USENIX Security 2024

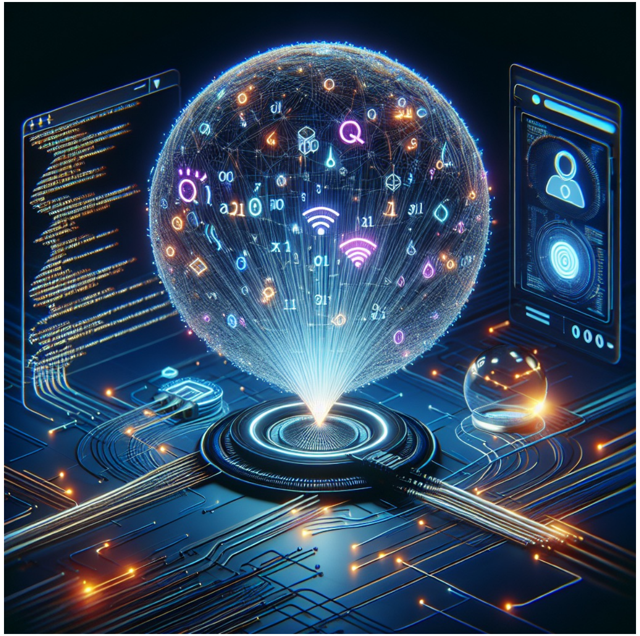
EaTVul: ChatGPT-based Evasion Attack Against Software Vulnerability Detection

Shigang Liu[1,2], Di Cao[2], Junae Kim[3], Tamas Abraham[3], Paul Montague[3], Seyit Camtepe[1], Jun Zhang[2], and Yang Xiang[2]

[1]CSIRO's Data61, [2]Swinburne University of Technology, [3]DST Group, Australia

Background

- Machine learning-as-a-service has been widely applied in software security.
- Adversarial learning has long been a threat for cybersecurity.
- There is lack of thorough assessment the security issues when facing adversarial learning.



CSIRO

A Motivation Example

Prediction: Vulnerable (93.2%)

```c
static struct ast_sip_pubsub_body_generator *subscription_get_generator_from_rdata(pjsip_rx_data *rdata,
    const struct ast_sip_subscription_handler *handler)
{
    pjsip_accept_hdr *accept_header = (pjsip_accept_hdr *) &rdata->msg_info.msg;
    char accept[AST_SIP_MAX_ACCEPT][64];
    size_t num_accept_headers = 0;

    while ((accept_header = pjsip_msg_find_hdr(rdata->msg_info.msg, PJSIP_H_ACCEPT, accept_header)) &&
        (num_accept_headers < AST_SIP_MAX_ACCEPT)) {
        int i;

        for (i = 0; i < accept_header->count && num_accept_headers < AST_SIP_MAX_ACCEPT; ++i) {
            if (!exceptional_accept(&accept_header->values[i])) {
                ast_copy_pj_str(accept[num_accept_headers], &accept_header->values[i], sizeof(accept[num_accept_headers]));
                ++num_accept_headers;
            }
        }
    }

    if (num_accept_headers == 0) {
        /* If a SUBSCRIBE contains no Accept headers, then we must assume that
         * the default accept type for the event package is to be used.
         */
        ast_copy_string(accept[0], handler->notifier->default_accept, sizeof(accept[0]));
        num_accept_headers = 1;
    }

    return find_body_generator(accept, num_accept_headers, handler->body_type);
}
```

Prediction: Benign (87.4%)

```c
static struct ast_sip_pubsub_body_generator *subscription_get_generator_from_rdata(pjsip_rx_data *rdata,
    const struct ast_sip_subscription_handler *handler)
{
    pjsip_accept_hdr *accept_header = (pjsip_accept_hdr *) &rdata->msg_info.msg;
    char accept[AST_SIP_MAX_ACCEPT][64];
    size_t num_accept_headers = 0;

    struct a_chan myAChan;
    struct b_const myBConst;

    // initialize values
    myAChan.value = 0;
    myBConst.value = 0.0;

    while ((accept_header = pjsip_msg_find_hdr(rdata->msg_info.msg, PJSIP_H_ACCEPT, accept_header)) &&
        (num_accept_headers < AST_SIP_MAX_ACCEPT)) {
        int i;

        for (i = 0; i < accept_header->count && num_accept_headers < AST_SIP_MAX_ACCEPT; ++i) {
            if (!exceptional_accept(&accept_header->values[i])) {
                ast_copy_pj_str(accept[num_accept_headers], &accept_header->values[i], sizeof(accept[num_accept_headers]));
                ++num_accept_headers;
            }
        }
    }
```

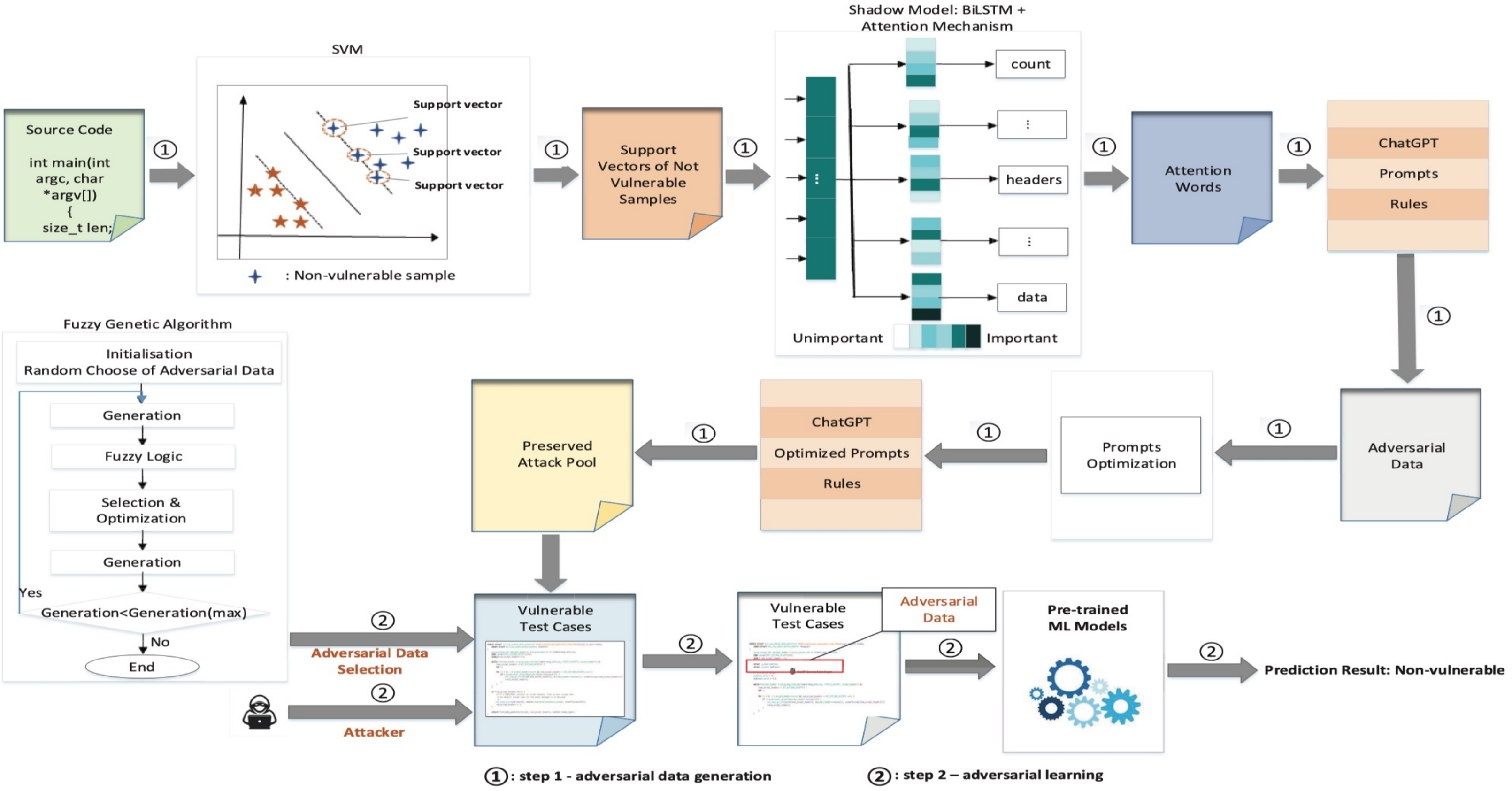
Our work aims to…

- Explore the susceptibility of machine learning/deep neural network models to adversarial attacks.
- Develop an effective scheme to generate adversarial code and inject it into vulnerable samples to bypass deep neural network systems.

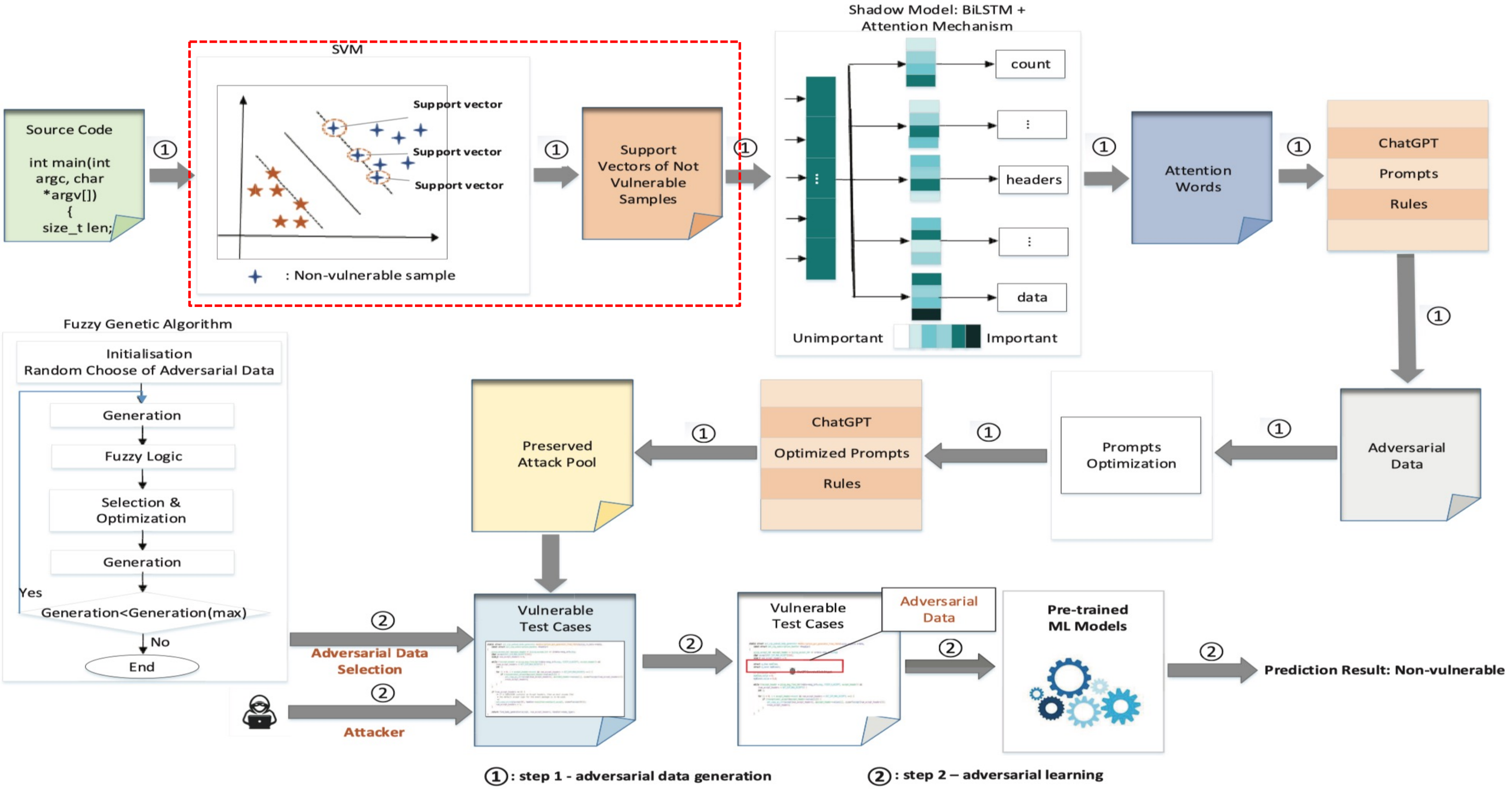
Assumption

- **Attacker's capability:** perturb the test queries and query the deployed vulnerability detection model.
- **Attacker's Knowledge:** no access to the architecture and parameters of target models.
- **Attacker's Goal:** deceive the targeted vulnerability detection tools through imperceptible modifications to the inputs.

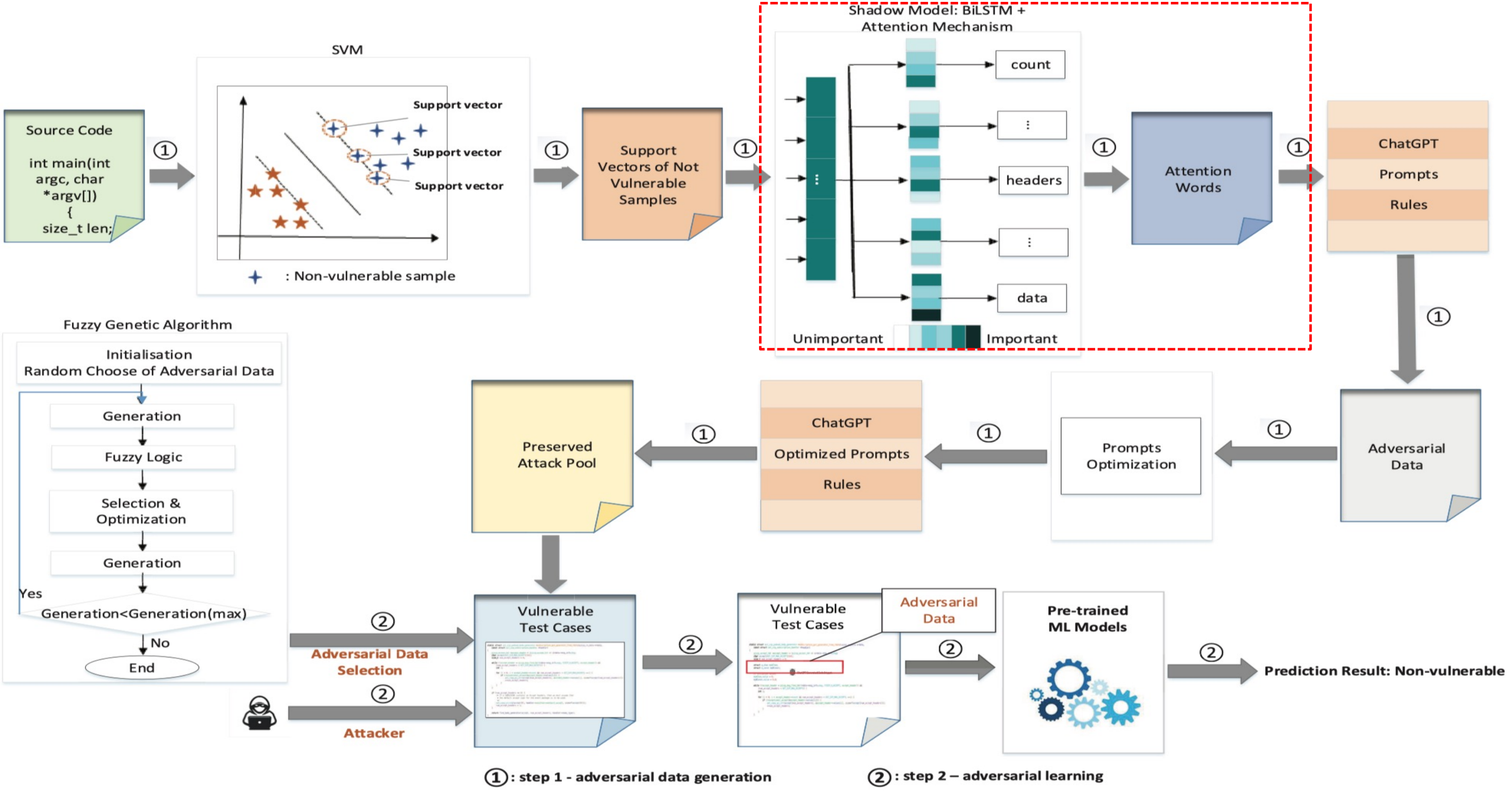
Proposed EaTVul



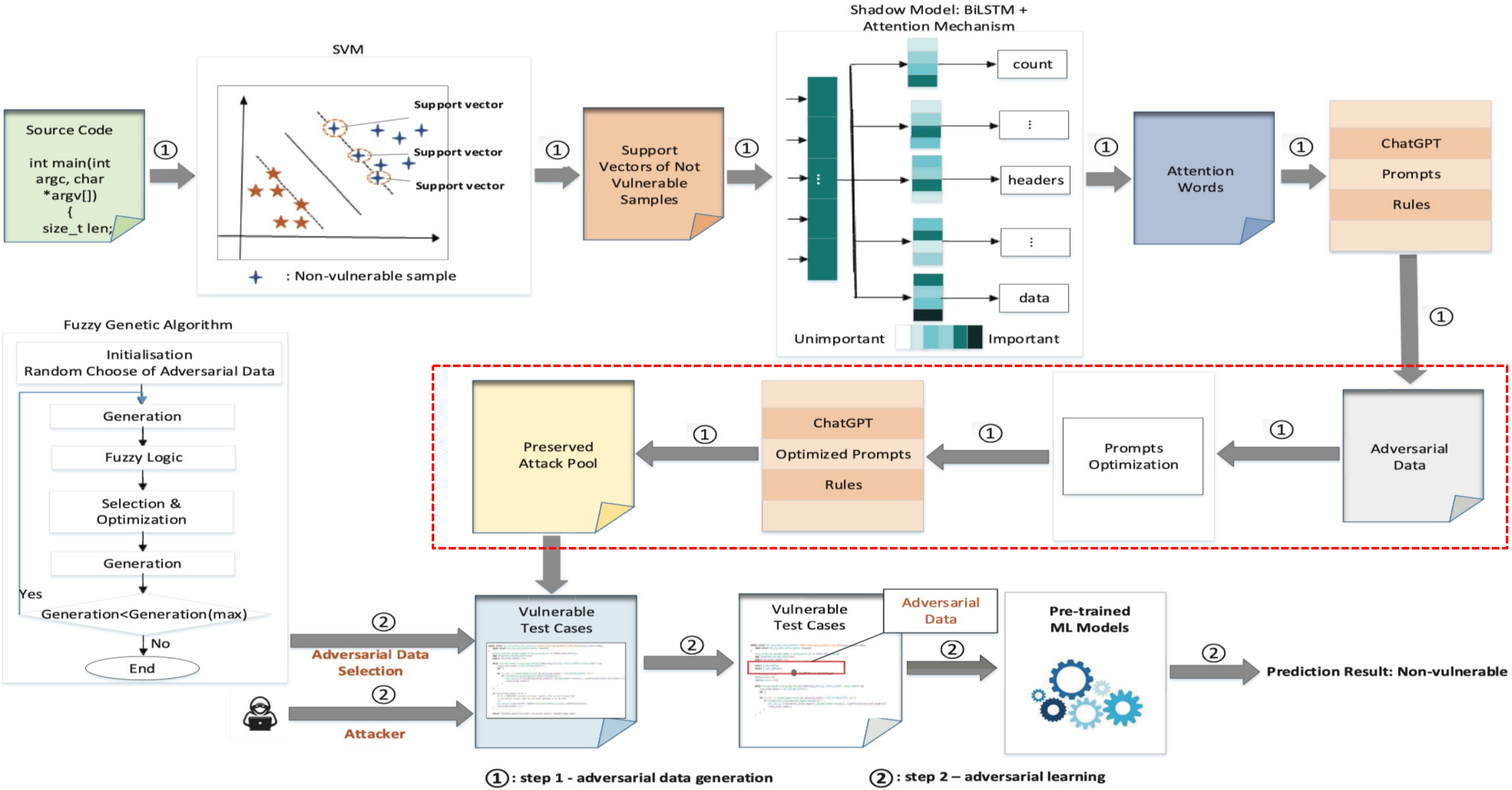
Proposed EaTVul



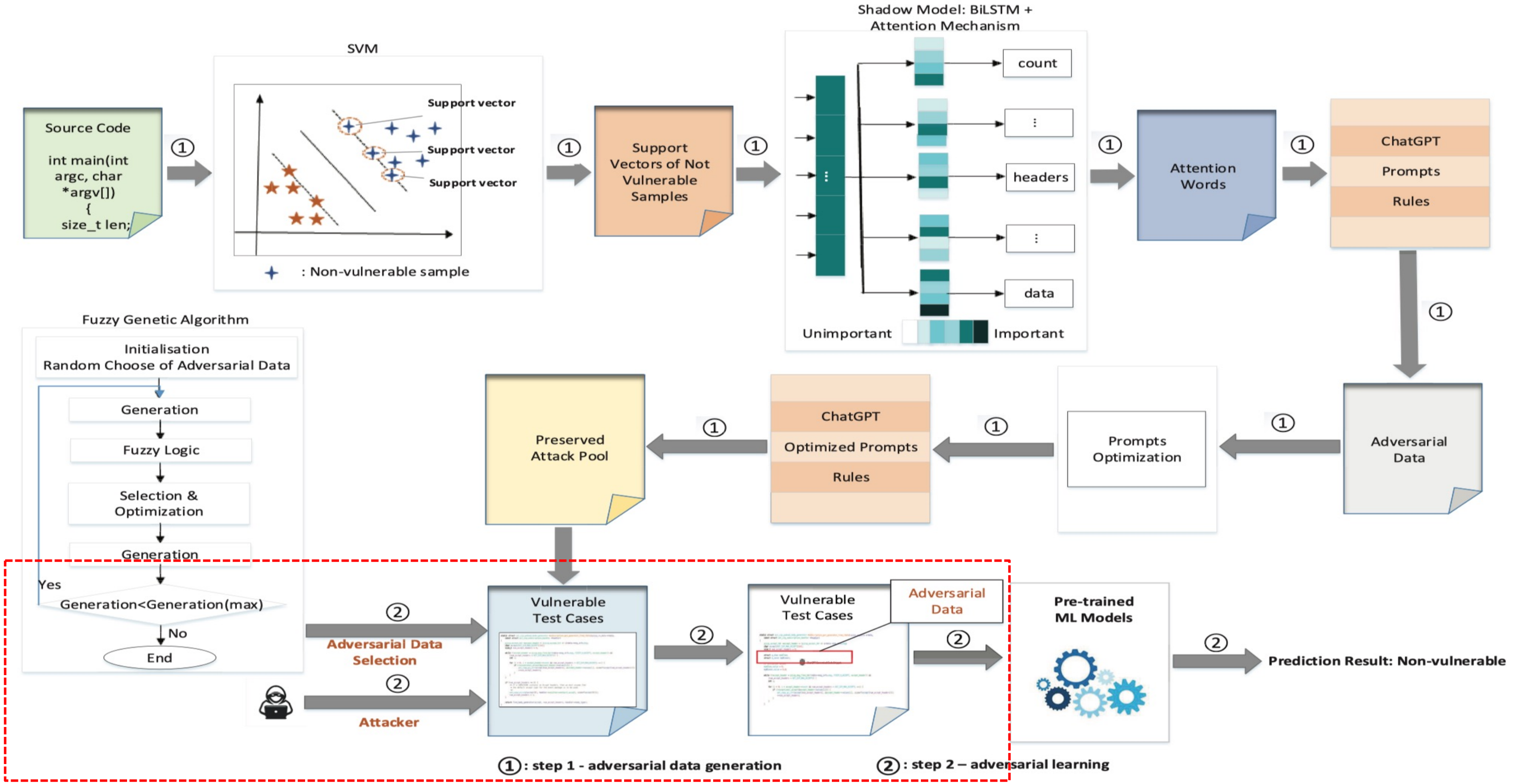
Proposed EaTVul



Proposed EaTVul



Proposed EaTVul



Example of Identified Features

- Display of the important features identified by attention mechanism, with the importance decreases from red to yellow.
- The identified features include ***static, const, strstr, strchr, val, sscanf***.

```
static void sdp_fmtp_get(const char *attributes, const char *name, int *attr)
 {
     const char *kvp = "";
     int val;

     if (attributes && !(kvp = strstr(attributes, name))) {
     return;
     }

   if (kvp != attributes && *(kvp - 1) != ' ' && *(kvp - 1) != ';') {
     /* Keep searching as it might still be in the attributes string */
     sdp_fmtp_get(strchr(kvp, ';'), name, attr);
   } else if (sscanf(kvp, "%*[^=]=%30d", &val) == 1) {
     *attr = val;
   }
 }
```

Example of Generated Adversarial Data

```c
#include <stdio.h>

struct a_chan {
    int value;
};

struct b_const {
    double value;
};

static const char a_array[] = "NULL";

void myFunction() {

    struct a_chan myAChan;
    struct b_const myBConst;

    myAChan.value = 0;
    myBConst.value = 0.0;

    for (int i = 0; i < 10; i++) {
        printf("Iteration %d:\n", i);

        myAChan.value += i;
        myBConst.value += 0.5 * i;

        printf("a_chan value: %d\n", myAChan.value)
        printf("b_const value: %f\n", myBConst.value)
    }

}
```

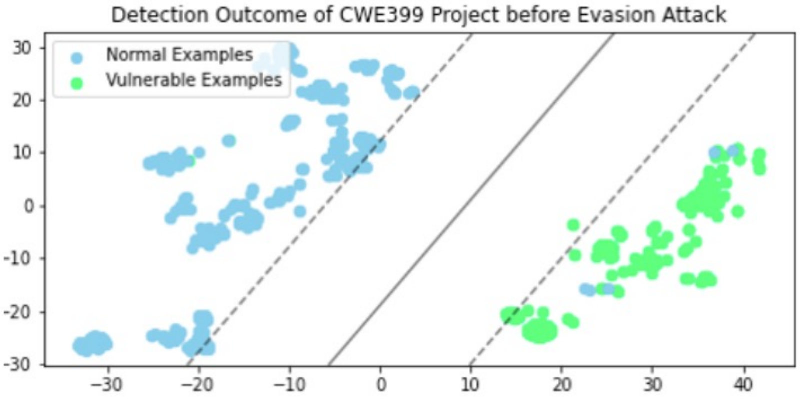
(a) Raw data generated by ChatGPT

```c
struct a_chan { int value; } queue;
struct b_const { double value; } p_project;
static const char a_array[] = "NULL";

queue.value = 0;
p_project.value = 0.0;

for (int i = 0; i < 10; i++)
    printf("i = %d\n", i);
```

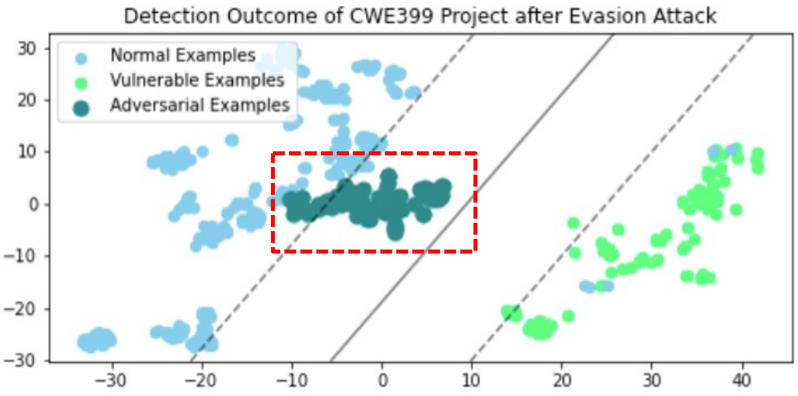
(b) Further optimized data by ChatGPT

Visualization

- The distribution of vulnerable, non-vulnerable samples, and the adversarial samples.



(a) Vulnerable and non-vulnerable (normal) samples



(b) Movement of adversarial samples

CSIRO

Evaluation

- RQ1: How effective is fuzzy genetic algorithm in selecting the seed adversarial data compared with randomization?
- RQ2: How effective is EaTVul based on adversarial data generated by ChatGPT originally and after optimization?
- RQ3: How effective is EaTVul with recently developed machine learning-based software vulnerability detection systems?
- RQ4: How effective is EaTVul when compared with state-of-the-art large language models (LLM) and other machine learning tools that using BiLSTM for software vulnerability detection?
- RQ5: How EaTVul behaves/performs regarding obfuscation/diversification methods?
- RQ6: How effective is EaTVul when generalized to other programming languages?

RQ4: EaTVul attack BiLSTM & LLMs

- Conducted experiment based on the datasets of CWE119, CWE399, Asterisk and OpenSSL.
- The target models include Poster-Lin, MDVD, CodeBERT, and CodeGen.

Target Model	Data	Snippet Size = 2				Snippet Size =3			
		Top@5	Top@ 10	Top@15	Top@ 20	Top@ 5	Top@ 10	Top@15	Top@20
Poster-Lin	Asterisk	0.900	0.867	0.756	0.758	1.000	1.000	0.878	0.858
	OpenSSL	1.000	1.000	0.745	0.717	1.000	1.000	0.911	0.875
	CWE119	0.933	0.934	0.899	0.867	1.000	0.967	0.956	0.925
	CWE399	1.000	0.833	0.823	0.767	1.000	1.000	1.000	0.917
MDVD	Asterisk	1.000	0.867	0.844	0.784	1.000	1.000	1.000	0.975
	OpenSSL	1.000	1.000	0.845	0.817	1.000	1.000	1.000	1.000
	CWE119	1.000	0.933	0.877	0.825	1.000	1.000	0.978	0.958
	CWE399	1.000	0.899	0.867	0.767	1.000	1.000	0.967	0.950
CodeBERT	Asterisk	0.900	0.850	0.803	0.740	0.900	0.885	0.845	0.832
	OpenSSL	0.800	0.800	0.768	0.735	0.900	0.864	0.858	0.835
	CWE119	0.900	0.840	0.834	0.785	1.000	0.935	0.911	0.865
	CWE399	0.900	0.825	0.786	0.776	1.000	0.920	0.878	0.858
CodeGen	Asterisk	0.900	0.867	0.844	0.784	0.933	0.925	0.899	0.867
	OpenSSL	0.900	1.000	0.845	0.817	0.956	0.911	0.875	0.845
	CWE119	1.000	0.933	0.877	0.825	1.000	1.000	0.928	0.875
	CWE399	1.000	0.899	0.867	0.767	1.000	0.950	0.917	0.880

RQ5: EaTVul attack Obfuscation/Diversification

- The baseline/attack models are EaTVul, Differentiable Obfuscator and Milo.
- The target models include Asteria and LineVul.
- Uses Juliet C/C++ Test Suite Datasets.

Dataset	Target Model	Attack Model	ASR	F1-Score
CWE119	Asteria	Differentiable Obfuscator	66.40	81.45
		Milo	35.80	
		EaTVul	**99.50**	
	LineVul	Differentiable Obfuscator	63.50	83.45
		Milo	44.30	
		EaTVul	**92.30**	
CWE399	Asteria	Differentiable Obfuscator	58.80	82.60
		Milo	26.30	
		EaTVul	**89.50**	
	LineVul	Differentiable Obfuscator	62.80	83.50
		Milo	28.70	
		EaTVul	**89.20**	
CWE416	Asteria	Differentiable Obfuscator	52.50	80.40
		Milo	20.35	
		EaTVul	**84.30**	
	LineVul	Differentiable Obfuscator	58.60	81.70
		Milo	19.80	
		EaTVul	**87.50**	

RQ6: Generalization of EaTVul

- Evaluate on Java programs.
- The attack models are EaTVul, Differentiable Obfuscator and Milo.
- The target models include FUNDED and VDet.

Target Model	Attack Model	ASR	F1-Score
FUNDED	Differentiable Obfuscator	53.50	85.35
	Milo	42.70	
	EaTVul	**88.60**	
VDet	Differentiable Obfuscator	63.50	86.60
	Milo	62.80	
	EaTVul	**87.30**	

CSIRO

Conclusion

- We propose a novel evasion attack approach, named EatVul, which produces code to evade ML/deep neural network-based vulnerability detectors.

- We have conducted an evaluation of EatVul against state- of-the-art baselines, and the experimental results demonstrated that our scheme achieved a 100% success rate in most cases with a snippet size of 4.

- We have made our proposed system, EatVul, available to the research community. The datasets and code are available at **https://github.com/wolong3385/EatVul-Resources**.

- This work demonstrates the susceptibility of vulnerable samples to manipulation and highlights the need for robust defense mechanisms capable of mitigating such adversarial attacks.

CSIRO DATA 61

SWINBURNE UNIVERSITY OF TECHNOLOGY

DST GROUP

Thank you!
Q&A