Universidade de Pernambuco
Escola Politécnica de Pernambuco
Programa de Pós-Graduação Acadêmica em Engenharia de Computação

RELATÓRIO FINAL

MODELAGEM CONCEITUAL E RACIOCÍNIO AUTOMÁTICO

Julio Cesar da Silva
Discente Doutorando
jcs2@ecomp.poli.br

Cleyton Mário de Oliveira Rodrigues
Prof.º Dr.
cleyton.rodrigues@upe.br

SUMÁRIO

ÍNDICE DE FIGURAS

1 CONTEXTUALIZAÇÃO

O mundo tem evoluído em termos de tecnologia e com isto também tem evoluído os ataques cibernéticos através da rede mundial de computadores. Vários países vêm atualizando suas leis civis e penais para tentar conter a crescente onda de ataques e roubo de dados. Conforme pode ser visualizado no relatório anual de segurança pública de 2024[1], desenvolvido pelo Fórum Brasileiro de Segurança Pública, os crimes contra o patrimônio, ao que se refere a Estelionato por meio eletrônico, tem aumentado em todos os estados do Brasil. Diante deste cenário, em 2021, o crime de Estelionato - Fraude eletrônica - passou a ser tipificado pelos parágrafos 2ºA, 2ºB e 3º do art. 171 do Código Penal. Desde de então, é possível acompanhar a estatística deste crime através dos dados enviados pelas Secretarias Estaduais de Segurança Pública e/ou Defesa Social dos estados, bem como pelas polícias civis.

2 PROBLEMA

Embora atualmente exista a possibilidade de os órgãos de segurança pública desenvolverem estatísticas baseadas na tipificação atualizada, a tarefa de se obter tais dados demonstrou-se não ser tão trivial. Em consultas anteriores a tais órgãos, constatou-se que há desorganização na estruturação dos dados destes órgãos, bem como a ausência de pessoal capacitado para responder aos questionamentos realizados pelos pesquisadores.

3 OBJETIVO

Diante do problema elencado, foi desenvolvido uma Ontologia para servir de fonte de conhecimento e listar as técnicas de ataques utilizadas por infratores a partir de dispositivos eletrônicos (*smpartphones*, computadores, *tablets*, etc.) e suas associações entre os Artigos do Código Penal para servir de referência para futuras pesquisas, além de auxiliar os agentes de segurança pública na divulgação de informações acerca das infrações que se utilizam de técnicas de engenhosidade social, mais especificamente do *Phishing*.

4 PROPOSTA

De posse desse conhecimento, foi sugerido o desenvolvimento de uma Arquitetura de Sistemas Multiagentes capaz de subsidiar os órgãos públicos com dados estruturados e organizados, a partir do mapeamento e associações de um determinado domínio de conhecimento. Com o mapeamento adequado, espera-se fornecer subsídios aos órgãos de segurança pública para decisões mais eficientes em nível nacional. Ademais, busca-se estabelecer uma base de dados nacional abrangente, reunindo

[1] https://forumseguranca.org.br/wp-content/uploads/2024/07/anuario-2024.pdf

informações sobre potenciais ataques cibernéticos e as técnicas empregadas em sua execução.

Com a intenção didática, apresenta-se na Figura 1 um Esquema de Proposta Arquitetural representado por Ontologia.

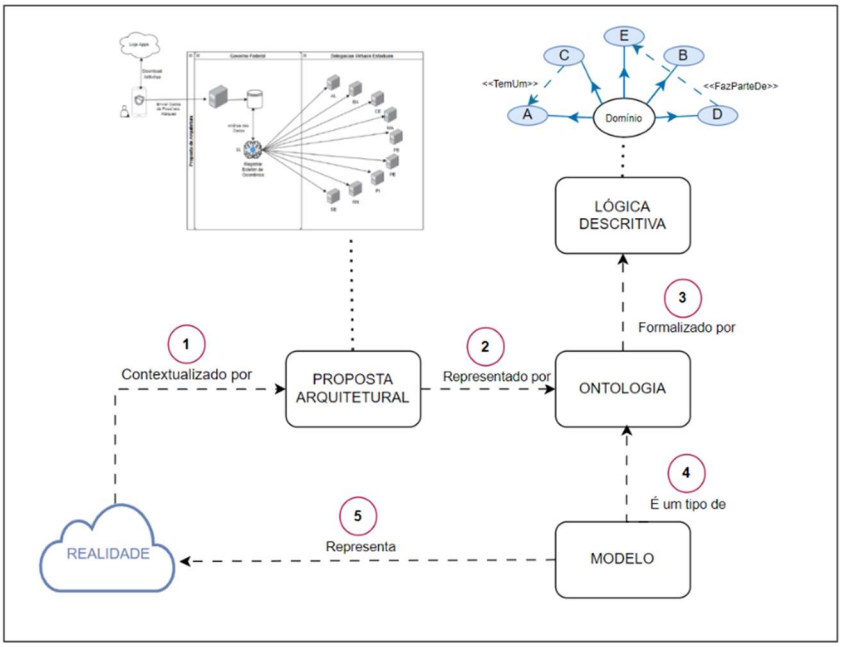

Figura 1 – Proposta Arquitetural representado por Ontologia. (Figura adaptada)

Acredita-se que a Proposta Arquitetural apresentada pode representar e abstrair uma realidade, seja atual (modelo AS-IS) ou futura (modelo TO-BE). Essa proposta pode ser formalizada por uma Ontologia, que descreve entidades e processos essenciais para um sistema. Ferramentas como OWL e UML podem ser usadas para esse propósito, sendo a OWL utilizada junto com o software Protégé para construir uma Ontologia voltada à classificação de delitos informáticos.

O Modelo gerado a partir da Ontologia seguirá regras de Lógica Descritiva, permitindo representar e raciocinar sobre Conceitos, Instâncias e relações de forma estruturada. Uma vez implementado, o Modelo estará pronto para ser utilizado por uma arquitetura

e Sistemas Multiagentes como a que é representada pela figura abaixo:

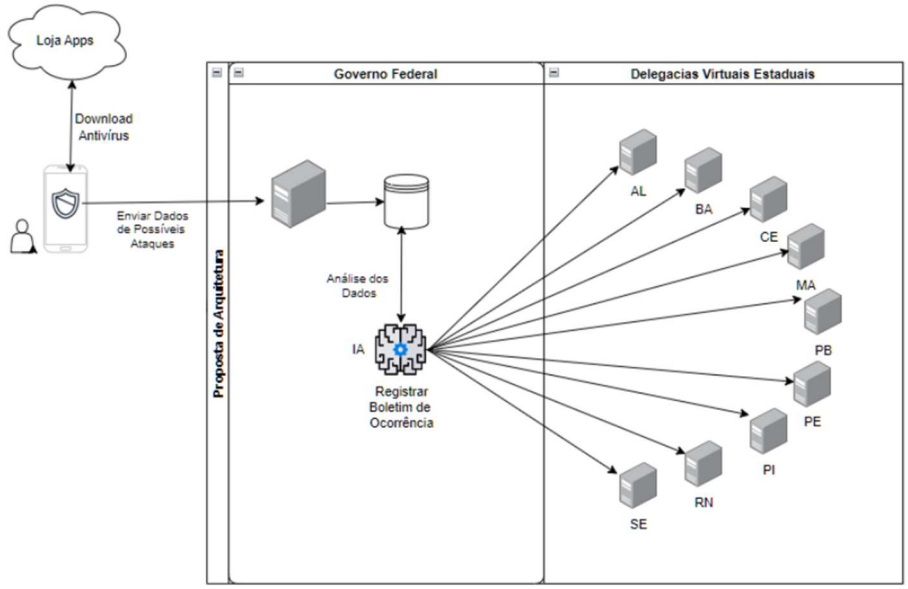

Figura 2 – Proposta de Arquitetura de Sistemas Multiagentes

A proposta de Arquitetura de Sistemas Multiagentes, apresentada na Figura 2, propõe em sua origem a criação de 4 agentes, contudo para efeitos de entrega relacionada à Disciplina de Modelagem Conceitual e Raciocínio Automático, foram desenvolvidos apenas dois agentes. Sendo tais agentes idealizadamente embarcados no disposto mobile representado na Figura 2 pela imagem do dispositivo móvel (celular).

O primeiro Agente tem objetivo de analisar a URL para verificar se a mesma é maliciosa. Caso positivo, envia a informação para o segundo Agente, que tem como objetivo consultar a Ontologia e retornar os artigos que possam tipificar o crime realizado.

O AGENTE_1 possui o papel de agente Investigador, pois:

- Analisa os dados relevantes a partir da fonte de dados que o usuário acessa. Por exemplo: Uma URL.
- Analisa padrões de URLs (*Uniform Resource Locator*) malignas e encontra um padrão associado a um crime de Estelionato Eletrônico e executado a partir de uma ação de *Phishing*.

O AGENTE_2 possui o papel de agente Consultor, pois:

- A partir do tipo da técnica de Phishing que será verificada pelo AGENTE_1, é realizada uma consulta à Ontologia pelo AGENTE_2 e se obtém os Artigos do código penal que possa tipificar o crime de estelionato eletrônico com uso da técnica de *phishing*.

5 AGENTES SCRIPT

Os scripts utilizados no desenvolvimento dos Agentes foram desenvolvidos em linguagem Python e executados na plataforma do *google colabs*.

AGENTE_1: - Investigador

```python
##### INICIO AGENTE_1 - VALIDAR TIPOS DE ATAQUES ########################

# Verificar tipo de ataque e processar
TIPOS_ATAQUE = ["EmailPhishing", "Smishing", "Vishing", "WebsitePhishing"]
def validar_tipo_ataque(tipo):
    if tipo not in TIPOS_ATAQUE:
        raise ValueError(f"Tipo de ataque '{tipo}' não reconhecido. Escolha entre: {', '.join(TIPOS_ATAQUE)}")

# Neste momento o agente indica que o ataque é de WebsitePhishing
tipo_ataque = "WebsitePhishing"

# Aqui é realizado uma validação para saber se o tipo de ataque informado,
# encontra-se na lista do tipo de ataque que deve ser validado.
validar_tipo_ataque(tipo_ataque)

# Informa na tela qual o tipo de ataque foi selecionado
print(f"Tipo de ataque selecionado: {tipo_ataque}")

######################### FIM AGENTE1 ###################################
```

AGENTE_2: - Consultor

```python
# Criar e salvar instâncias para o tipo de ataque selecionado
def criar_instancia_e_raciocinar(ontologia, tipo_ataque):
    # Validar a ontologia
    if ontologia is None:
        raise ValueError("A ontologia não foi carregada corretamente.")
    cls = ontologia[tipo_ataque]
    if cls:
      print("\nEntrou no if do CLS")
      # Criar instância do tipo de ataque
      instance_name = f"{tipo_ataque.lower()}_instance"
      if isinstance(cls, ThingClass):
        instance = cls(instance_name, world=ontologia.world)
        # Verificar as instâncias da classe
        print(f"Instâncias da classe '{tipo_ataque}':")
        for inst in cls.instances():
          print(inst.name)
        # Iniciar raciocínio
        print("\nIniciando raciocínio com Pellet...")
        with ontologia:
            sync_reasoner_pellet(infer_property_values=True, infer_data_property_values=True)
        print("Raciocínio concluído.")
        # Consultar artigos relacionados ao ataque
        query = f"""
          PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
          PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
          PREFIX ex: <http://www.delitosinformaticos.org/ontologia#>

          SELECT DISTINCT ?associatedClass
          WHERE {{
              ex:{instance_name} ex:éTipificadoPor ?associatedEntity.
              ?associatedEntity rdf:type ?associatedClass.
          }}
          """
        resultados = consulta_sparql(query)
        # Exibir os resultados da consulta
        print("\nResultados da consulta SPARQL:")
        for resultado in resultados:
            print(f" - {resultado}")
      else:
          print(f"Error: '{tipo_ataque}' is not a ThingClass. It is of type {type(cls)}")
          #print(f"Classe '{tipo_ataque}' não encontrada na ontologia.")
    else:
      print(f"Classe '{tipo_ataque}' não encontrada na ontologia.")

```

Resultado após consulta à Ontologia:

```
Instâncias na ontologia:
 - art_171_paragrafo_2A (<FusionClass DrOntoDelitosInformaticos (2).Art_171_Paragrafo_2A, DrOntoDelitosInformaticos (2).Art_171_Paragrafo_2B>)
 - websitephishing1 (DrOntoDelitosInformaticos (2).WebsitePhishing)
 - websitephishing2 (DrOntoDelitosInformaticos (2).WebsitePhishing)
```

O Script completo pode ser visualizado e baixado a partir do repositório Gitlab. O acesso ao repositório está disponível no ANEXO A deste relatório.

O script do AGENTE_1 tem por objetivo verificar numa possível fonte se a URL que está sendo usada é maligna, caso seja positivo deve ser chamado o AGENTE_2

O AGENTE_2 carrega a Ontologia, lista todas as propriedades dela, cria uma Instância e envia como parâmetro para uma consulta SPARQL. Espera-se que a Ontologia realize um raciocínio e retorne todas as classes que estejam associadas a essa instância. Dessa forma, passaremos a saber quais são as Classes, que representam os artigos do código penal, que podem tipificar o crime de Estelionato Digital a partir do uso da Técnica de *Phishing.*

6 A ONTOLOGIA

A Ontologia utilizada como fonte de conhecimento foi desenvolvida em pesquisas relacionadas ao Mestrado e atualizada na disciplina em questão. Atualmente conta com a seguinte métrica:

Axiom	1.348
Logical axiom count	331
Declaration axioms count	227
Class count	158
Object property count	55
Data property count	21
Individual count	30
Annotation Property count	16

Figura 3 – Métricas da Ontologia

Embora não tenha sido aplicada integralmente neste trabalho, foram utilizadas as classes, relacionamentos, propriedades de objetos e dados pertinentes às classes associadas à técnica de *Phishing*.

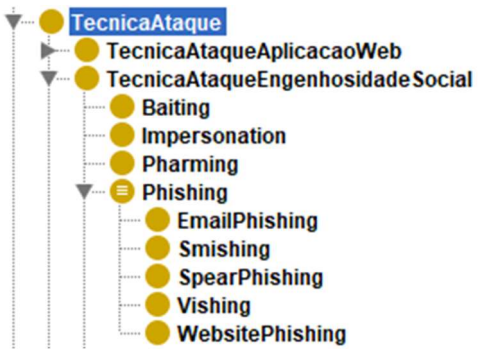

Figura 4 – Estrutura das classes relacionadas a Técnicas de ataques mapeados

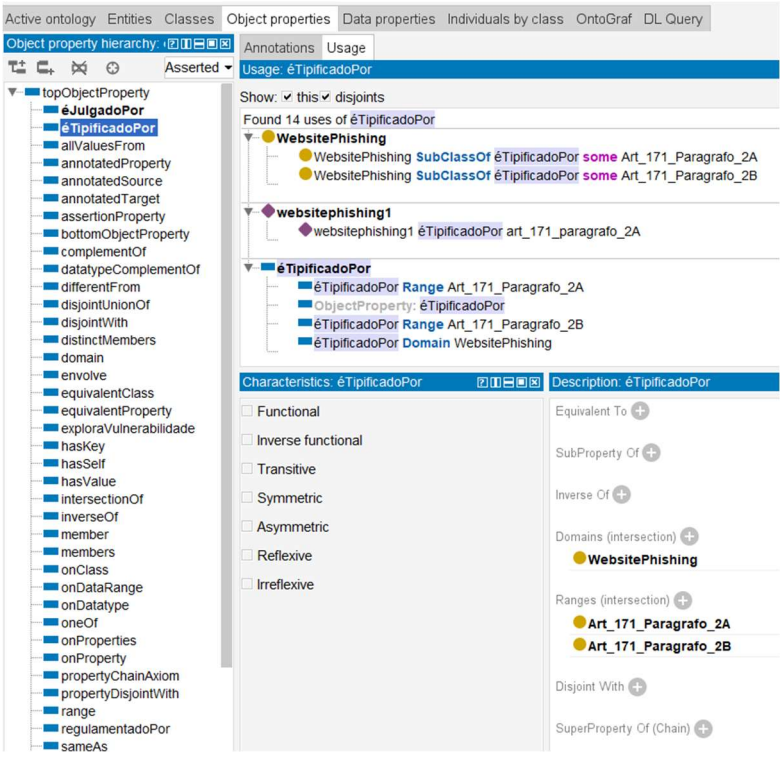

Figura 5 – Associação de *Object Properties*

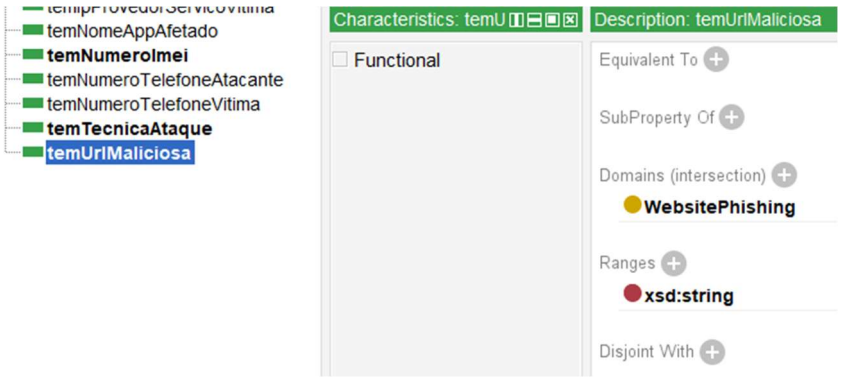

Figura 6 – Associação de *Data Properties*

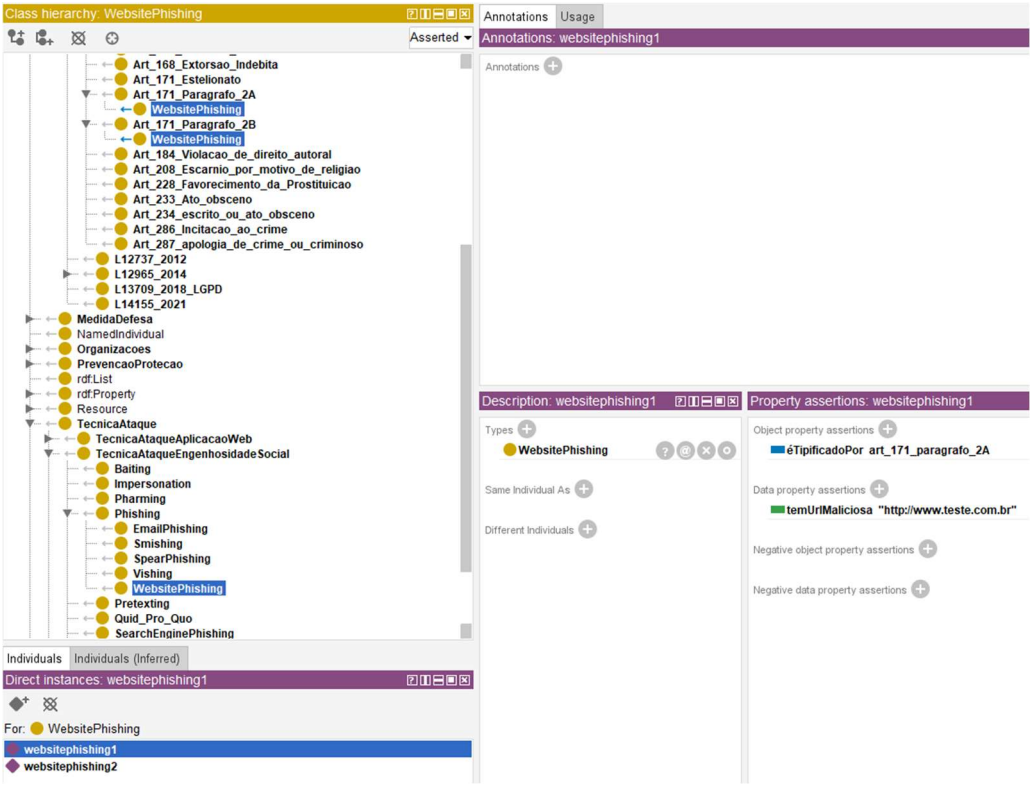

Figura 7 – Associação da Classe *WebsitePhishing* e suas relações com as classes que tipificam o crime

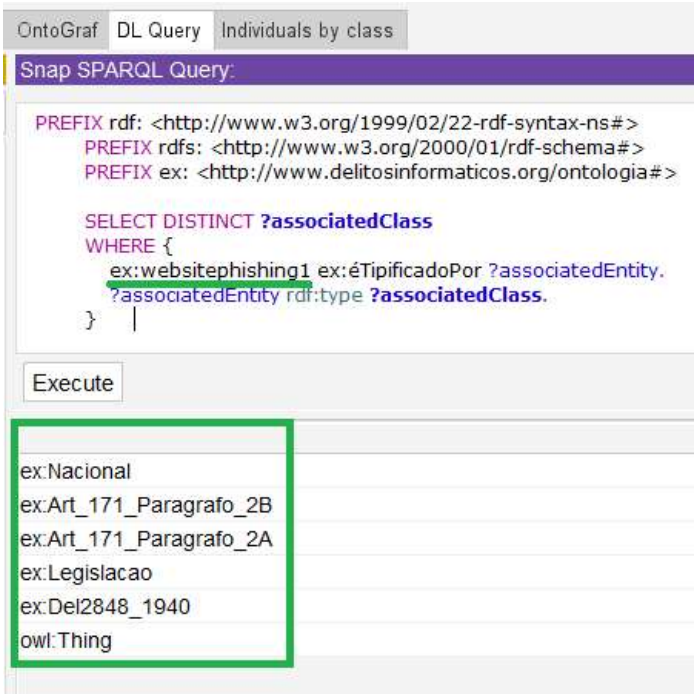

Figura 8 – Script SPARQL que consulta a ontologia

O script apresentado na Figura 8, representado em linguagem SPARQL, tem o objetivo de consultar a Ontologia e retornar todas as classes cuja a instância (individuo) ex:websitephishing1 pertence.

7 CONCLUSÃO:

Este trabalho aborda uma proposta para modelar e automatizar o raciocínio em cenários de crimes cibernéticos, especialmente em casos de phishing. A arquitetura proposta integra agentes computacionais e uma ontologia, o que possibilita a organização e a utilização estruturada de dados relevantes para a tipificação de crimes. Os agentes desempenham papéis complementares: um foca na detecção de URLs maliciosas, enquanto o outro interpreta as informações obtidas com base no conhecimento ontológico, associando-as a artigos do Código Penal.

Embora os resultados preliminares demonstrem potencial, a proposta não foi implementada em sua totalidade, limitando-se a dois agentes e a funcionalidades básicas. Isso indica a viabilidade inicial, mas também aponta para lacunas que precisam ser abordadas em estudos futuros.

ANEXO A – Repositório para acesso aos artefatos

Todos os artefatos produzidos nesta disciplina, tais como Ontologia, script em Python, este relatório e vídeo da apresentação, podem ser visualizados e baixados a partir do repositório no Github.

Link para acesso: https://github.com/0jcs0/mora