsto anche l'inoltro dei pacchetti ricevuti da C a seguito dell'attacco verso B)
        - Devo poter fare il drop dei pacchetti desiderati.
        - Devo poter modificare i pacchetti che hanno comandi write.
        - (https://www.kitploit.com/2018/08/polymorph-real-time-network-packet.html)

Info utili

LINK:
https://github.com/marcolucc/networkSecurityProject/
mandare una mail con mail utilizzata su github.com a marco.lucchese@univr.it per essere aggiunti

REGOLE:

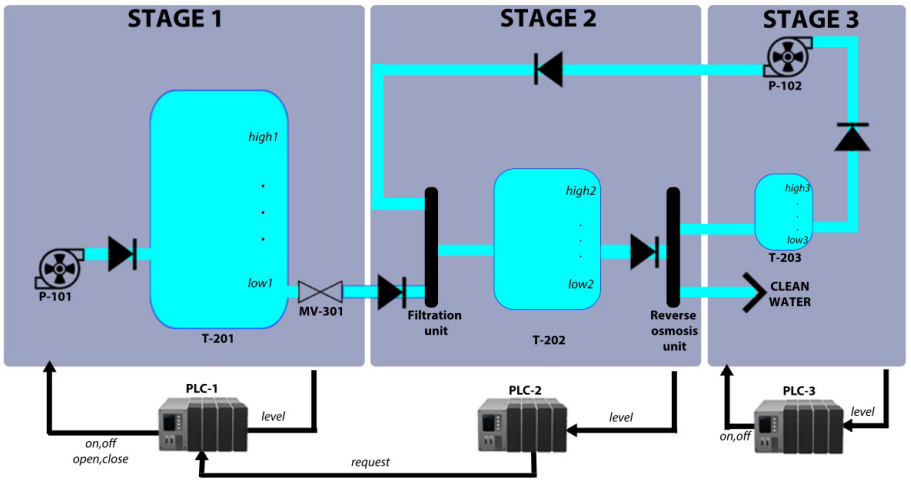
- Il progetto dovrà integrarsi con un sistema di base già creato.
- Il progetto deve essere scelto in anticipo e la scelta discussa coi docenti per essere sicuri della comprensione della specifica.
- I branch saranno utilizzati per separare lo sviluppo dei diversi progetti.
- Verrà fornito un branch di sviluppo per ogni progetto su github che partirà dal main.
- Assicurarsi di mantenere il branch di sviluppo aggiornato con le modifiche apportate al ramo principale. Prima di iniziare a lavorare, lanciare il comando "git pull origin main" per ottenere le ultime modifiche dal ramo principale.
- Integrazione con il main: Una volta completato lo sviluppo di una funzionalità o una modifica del branch di sviluppo, è necessario integrare le modifiche nel ramo principale. Si può fare ciò attraverso un processo di merge o richiedendo una pull request.
- Durante il processo di integrazione, potrebbero verificarsi conflitti tra i branch di sviluppo e il ramo principale. È importante risolvere questi conflitti in modo appropriato, prendendo in considerazione le modifiche apportate da entrambi i rami e assicurandosi che il codice finale sia funzionante e senza errori.
- Dopo l'integrazione, è necessario eseguire i test necessari per verificare che il codice funzioni correttamente nel contesto del sistema di base esistente. I test vanno eseguiti sia sul branch di sviluppo che sul ramo principale per garantire l'integrità del codice.

CERTIFICATI:
Per testare il tool verranno forniti dei certificati di accesso ad una VPN su una rete con 3 PLC ed 1 HMI (come descritta a laboratorio).
Per connettersi basta digitare i seguenti comandi:

```
sudo apt update
sudo apt install openvpn
sudo openvpn certificato_scaricato.ovpn
```

(Verrà richiesto un nome utente ed una password, il nome utente sarà “userX” dove X è un numero indicato nel nome del certificato, la password sarà “password”)

CASE STUDY:



Sarà fornito il testbed sopra.
composto da 3 PLC su container e una HMI su container.

GUIDA:
Creare un account su GitHub: Accedere al sito web di GitHub (https://github.com/) e creare un account gratuito. Verificare l'email per completare la registrazione.

Creare un nuovo repository: Fare clic sul pulsante "New" nella pagina principale di GitHub per creare un nuovo repository. Assegnare un nome al repository e selezionare le opzioni desiderate.

Clonare il repository: Per lavorare localmente sul proprio computer, clonare il repository sul sistema. Aprire il terminale e utilizzare il comando git clone seguito dall'URL del repository. Ad esempio:

    git clone https://github.com/tuo-username/nome-repository.git

Creare un nuovo branch: Per lavorare su una nuova funzionalità o una modifica senza influire sul branch principale, creare un nuovo branch. Utilizzare il comando git branch seguito dal nome del nuovo branch. Ad esempio:

    git branch nome-branch

Passare al nuovo branch: Per iniziare a lavorare sul nuovo branch, passare ad esso utilizzando il comando git checkout seguito dal nome del branch. Ad esempio:

    git checkout nome-branch

Effettuare le modifiche: Aprire l'editor di codice preferito e apportare le modifiche necessarie ai file del progetto presenti nella cartella clonata.

Aggiungere e committare le modifiche: Dopo aver apportato le modifiche, aggiungere i file modificati al repository utilizzando il comando git add seguito dal nome del file o utilizzando

git add . per aggiungere tutti i file modificati. Quindi, eseguire il commit delle modifiche utilizzando il comando git commit seguito da un messaggio di commit descrittivo. Ad esempio:

```
git add .
git commit -m "Aggiunte modifiche al file X"
```

Sincronizzare con il repository remoto: Per inviare le modifiche al repository remoto su GitHub, utilizzare il comando git push seguito dal nome del branch. Ad esempio:

```
git push origin nome-branch
```

Creare una pull request: Una volta caricato il branch con le modifiche sul repository remoto, è possibile creare una pull request. Accedere alla pagina del repository su GitHub e fare clic sul pulsante "New pull request". Selezionare il branch di origine (il proprio branch) e il branch di destinazione (solitamente il branch principale). Fornire una descrizione della pull request e inviarla.

Revisione e merge: Gli altri collaboratori del progetto possono quindi esaminare la pull request, commentare le modifiche e discuterne. Una volta approvata, un amministratore del repository può eseguire il merge della pull request nel branch principale, integrando così le modifiche nel progetto.