UNIVERSIDADE FEDERAL DE SANTA CATARINA
CAMPUS ARARANGUÁ

CLARA MARCELA GROSSL

Simulação de Ataques DoS e Replay com PIC16F877A

Araranguá/SC
2025

RESUMO

O avanço dos sistemas embarcados automotivos trouxe à tona preocupações críticas sobre a segurança dos barramentos de comunicação, especificamente o protocolo CAN (*Controller Area Network*). Este trabalho apresenta o desenvolvimento de uma plataforma de simulação de ciberataques e defesas utilizando microcontroladores PIC16F877A. O sistema emula uma ECU (*Electronic Control Unit*) e um dispositivo atacante (*Fuzzer*) comunicando-se via protocolo serial. Foram implementados e validados ataques de *Sniffing* (interceptação de dados), *Replay Attack* (clonagem de credenciais) e *Denial of Service* (DoS) por inundação de buffer. Como contramedida, desenvolveu-se um mecanismo de recuperação automática (*Self-Healing*) baseado em *Watch Dog Timer* (WDT) e monitoramento de integridade por interrupções.

Palavras-chave: Cibersegurança Automotiva. Sistemas Embarcados. UART. DoS.

Lista de Figuras

Sumário

1 INTRODUÇÃO

A arquitetura elétrica dos veículos modernos evoluiu de sistemas puramente mecânicos para redes complexas de processamento distribuído. Uma ECU (*Electronic Control Unit*) moderna gerencia desde a injeção eletrônica até o destravamento das portas. O padrão de comunicação dominante, o protocolo CAN, foi projetado na década de 1980 com foco em robustez elétrica, mas sem mecanismos nativos de segurança cibernética, como criptografia ou autenticação de pics.

A ausência de segurança no nível do protocolo permite que um agente malicioso, com acesso físico ao barramento (via porta OBD-II), injete dados arbitrários, podendo assumir o controle do veículo. Este trabalho visa explorar essas vulnerabilidades em um ambiente controlado, desenvolvendo ferramentas de ataque e estratégias de defesa em hardware.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

Projetar e implementar um sistema embarcado modular composto por um pic atacante e um pic vítima, capaz de simular a injeção de falhas em protocolos de comunicação serial e validar mecanismos de recuperação de sistema.

1.1.2 Objetivos Específicos

- Desenvolver um protocolo sobre UART para emular o tráfego de dados.
- Implementar rotinas de *Sniffing* com armazenamento persistente em EEPROM.
- Demonstrar a saturação de buffer e travamento de CPU através de ataques de DoS.
- Utilizar *Watch Dog Timer* (WDT) e Interrupções de Timer para garantir a confiabilidade e o reinício automático do sistema sob falha.

2 EMBASAMENTO TEÓRICO

2.1 VULNERABILIDADES DE PROTOCOLO

- **Replay Attack:** Consiste na captura de um sinal válido (ex: comando de destravamento) e sua retransmissão posterior. Como o protocolo não possui *timestamp* ou *nonce*, a vítima aceita o comando clonado como legítimo.
- **Denial of Service (DoS):** Explora a limitação de processamento da vítima. Ao inundar o barramento com dados de alta prioridade ou lixo (`0x00`), o buffer de recepção estoura (*Overrun Error*), levando a CPU a um estado de falha.

2.2 WATCH DOG TIMER (WDT)

O WDT é um contador de hardware independente que reinicia o processador caso o software principal pare de responder (trave). Em sistemas críticos, como automotivos, o WDT é obrigatório para garantir que o veículo não permaneça em um estado inoperante após uma falha transiente ou ataque.

3 IMPLEMENTAÇÃO PRÁTICA E HARDWARE

O sistema foi desenvolvido utilizando dois microcontroladores PIC16F877A operando a 20MHz, comunicando-se via interface serial assíncrona (UART) a 9600 bps.

3.1 ARQUITETURA DO SISTEMA

A topologia consiste em dois pics distintos:

1. **Pic Vítima (ECU de Chassi):** Responsável pelo controle de acesso, monitoramento de velocidade (ADC) e interface com o usuário (LCD/LEDs).
2. **Pic Atacante (Fuzzer):** Dispositivo de auditoria capaz de ler o barramento, gravar dados na EEPROM e injetar pacotes maliciosos.

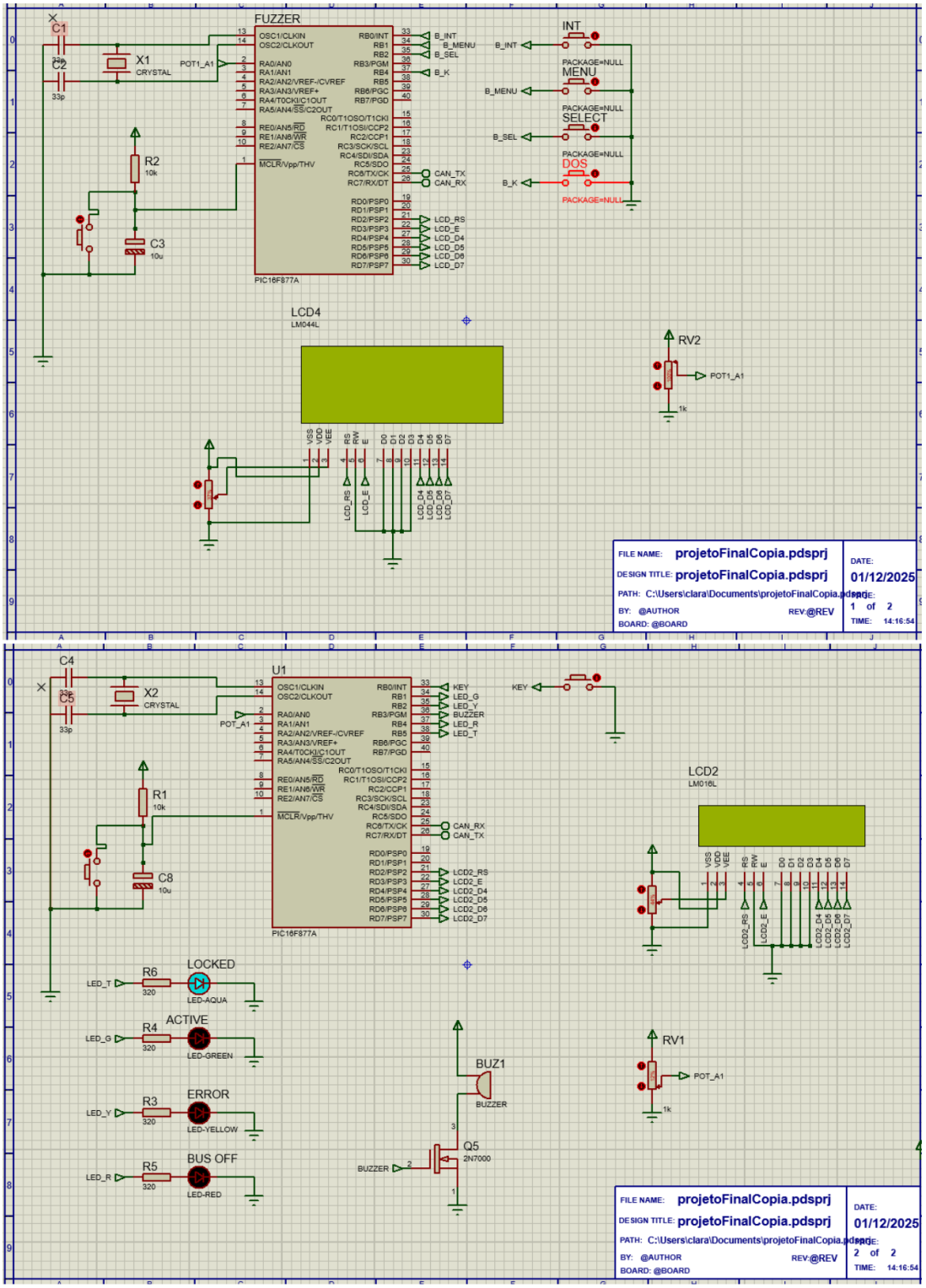

Figura 1: Esquemático do sistema: Atacantene Vítima interligados via TX/RX.

3.2 DETALHES DO FIRMWARE

3.2.1 Mecanismo de Defesa (Vítima)

A Vítima implementa uma rotina de verificação de integridade.

- **Timer1:** Gera interrupções periódicas para telemetria, independente do loop principal.
- **WDT:** Configurado com *prescaler* 1:128 (aprox. 2.3s). O código alimenta o WDT com `CLRWDT()`. Se o ataque DoS prender o processador em um loop infinito, o WDT estoura e reinicia o sistema.

3.2.2 Lógica de Ataque (Fuzzer)

O pic atacante utiliza a EEPROM interna para persistir dados.

- **Sniffer:** Lê o barramento RX. Ao detectar o cabeçalho `#`, armazena os bytes subsequentes na EEPROM.
- **Kill (DoS):** Utiliza um potenciômetro para definir a intensidade do ataque. Acima de 95% de carga, o sistema envia bytes nulos (`0x00`) continuamente, forçando um erro de *Framing* ou *Overrun* na vítima.

4 ANÁLISE DE RESULTADOS

Os testes realizados em simulação confirmaram as vulnerabilidades e a eficácia das defesas.

4.1 CENÁRIO DE SNIFFING E REPLAY

Ao acionar a chave original, o pacote foi transmitido. O Atacante, em modo *Sniffer*, capturou e gravou a sequência na EEPROM. Posteriormente, ao acionar o modo *Replay*, o Atacante enviou a cópia do sinal, e a Vítima destravou as portas (LED Verde), validando a falha de autenticação.

4.2 CENÁRIO DE ESTRESSE E RECUPERAÇÃO (DoS)

Submeteu-se a vítima a cargas crescentes de dados inúteis.

1. **Carga Média (Amarelo):** O sistema apresentou latência, mas continuou operando.
2. **Carga Alta (Vermelho):** Ao configurar o potenciômetro atacante entre 80 e 95 porcento, é simulado um ataque do qual o WDT consegue reiniciar a vítima, trazendo o sistema de volta ao estado operacional seguro.
3. **Carga Fatal (Vermelho):** Ao injetar `0x00` continuamente, a vítima entrou em estado de travamento contínuo (*Deadlock*).

5 CONCLUSÃO

O projeto atingiu os requisitos propostos ao implementar um sistema complexo de comunicação serial bidirecional com tratamento de falhas.

REFERÊNCIAS

MICROCHIP TECHNOLOGY. **PIC16F87XA Data Sheet**. 2003.

RAGHAVAN, Siranjeevi S. **CAN Bus Security: The Unseen Cybersecurity Battle in Connected Vehicles**. TechLatest, 2025.

EUROSENS. **How to Decode Vehicle's CAN Bus Data**. Instructables, 2025.