



# Smart Shield Proxy

## หลักการและเหตุผล

ปัจจุบันการโจมตีเว็บไซต์มีความซับซ้อนมากขึ้น ระบบป้องกันแบบเดิม เช่น WAF ที่ใช้กฎตายตัว มักตรวจจับการโจมตีรูปแบบใหม่หรือการดัดแปลงคำสั่งได้ยาก โพรเจกต์นี้จึงเสนอแนวคิด Hybrid Security โดยใช้ Machine Learning วิเคราะห์ Payload ร่วมกับ ModSecurity เพื่อเพิ่มความแม่นยำในการตรวจจับและตอบสนองต่อภัยคุกคามแบบ Real-time.

## วัตถุประสงค์

1. พัฒนาระบบ Web Proxy อัจฉริยะ สำหรับกรองกราฟฟิกเว็บ
2. ใช้ Machine Learning เพื่อเพิ่มประสิทธิภาพการตรวจจับภัยคุกคาม
3. ใช้แนวคิด Defense-in-Depth เพื่อเสริมความปลอดภัยหลายชั้น
4. สร้างระบบ Real-time Monitoring สำหรับตรวจสอบและรายงานผล
5. จัดเตรียม Dataset เพื่อให้ระบบเรียนรู้และพัฒนาต่อเนื่อง

## ขอบเขตระบบ

การตรวจจับ: เน้นตรวจจับ SQL Injection และ XSS ผ่าน HTTP  
ระบบป้องกัน: 2 ชั้น — Layer 1: ML Model, Layer 2: ModSecurity  
การตอบสนอง: บล็อกด้วย HTTP 403 Forbidden และบันทึก Log อัตโนมัติ  
การแสดงผล: Dashboard แสดงสถิติการโจมตีและความแม่นยำของโมเดล

## ขั้นตอนการทำงาน

1. Attacker ส่ง Request โจมตีมายัง Proxy (Port 8080)
2. ML Proxy ทำ Normalize และให้ ML Model วิเคราะห์
3. Layer 1 ถ้า ML ตรวจพบการโจมตี → Block และบันทึก Log
4. Layer 2 ถ้า ML ปล่อยให้ผ่าน → ส่งไป Web Server (Port 80) ให้ ModSecurity ตรวจสอบซ้ำ
5. Monitoring ผู้ดูแลระบบดูผลผ่าน Dashboard

## เครื่องมือที่ใช้

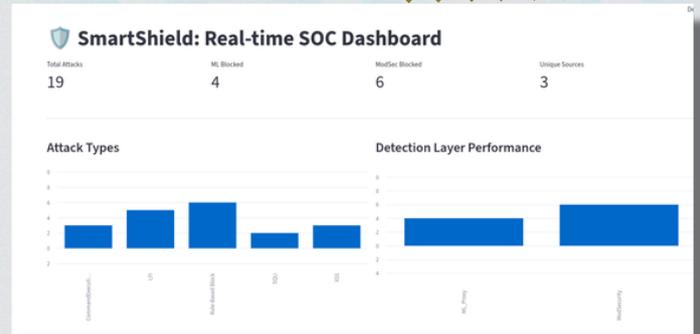


Streamlit



scikit learn

## RESULT



**Recent Attack Feed**

Timestamp	Type	payload	Source	Detector
2024-03-20 18:39:05	Rule Based Block	user=admin&password=00	192.168.1.10	ModSecurity
2024-03-20 18:39:05	Rule Based Block	user=admin&password=00	192.168.1.10	ModSecurity
2024-03-20 18:39:04	Rule Based Block	user=admin&password=00	192.168.1.10	ModSecurity
2024-03-20 18:39:04	Rule Based Block	user=admin&password=00	192.168.1.10	ModSecurity
2024-03-20 18:39:04	Rule Based Block	user=admin&password=00	192.168.1.10	ModSecurity
2024-03-20 18:39:04	Rule Based Block	user=admin&password=00	192.168.1.10	ModSecurity
2024-03-20 18:27:18	LFI	file:///etc/passwd	192.168.1.10	ML_Proxy
2024-03-20 18:27:12	XSS	file:///etc/passwd"/></script>	192.168.1.10	ModSecurity
2024-03-20 18:27:04	SQI	id=1 or 1=1	192.168.1.10	ML_Proxy
2024-03-20 17:28:05	LFI	cmd=ls;pwd	127.0.0.1	ML_Proxy
2024-03-20 17:28:00	Rule Based Block	id=1 or 1=1	127.0.0.1	ModSecurity

## สรุปผลการทำงาน

ระบบสามารถทำงานได้อย่างมีประสิทธิภาพตามสมมติฐาน โดย Machine Learning ช่วยให้การตรวจจับ Payload ที่มีความหลากหลายทำได้ดีขึ้น ในขณะที่ ModSecurity ช่วยเสริมความปลอดภัยในส่วนของกฎมาตรฐานสากล ผลการทดสอบพบว่าระบบสามารถบล็อกการโจมตีได้แบบ Real-time และช่วยให้ผู้ดูแลระบบมองเห็นภาพรวมของภัยคุกคามผ่าน Dashboard ได้อย่างชัดเจน พร้อมรองรับการอัปเดต Dataset เพื่อเพิ่มความฉลาดของโมเดลได้ในอนาคต

### CODE

```

from flask import Flask, request
import requests
import os
from datetime import datetime
from detect_request import detect
from urllib.parse import urlparse

app = Flask(__name__)

@app.route('/')
def index():
    request_headers = request.headers
    request_body = request.get_data()
    request_url = request.url

    # Detect request
    is_malicious = detect(request_headers, request_body, request_url)

    if is_malicious:
        return "Blocked"
    else:
        # Forward request to web server
        response = requests.get(request.url)
        return response

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=8080)

```

```

#!/usr/bin/env python3
import sys
import os
import logging
import argparse
import subprocess
import time
import json
import requests
import urllib3
import urllib3.util.retry
import urllib3.util.retry.Retry
import urllib3.util.retry.Retry

# Configure logging
SRC_LOG = "/var/log/apache2/modsec_audit"
DEST_LOG = os.path.expanduser("~/CyberP")

def main():
    parser = argparse.ArgumentParser()
    parser.add_argument("-s", "--src-log", type=str, default=SRC_LOG)
    parser.add_argument("-d", "--dest-log", type=str, default=DEST_LOG)
    parser.add_argument("-v", "--verbose", action="store_true")
    parser.add_argument("-m", "--mode", type=str, default="audit")
    parser.add_argument("-u", "--url", type=str, default="http://127.0.0.1:8080")
    parser.add_argument("-p", "--port", type=int, default=8080)
    parser.add_argument("-t", "--timeout", type=int, default=10)
    parser.add_argument("-r", "--retry", type=int, default=3)
    parser.add_argument("-c", "--cert", type=str, default="")
    parser.add_argument("-k", "--key", type=str, default="")
    parser.add_argument("-e", "--engine", type=str, default="audit")
    parser.add_argument("-f", "--file", type=str, default="")
    parser.add_argument("-l", "--log-level", type=str, default="INFO")
    parser.add_argument("-o", "--output", type=str, default="")
    parser.add_argument("-q", "--quiet", action="store_true")
    parser.add_argument("-h", "--help", action="store_true")
    parser.add_argument("-V", "--version", action="store_true")

    args = parser.parse_args()

    # Setup logging
    logging.basicConfig(level=args.log_level, filename=args.dest_log)

    # Setup urllib3
    urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)

    # Setup requests
    session = requests.Session()
    session.verify = args.cert
    session.cert = args.key

    # Setup headers
    headers = {}
    if args.c:
        headers["Content-Type"] = args.c

    # Setup cookies
    cookies = {}
    if args.k:
        cookies[args.k] = args.c

    # Setup proxies
    proxies = {}
    if args.p:
        proxies["http"] = args.p
        proxies["https"] = args.p

    # Setup timeout
    timeout = args.timeout

    # Setup retry
    retry = Retry(total=args.retry, backoff_factor=1)

    # Setup engine
    engine = args.engine

    # Setup file
    file = args.file

    # Setup quiet
    quiet = args.quiet

    # Setup version
    version = args.version

    # Setup help
    help = args.help

    # Main loop
    while True:
        # Send request
        response = session.get(args.url, headers=headers, cookies=cookies, proxies=proxies, timeout=timeout, allow_redirects=False)

        # Parse response
        parse_response(response)

        # Sleep
        time.sleep(1)

    # Exit
    sys.exit(0)

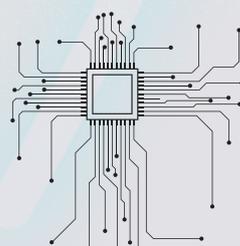
def parse_response(response):
    # Parse response
    logging.info("Request: %s", response.request.url)
    logging.info("Status: %s", response.status_code)
    logging.info("Headers: %s", response.headers)
    logging.info("Body: %s", response.text)

    # Detect request
    is_malicious = detect(response.request.headers, response.request.body, response.request.url)

    if is_malicious:
        logging.info("Blocked")
    else:
        logging.info("Allowed")

if __name__ == '__main__':
    main()

```



- Member of Group 10 Sec 1
- 673380484-6 กัทภภูมิ ไชยันโต
  - 673380486-2 ยศพล สุกแสง
  - 673380192-9 สิริมงคล ม่วงลาย
  - 673380013-5 วณัญฐนนท์ รือหาร
  - 673380006-2 ชิตธรรม นาเมืองรักษ์