ERENO-UI: Uma Ferramenta para a Geração de Conjuntos de Dados para a Detecção de Intrusão em Redes IEC–61850

Silvio E. Quincozes[1], Vagner E. Quincozes[2], Célio Albuquerque[3], Diego Passos[3,4], Daniel Mossé[5]

[1] Universidade Federal de Uberlândia - UFU

[2]Universidade Federal do Pampa - UNIPAMPA

[3]Universidade Federal Fluminense - UFF

[4]Instituto Superior de Engenharia de Lisboa - ISEL

[5]Universidade de Pittsburgh - PITT

sequincozes@gmail.com, vagnerquincozes.aluno@unipampa.edu.br,

celio@ic.uff.br, dpassos@ic.uff.br, mosse@pitt.edu

Resumo. *Os protocolos de comunicação propostos pelo padrão IEC–61850 ampliam a conectividade em subestações elétricas digitais. Contudo, novas vulnerabilidades também são introduzidas. Assim, Sistemas de Detecção de Intrusão (IDSs) tornam-se essenciais para proteger as subestações. No entanto, o emprego de IDSs nesses cenários é limitado pela falta de assinaturas de ataques. Neste trabalho é proposto o ERENO-UI: um componente baseado em uma interface gráfica amigável que funciona de maneira integrada com o gerador de datasets ERENO. Os resultados demonstram que o ERENO-UI permite a configuração apropriada do ERENO para a reprodução de mensagens realistas de protocolos GOOSE e SV, consistentes com os padrões IEC–61850.*

1. Introdução

A norma IEC–61850 [Commission 2003] introduziu novos protocolos de comunicação para que dispositivos eletrônicos inteligentes, do inglês *Intelligent Electronic Devices* (IEDs), possam trocar mensagens críticas em redes de subestações digitais que transmitem e distribuem energia elétrica. Dentre as padronizações que tal norma prevê, incluem-se os protocolos *Sampled Values* (SV), utilizado para transmitir amostragens de valores de corrente e tensão dos equipamentos elétricos para IEDs, e *Generic Object Oriented Substation Event* (GOOSE), adotado para a transmissão de eventos críticos entre IEDs.

Em aplicações críticas, a disponibilidade, bem como o atendimento de requisitos de tempo real, muitas vezes se sobrepõem aos outros requisitos de segurança da informação, tal como a autenticidade. De fato, em subestações baseadas na norma IEC-61850 há uma grande preocupação quanto à entrega de mensagens para o funcionamento devido dos sistemas, tais como aquelas que executam funções de proteção (*i.e.*, enviando comandos para a manipulação de disjuntores e isolamento de faltas elétricas). Por outro lado, tais mensagens não contemplam requisitos como a autenticidade, confidencialidade e integridade [Hong and Liu 2019]. Embora as redes IEC–61850 sejam muitas vezes segmentadas e fisicamente isoladas, atacantes ainda podem explorar vulnerabilidades através de acesso remoto por engenheiros responsáveis pela manutenção do sistema elétrico ou ataques internos que podem ocorrer através de *malwares* que são transmitidos por *pendrives* ou atualizações em *softwares* de fabricantes de dispositivos elétricos. Por esse

motivo, a implementação de Sistemas de Detecção de Intrusões, do inglês *Intrusion Detection Systems* (IDSs) [Quincozes et al. 2021], é fundamental para a segurança das redes e dispositivos baseados na norma IEC–61850.

Neste trabalho é proposta a ferramenta ERENO-UI: uma aplicação *web* com uma interface gráfica amigável para a configuração e acesso às saídas da ferramenta ERENO [Quincozes et al. 2022], a qual permite a reprodução de mensagens GOOSE e SV realistas e consistentes com os padrões IEC–61850 [Commission 2003]. Essas mensagens podem simular situações normais (*i.e.,* condições estáveis e/ou faltas elétricas típicas) e situações de ataques, podendo, portanto, constituir um *dataset* realístico para a avaliação de IDSs em subestações empregando o padrão IEC–61850.

Em síntese, este trabalho apresenta as seguintes contribuições técnicas:

1. Especificação de uma aplicação cliente para a facilitar a interação do usuário com a ferramenta ERENO;
2. Definição de uma Interface de Programação de Aplicação (API) com *end-points* que permitem o acesso às funcionalidades da ferramenta ERENO via Web;
3. Implementação da aplicação proposta na forma de uma aplicação para a Web denominada ERENO-UI;
4. Demonstração prática da geração de um *dataset* para a detecção de intrusões através da ferramenta proposta.

Tanto o código-fonte da aplicação proposta e implementada quanto o conjunto de dados gerados são disponibilizados publicamente. Nas próximas seções, a arquitetura da ferramenta e sua implementação são apresentadas (Seções 2 e 3, respectivamente). Em seguida, as Seções 4 e 5 apresentam uma prova de conceito e considerações finais.

2. Arquitetura

A Figura 1 ilustra os componentes da arquitetura proposta. Os componentes da camada de visualização, em azul, são novos componentes propostos neste trabalho. Os componentes da Camada de Processamento, em laranja, são baseados na proposta original do ERENO [Quincozes et al. 2022]. No entanto, tais componentes foram modificados arquiteturalmente de modo a operar de forma distribuída por meio de microsserviços. Por fim, um novo componente é introduzido na Camada de Armazenamento. A seguir, todos os componentes são detalhados.

2.1. Camada de Visualização

A camada de visualização é composta por quatro módulos:

1. **Configurações.** Esse módulo é responsável pela configuração de parâmetros que permitem que o gerador ERENO obtenha dados compatíveis com fluxos de mensagens GOOSE de IEDs de uma subestação real. Dentre esses parâmetros, ressalta-se a periodicidade entre mensagens, os endereços MAC de origem e destino, o identificador de aplicação ($GOOSEAppID$), uma *tag* $TPID$ (de acordo com o padrão IEC–61850, usa-se 0x8100 para o GOOSE), o *GOOSE Control Block Reference name* ($gocbRef$), entre outras informações de IEDs ou de subestações.
2. **Ataques.** É um módulo extensível, que permite a modelagem de ataques, incluindo a definição de novos modelos de diferentes tipos e classes. Por exemplo, podem ser

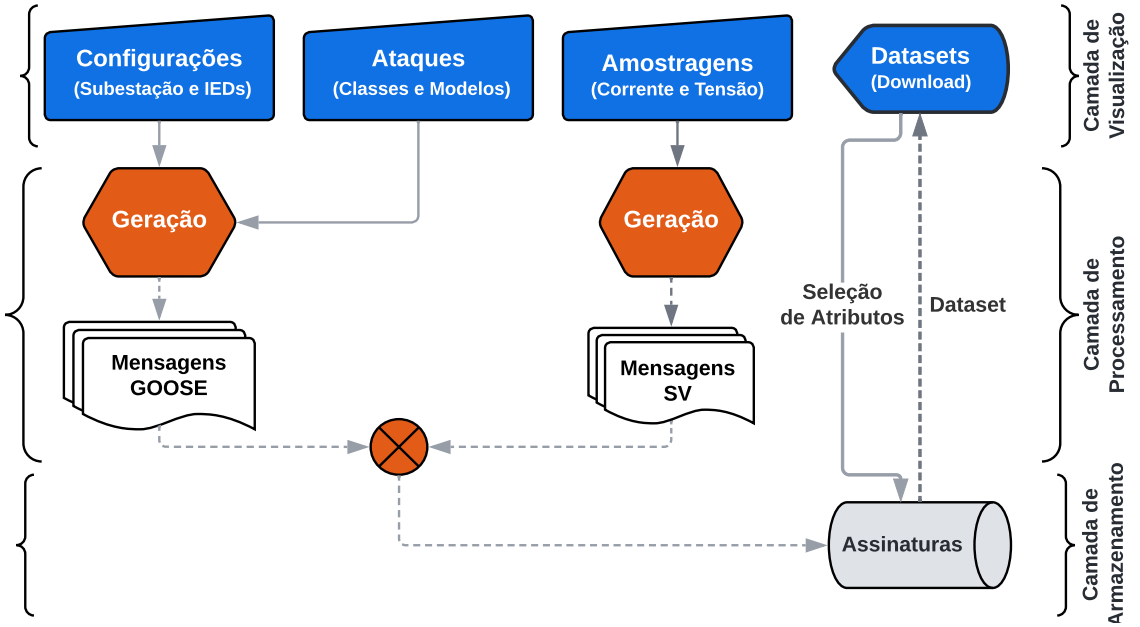


Figura 1. A arquitetura proposta com integração do ERENO-UI com o ERENO. Os componentes do ERENO-UI residem na camada de visualização, novas contribuições deste trabalho, e são destacados em azul.

implementados ataques de retransmissão e injeção de mensagens, negação de serviço, mascaramento, entre outros. Em síntese, é possível estender o ERENO para qualquer tipo de ataque, pois o mesmo adota o padrão de projeto *Factory Method*, que permite que um dispositivo atacante modifique o comportamento de um IED legitimo.

3. **Amostragens.** Tal módulo é responsável por receber amostras de corrente e tensão, as quais podem ser obtidas por meio de um simulador ou de dispositivos reais. Os dados são enviados ao ERENO por meio de *upload* na ERENO-UI deste módulo. Vale ressaltar que o ERENO não gera/simula sinais elétricos, apenas fornece uma interface para que um usuário consiga informar seus próprios dados. Na Seção 4, será disponibilizado um arquivo contendo um conjunto de amostras que pode ser usados por padrão para esta finalidade, caso o usuário da ferramenta não disponha de tal artefato.
4. **Download de Datasets.** O módulo de *download* de *datasets* fornece o acesso aos conjuntos de dados que foram gerados pelos componentes da Camada de Processamento. Tal geração acontece com base nas especificações de configurações, ataques e amostragens de sinais elétricos que foram informados na camada de visualização. No momento do *download*, o usuário pode definir quais são as *features* (*i.e.,* colunas do *dataset*) que farão parte do arquivo a ser disponibilizado.

2.2. Camadas de Processamento e Armazenamento

As funcionalidades dos componentes da Camada de Processamento são as mesmas propostas na versão original da ferramenta ERENO [Quincozes et al. 2022], a qual não possui suporte à comunicação com outras aplicações. Portanto, de modo a permitir a integração com o ERENO-UI proposto neste trabalho, os componentes originais do ERENO são modificados arquiteturalmente. Tal modificação consiste na extensão do ERENO de modo a permitir que o mesmo opere de forma distribuída por meio de microsserviços. Assim, é possível que as ferramentas operem de maneira desacoplada

e suas funcionalidades sejam acessadas tanto local quanto remotamente (*i.e.,* via Internet).

Na Figura 2, é exibido o diagrama de classes referente à Camada de Processamento do ERENO [Quincozes et al. 2022]. De modo a permitir a criação de novas classes de dispositivos infectados, o padrão de projeto *Factory Method* [Gamma 2009] foi adotado. Tal padrão criacional baseia-se em uma classe abstrata que usa uma interface para padronizar a criação de objetos concretos. No contexto da ferramenta proposta, tal classe abstrata representa um IED que usa a interface `MessageCreator` para a criação de novas mensagens. As classes concretas `MergingUnit` e `ProtectionIED` herdam as funcionalidades e atributos da classe IED e têm acesso à geração de mensagens por meio do método *run*. O tipo de mensagem a ser gerado depende do tipo do objeto que estende a classe IED. As cores usadas na Figura 2 ilustram as correspondências de dispositivos, *end-points* e mensagens. Em amarelo, ilustra-se a classe concreta `MergingUnit`, que é um IED que usa a implementação `SVCreator` para a geração de mensagens SV. Por outro lado, em azul, o IED de proteção (classe concreta `ProtectionIED`) usa a implementação `GOOSECreator` para a geração de mensagens GOOSE.

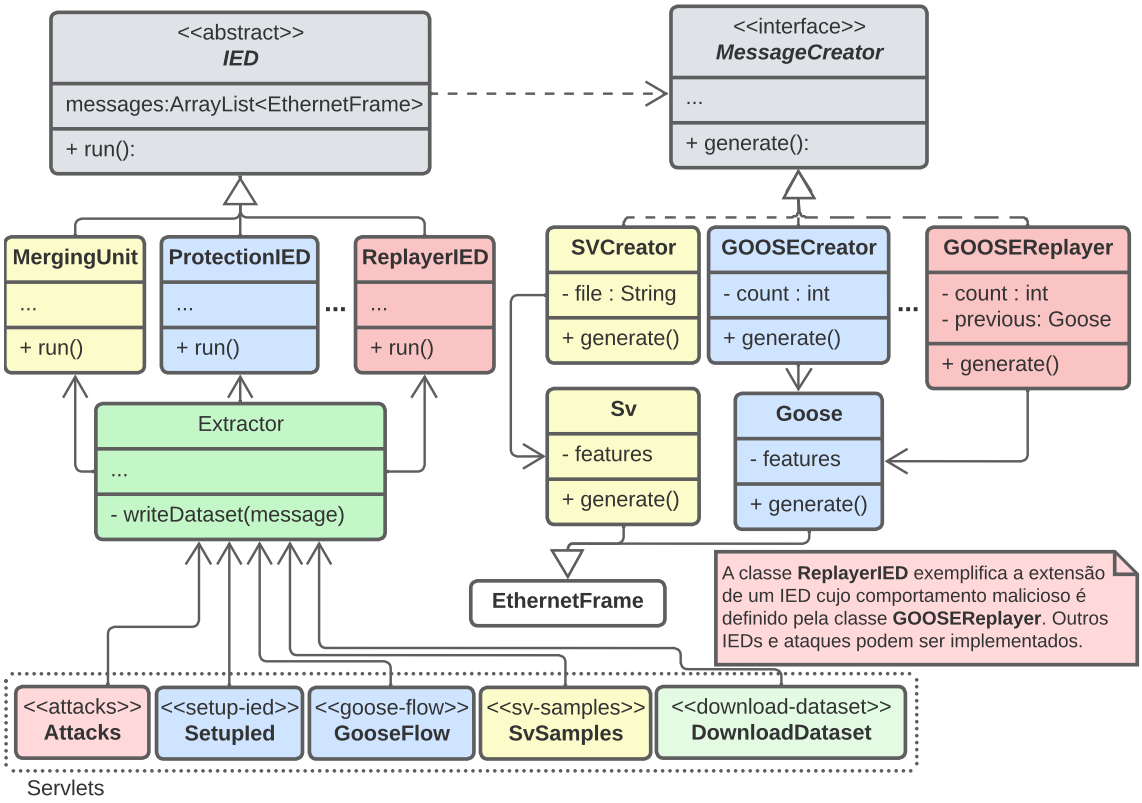


Figura 2. Diagrama de classes da Camada de Processamento.

Além das classes que representam o comportamento típico de IEDs de uma subestação (*i.e.,* `ProtectionIED` e `MergingUnit`), novas classes, com novos comportamentos, podem ser implementados usando o mesmo padrão de projeto. No diagrama, é exemplificada na cor vemelho a definição de um IED infectado com um *malware* (*ReplayerIED*) que executa ataques de reprodução de mensagens. O padrão comportamental do atacante é definido no método *run* dessa classe, o qual usa o método *generate* da

implementação `GOOSEReplayer` para reproduzir mensagens de dispositivos legítimos.

A classe *Extractor* é a responsável por inicializar, executar e extrair as informações relevantes de cada IED abstrato (*e.g., MergingUnit*, *ProtectionIED*, *ReplayerIED*, etc.). Tal classe possui um método *main*, portanto, ela pode ser executada diretamente no ambiente de desenvolvimento integrado, do inglês, *Integrated Development Environment* (IDE). Alternativamente, seus parâmetros e definições podem ser alterados e acessados por aplicações terceiras que interagem com os *servlets* exibidos na Figura 2. Tais *servlets* podem ser acessados remotamente por qualquer dispositivo conectado à Internet. Todos os conjuntos de dados gerados pela ferramenta são armazenados na *Camada de Armazenamento*, a qual consiste de um dispositivo como o disco rígido local ou uma unidade de armazenamento externa em nuvem.

3. Implementação

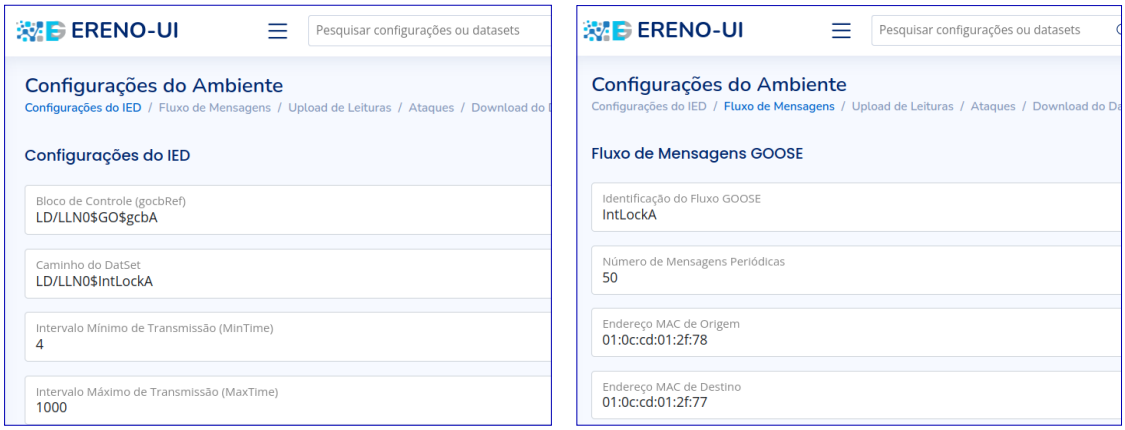
A implementação do ERENO-UI é voltada para qualquer pessoa interessada em produzir seu próprio *dataset* composto pelos protocolos GOOSE e SV, do padrão IEC-61850. Os principais objetivos dessa implementação consistem em: (a) permitir, através de interfaces visuais, que usuários comuns consigam parametrizar o ERENO e acessar os dados com facilidade, e (b) disponibilizar a API proposta para programadores de outras aplicações, o que permitirá integrações com *softwares* de terceiros, fornecendo novos parâmetros, incluindo detalhamentos de ataques, configurações de IEDs, definições de fluxos de mensagens e amostragens de corrente e tensão típicas do cenário dessas aplicações.

Na implementação do ERENO-UI, adotou-se o conceito de *Servlet* para estender funções da ferramenta ERENO [Quincozes et al. 2022]. Em síntese, um *Servlet* consiste em uma classe na linguagem de programação *Java* que utiliza a *API Java Servlet* [Farley et al. 2006] para receber, processar e responder requisições de clientes que são realizadas através do protocolo *Hypertext Transfer Protocol* (HTTP). Dessa forma, *servlets* são normalmente representados como objetos dessa classe. A partir do momento que tal objeto recebe uma mensagem do tipo *request*, este processa a requisição e responde com uma mensagem do tipo *response*. Uma mensagem de resposta pode ser, por exemplo, uma página *web* dinâmica ou qualquer outro tipo de arquivo, como imagens, textos, etc. Com isso, uma possível resposta para uma requisição GET consiste no conjunto de dados gerados com base nos parâmetros enviados ao ERENO. A seguir, as interfaces gráficas e APIs de programação serão discutidas.

3.1. Interfaces Visuais (*Front-End*)

Nesta seção, algumas das interfaces visuais do ERENO-UI são apresentadas. Em síntese, para cada componente proposto na arquitetura implementou-se uma interface visual. A Figura 3(a) ilustra a primeira etapa do sistema, a qual permite configurar parâmetros do IED. Por exemplo, o usuário pode configurar blocos de controle, caminho do *datSet*[1], dentro do IED, além de vários outros parâmetros não demonstrados no texto por restrição de espaço. Após configurar os parâmetros, a segunda etapa (Figura 3(b)) permite controlar o fluxo de mensagens GOOSE, incluindo o número de mensagens que serão geradas.

[1]O *datSet* é uma estrutura de dados interna de mensagens GOOSE e não deve ser confundido com o termo *dataset*, que corresponde ao arquivo de amostras gerada pelo ERENO.

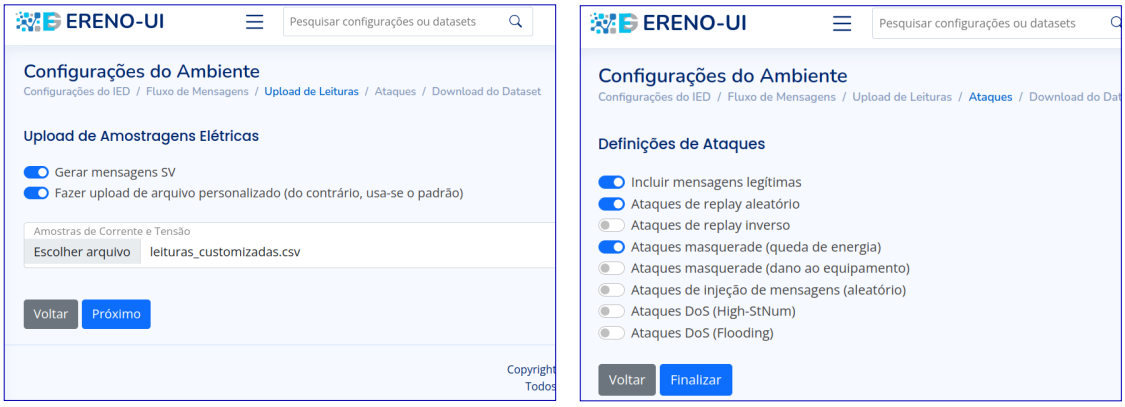


(a) Configuração de um IED. (b) Configuração do Fluxo de Mensagens GOOSE.

Figura 3. Interfaces gráficas para o módulo de configuração.

A terceira etapa, ilustrada na Figura 4(a), é opcional e deve ser usada apenas caso o usuário opte pela geração de mensagens SV correspondentes às mensagens GOOSE (mais detalhes em [Quincozes et al. 2021]). Para habilitar a geração de mensagens SV, o usuário pode optar por utilizar o conjunto de amostras disponibilizado pelos autores do ERENO-UI ou fazer o *upload* de amostras de corrente e tensão obtidas a partir de seus próprios dispositivos elétricos, por meio de um arquivo no formato *Comma Separated Values* (CSV). Essa etapa é importante para permitir a geração de mensagens SV realistas.

Na Figura 4(b), a última etapa de configuração é apresentada. Nela, é possível definir ataques, como os de *replay* aleatório e inverso, ataques DoS, dentre outros — os ataques disponíveis nesta interface são os definidos em [Quincozes et al. 2022]. Por questões de usabilidade, foram adotados botões do tipo *toggle switch*. Assim, o usuário pode selecionar de maneira simplificada os ataques para a geração do dataset. Após clicar em finalizar, o *dataset* resultante é disponibilizado para *download* no formato *Attribute-Relation File Format* (ARFF), cuja estrutura é exibida na Seção 4.



(a) *Upload* de amostras de corrente e tensão (para mensagens SV). (b) Seleção de modelos de ataques.

Figura 4. Interfaces gráficas para o módulo de ataques.

3.2. Interfaces de Programação de Aplicações (APIs)

De modo a permitir a configuração de IEDs por aplicações terceiras, é proposto o conceito de *ERENO Configuration File* (ECF). Os ECF consistem em arquivos de configurações no formato JSON que podem ser utilizados para a definição dos mesmos parâmetros que podem ser personalizados através da aplicação ERENO-UI. O *upload* do arquivo ECF deve utilizar uma requisição HTTP do tipo `POST`. Para obter-se uma versão do arquivo ECF atualmente configurado, pode-se fazer uma requisição tipo `GET`. Existem quatro *end-points* aos quais pode-se utilizar o padrão ECF para a configuração do ERENO. O propósito de cada um deles é descrito a seguir:

- `setup-ied`: configurações do IED (correspondente à etapa 1 do ERENO-UI);
- `goose-flow`: fluxos GOOSE (correspondente à etapa 2 do ERENO-UI);
- `sv-samples`: sinais elétricos (correspondente à etapa 3 do ERENO-UI);
- `attacks`: seleção de ataques (correspondente à etapa 4 do ERENO-UI);

Por fim, o *download* do *dataset* resultante das configurações definidas via tais *end-points* pode ser acessado pelo *end-point* descrito a seguir:

- `download-dataset`: retorna o *dataset* resultante das definições realizadas.

4. Prova de Conceito e Demonstração da Aplicação

De modo a executar a aplicação ERENO-UI, o usuário deve acessar e clonar o repositório do código-fonte[2] que contém tanto o *front-end* quanto o *back-end* do projeto. Para a execução do projeto, se faz necessário um computador com um servidor de aplicações Web com suporte à linguagem de programação Java. Recomenda-se o Apache Tomcat, versão 9.0.65[3], o qual já foi validado durante os testes internos da ferramenta.

Uma vez que o ERENO-UI encontra-se em execução, sua interface *web* pode ser acessada através do endereço e porta correspondente ao servidor *web* instalado (tipicamente o acesso local se dá pela URL `https://localhost:8080`). Alternativamente, pode-se configurar o ERENO através de requisições do tipo `POST` para os *end-points* listados na Seção 3.2. Para a obtenção do *dataset* resultante, pode-se enviar uma requisição do tipo `GET` para a URL `https://localhost:8080/dataset`. Mais informações sobre a instalação e execução estão disponíveis no repositório on-line. A demonstração do sistema ERENO-UI é organizada em cinco etapas:

1. Configuração do IED (valores padrões são sugeridos pela ferramenta);
2. Configuração do fluxo de mensagens GOOSE (valores padrões são sugeridos);
3. Carregamento de amostragens de corrente e tensão para mensagens SV (será disponibilizado um arquivo contendo tais amostras);
4. Seleção dos ataques que serão incluídos no *dataset*;
5. Download do *dataset*. Um trecho do *dataset* resultante do processo de geração através do ERENO-UI é exibido na Figura 5.

Para a geração de um dataset contendo 100 mil mensagens GOOSE (90 mil mensagens legítimas e 10 mil mensagens provindas de um IED malicioso, que executa ataques de repetição) são consumidos 754 *ms* (*i.e.,* são geradas aproximadamente 132.625 mensagens por segundo). O arquivo resultante contém 21,8 *MiB*.

[2]Disponível em: `https://github.com/sequincozes/ereno`
[3]Disponível em: `https://tomcat.apache.org/download-90.cgi`

```
@attribute sqDiff numeric
@attribute gooseLengthDiff numeric
@attribute cbStatusDiff numeric
@attribute apduSizeDiff numeric
@attribute frameLengthDiff numeric
@attribute timestampDiff numeric
@attribute tDiff numeric
@attribute timeFromLastChange numeric
@attribute delay numeric
@attribute @class@ {normal, random_replay, inverse_replay, masquerade_fake_fault, masquera
@data
3635.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
3635.0002,2.2188095E-7,7.6845135E-9,-2.300546E-7,2.4746007E-5,2.0418714E-5,-4.632196E-5,3.
3635.0005,4.3133646E-6,-1.7631729E-6,-2.4526248E-6,9.191107E-5,8.91385E-5,-1.8671055E-4,6.
3635.0007,2.9725237E-5,-1.39711665E-5,-1.4950079E-5,1.8915212E-4,2.0208803E-4,-4.051077E-4
3635.0007,1.0609898E-4,-4.625997E-5,-5.7189714E-5,3.046762E-4,3.707424E-4,-6.9988327E-4,1.
3635.001,2.2267374E-4,-1.11844754E-4,-1.04561834E-4,4.3682393E-4,6.153771E-4,-0.0010883508
3635.0012,3.937294E-4,-2.3461298E-4,-1.4699862E-4,5.8955746E-4,9.610491E-4,-0.0015944289,8
3635.0015,6.6561904E-4,-4.14338E-4,-2.3027796E-4,7.354921E-4,0.001423612,-0.0022059851,1.4
3635.0017,0.0011136136,-6.568473E-4,-4.2141817E-4,8.101981E-4,0.00202061,-0.002877283,2.43
3635.002,0.0018022635,-9.5235556E-4,-7.9671835E-4,7.725269E-4,0.0027540552,-0.003571764,3.
```

Figura 5. Trecho do *dataset* gerado através da ferramenta ERENO-UI.

5. Considerações Finais

Neste trabalho, propõe-se o ERENO-UI: uma ferramenta gráfica que se integra ao gerador de *datasets* ERENO [Quincozes et al. 2022]. Tais *datasets* são compostos por mensagens dos protocolos GOOSE e SV. A interface gráfica proposta possibilita a configuração do ERENO (*e.g.*, definição de parâmetros e ataques, inserção de amostragens de corrente e tensão) de forma amigável, até mesmo para usuários comuns. A ferramenta é disponibilizada gratuitamente e permite o *download* do conjunto de dados resultante. Dentre as vantagens da ferramenta, inclui-se (i) a fácil parametrização da ferramenta ERENO por meio do ERENO-UI, (ii) a fácil integração com *softwares* de terceiros por meio da API disponibilizada para programadores e (iii) o fácil uso do *dataset* gerado, o qual se baseia em um formato usado por ferramentas clássicas de aprendizado de máquina (*i.e.,* ARFF).

Referências

Commission, I. E. (2003). *Communication networks and systems in substations*. IET.

Farley, J., Crawford, W., Malani, P., Norman, J. G., and Gehtland, J. (2006). *Java Enterprise in a Nutshell*. "O'Reilly Media, Inc.".

Gamma, E. (2009). *Padrões de Projetos: Soluções Reutilizáveis*. Bookman.

Hong, J. and Liu, C. (2019). Intelligent Electronic Devices With Collaborative Intrusion Detection Systems. *IEEE Transactions on Smart Grid*, 10(1):271–281.

Quincozes, S. E., Albuquerque, C., Passos, D., and Mossé, D. (2021). A Survey on Intrusion Detection and Prevention Systems in Digital Substations. *Computer Networks*, 184:107679.

Quincozes, S. E., Albuquerque, C. V. N., Passos, D., and Mossé, D. (2022). *ERENO: An Extensible Tool for Generating Realistic IEC–61850 Intrusion Detection Datasets*. Tese de doutorado, Universidade Federal Fluminense, Niterói. Disponível em `https://www.facom.ufu.br/~sequincozes/assets/tese.pdf`.