WAF Signature Generation with Real-Time Information on the Web

Masahito Kumazaki, Yukiko Yamaguchi,
Hajime Shimada, and Hirokazu Hasegawa
Graduate School of Informatics, Nagoya University

kumazaki@net.itc.nagoya-u.ac.jp

IARIA 名古屋大学 NAGOYA UNIVERSITY

Short resume

Presenter's profile

- Received Bachelor's degree from Nagoya University, Japan in 2020.
- Currently, master course student of Graduate School of Informatics, Nagoya University.
- My research interest includes Web security and network security

Outline

- Background
- Proposed system
- Experiments
- Conclusion and future works

Background

- Cyberattacks are increasing and sophisticated.

- We focused on two types of attacks.
  - Attacks that used published vulnerabilities
  - Zero-day attacks
    - Attacks before a provision of patches

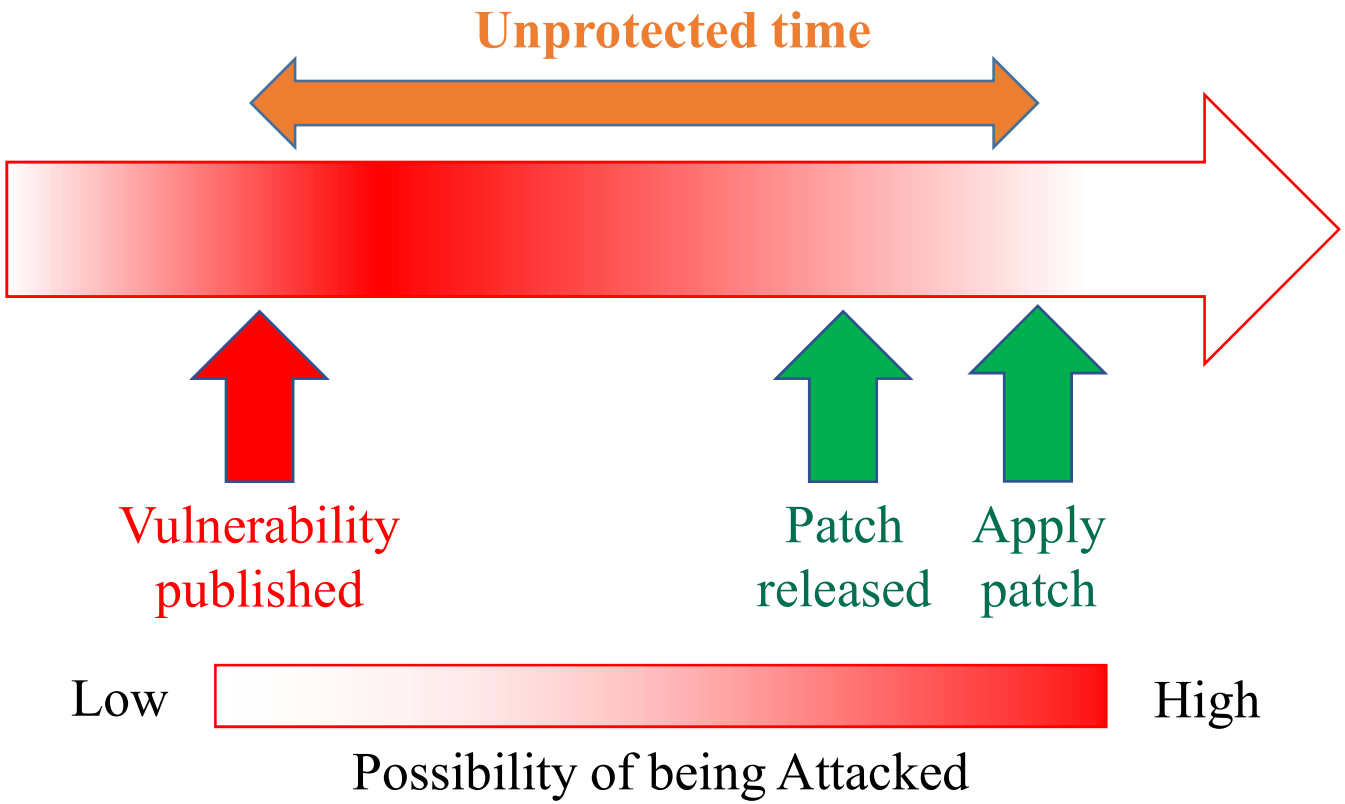
- There's already a lot of damage.
  - Ex.) Information leakage from the Equifax(2017)
    - This attack used published vulnerability of the Apache Struts 2.

Existing Countermeasures

- Multilayered defense
  - Access control with firewall
  - Network management with IDS/IPS
  - Prohibit the execution of unknown programs using a white list
- Apply fixed patches on time

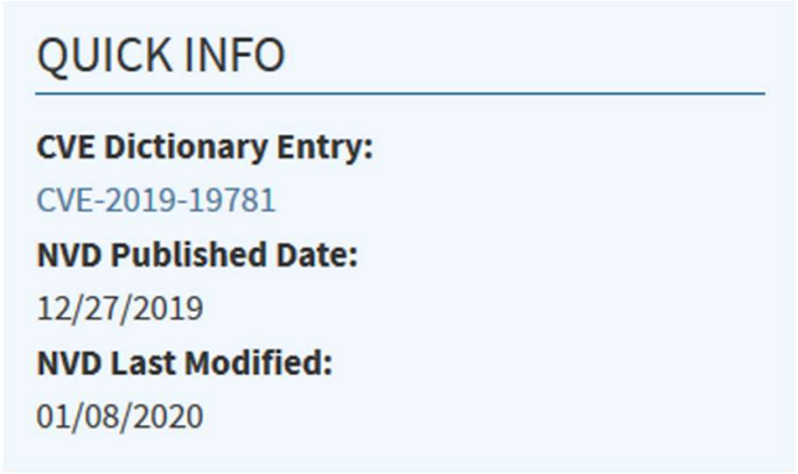
Problems of Countermeasure

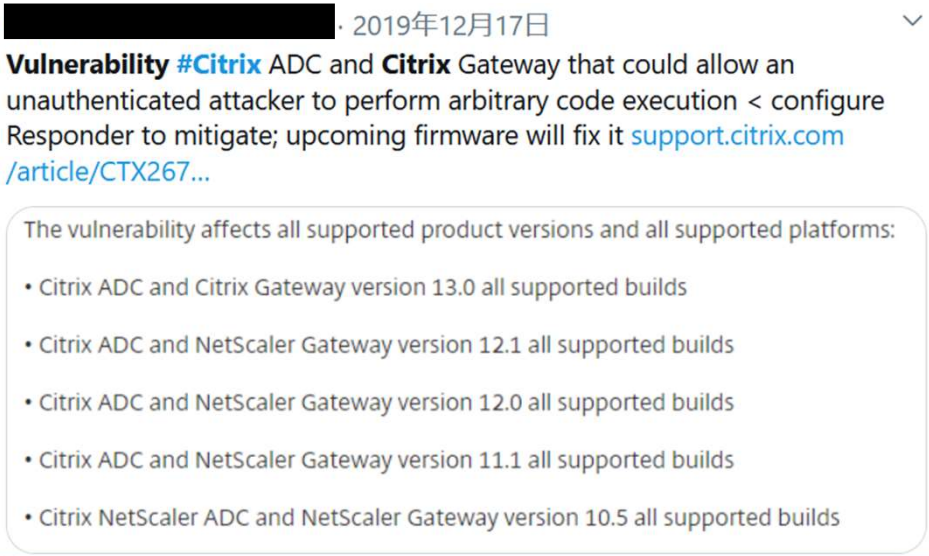
- Cyber attacks that slip through multilayered defense
  - Unprotected time to attack from vulnerability publication to patch release and we apply it



Advantage Using Real-time Information

- Recently, Vulnerabilities are often mentioned on SNS before they are officially published or registered in the DB.
  - Vulnerability of Citrix
    - It was registered in the NVD on December 27.
    - It was mentioned on the Twitter on December 17.





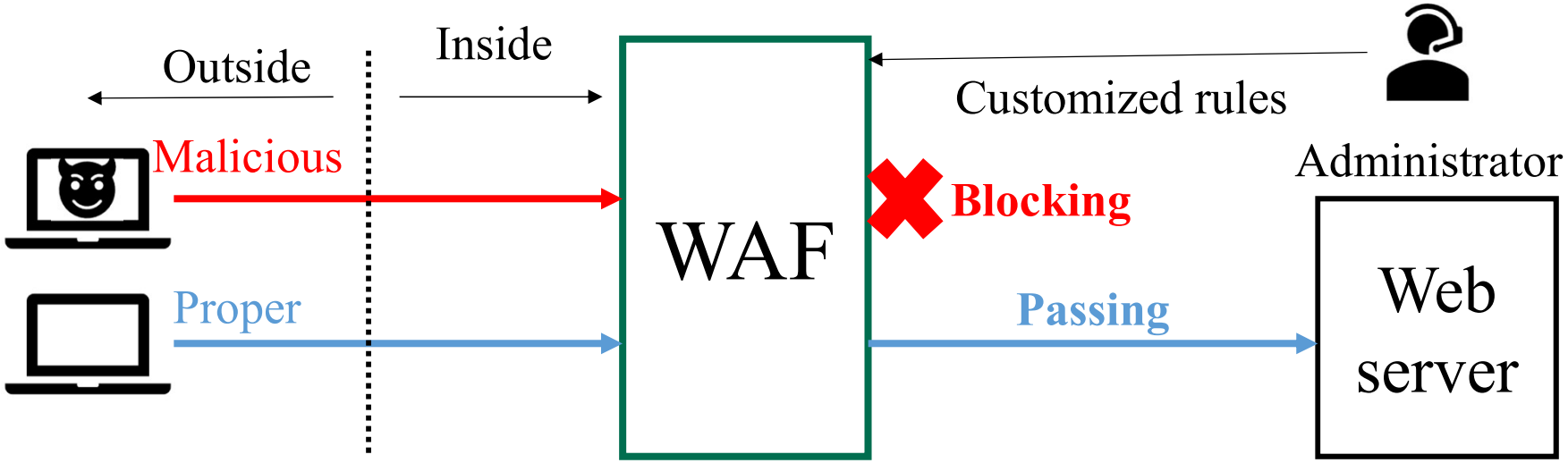
Proposed Method

- Collect vulnerability information from real-time information sources such as SNS
  - We can response vulnerability faster than using the vulnerability information database.
  - We may get more information sooner than the official announcement.

WAF signature generation system using real-time information on the Web

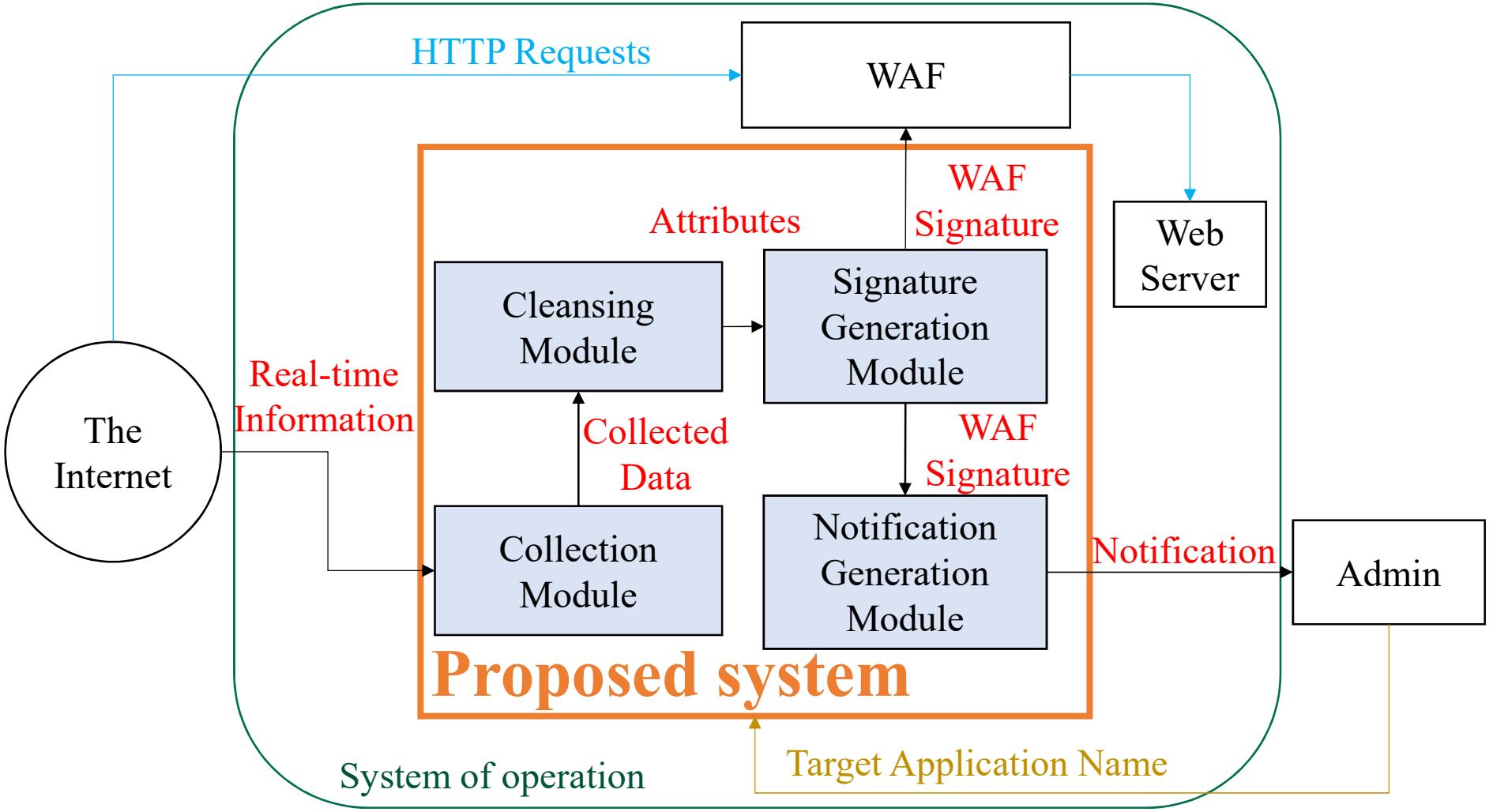
What's WAF and Assumption

- The organization running this system is running a Web Application Firewall
  - Web Application Firewall (WAF)
    - Check HTTP communications based on the rules
    - Depending on checking results, process them by passing, blocking etc.
- The administrator can customize rules of WAF



Proposed System Architecture and Data Frow

Proposed system consists 4 modules



Collection Module

- This module collects vulnerability information from these sources
  - Twitter
  - Stack Overflow
  - Reddit
  - teratail
  - Security StackExchange
- These sources equip IDs and timestamps, so this module stores them and body

Cleansing Module

- This module performs data cleansing on collected data to remove duplicate information and get the necessary information
- This module extracts the following attributes from the body of collected data
  - Application name
  - Vulnerability type
  - Version Information
  - Vulnerability identification information
    - Such as a Common Vulnerabilities and Exposures (CVE)

Signature Generation Module

- If the Web application name which is used in operating Web application system is included in attributes from the cleansing module, this module generates a WAF signature to block HTTP requests to that Web application.

- In addition, this module notifies the notification generation module regarding the signature generatiton.

Notification Generation Module

- This module generates a notification to the administrator
  - This notification include such as the name of the vulnerable Web application and its version

- The administrator will use this notification to remove the signature when the vulnerability is resolved

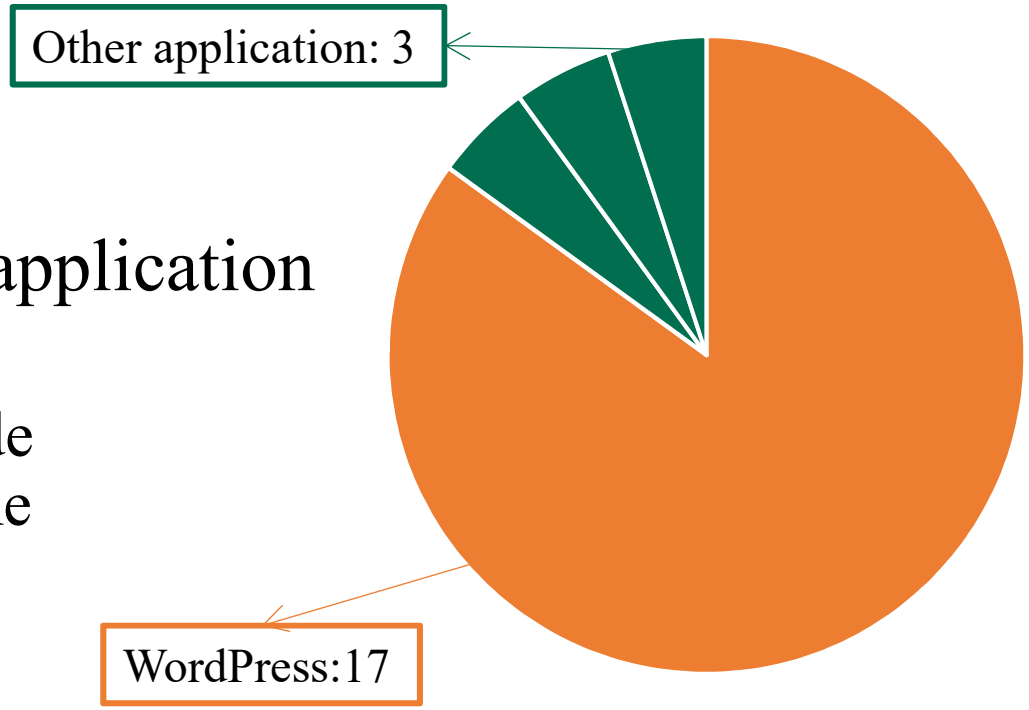
Experiment 1: Method

- WAF signature generation using CVE information
  - Use NVD's data feed which contains CVE information
  - Target application: Wordpress
- We extracted the following information from the data feed and generated signatures
  - CVE-ID
  - CPE name
    - It is identifies the platforms such as a product name
  - Version information
- We performed this daily process for 10 days from December 21, 2019 to December 30, 2019.

Experiment 1: Result-Extracted Information

- Extracted information
  - The results for all days during the collection period was same

- The system extracted 17 cases of the WordPress vulnerability and 3 cases of the other application vulnerability
  - Other applications include "wordpress" in their name



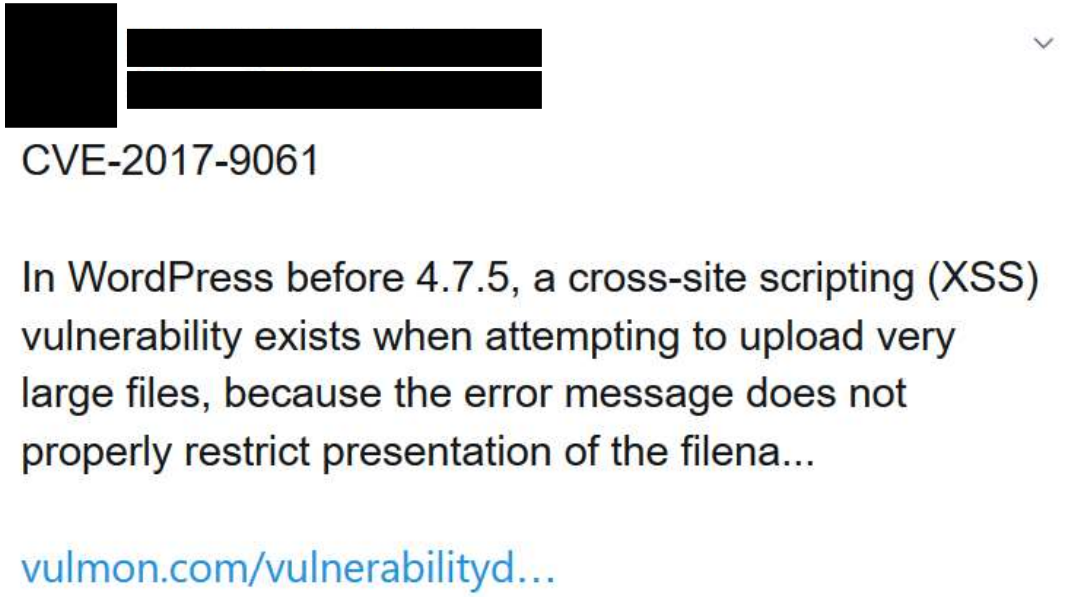
Experiment 1: Result-Generated Signatures

- The system generated the following signature from the extracted information
  - The signature for WordPress was generated from 17 cases of WordPress vulnerability information.
  - Other signatures were generated from 1 case corresponding to each application.
- The applications whose name include targeted application's name are blocked.
  - We need to improve the signature generation rule and tune it.

```
1 SecRule REQUEST_COOKIES|!REQUEST:/__utm/|
    REQUEST_COOKIES_NAMES|ARGS_NAMES|ARGS|XML:/* "
    import_export_wordpress_users" "phase:2,block,msg:'
    WordPress injection.'severity:'2',id:'15001'"
2 SecRule REQUEST_COOKIES|!REQUEST:/__utm/|
    REQUEST_COOKIES_NAMES|ARGS_NAMES|ARGS|XML:/* "
    wordpress" "phase:2,block,msg:'WordPress injection.'
    severity:'2',id:'15002'"
3 SecRule REQUEST_COOKIES|!REQUEST:/__utm/|
    REQUEST_COOKIES_NAMES|ARGS_NAMES|ARGS|XML:/* "
    wordpress_download_manager" "phase:2,block,msg:'
    WordPress injection.'severity:'2',id:'15003'"
4 SecRule REQUEST_COOKIES|!REQUEST:/__utm/|
    REQUEST_COOKIES_NAMES|ARGS_NAMES|ARGS|XML:/* "
    wordpress_ultra_simple_paypal_shopping_cart" "phase:2,
    block,msg:'WordPress injection.'severity:'2',id
    :'15004'"
```

Experiment 2: Method

- We collected tweets which are related WordPress and chose one tweet.



- Extracted information from the tweet and generated signature.

Experiment 2: Result

- We got the following attributes from the text.
  - Application name: WordPress
  - Vulnerability: XSS
  - Version Information: 4.7.5 and earlier
  - CVE-ID: CVE-2017-9061
- We generated the following signature

```
1 SecRule REQUEST_COOKIES|!REQUEST:/__utm/|
    REQUEST_COOKIES_NAMES|ARGS_NAMES|ARGS|XML:/* "
    wordpress" "phase:2,brock,msg:'WordPress XSS.'severity
    :'2',id:'15001'"
```

- From the result, it was confirmed that WAF signatures could be generated from collected tweets.

Experiment 3: Method (1/4)

- we extracted and filtered vulnerability information from the tweets.
- Through twitter API We collected tweets every day for the following period. After eliminating the duplicates of the collected tweets, we further processed 1,116 tweets.
  - Collection period: December 4, 2019 – January 8, 2020
  - Search query: Following three queries
    - WordPress AND Vulnerability
    - WordPress AND XSS
    - WordPress AND injection

Experiment 3: Method (2/4)

- To check the results of the filter, we manually assigned the following labels to the tweets based on the relevance to the WordPress vulnerability.
  - 0: WordPress vulnerability information
  - 1: other information

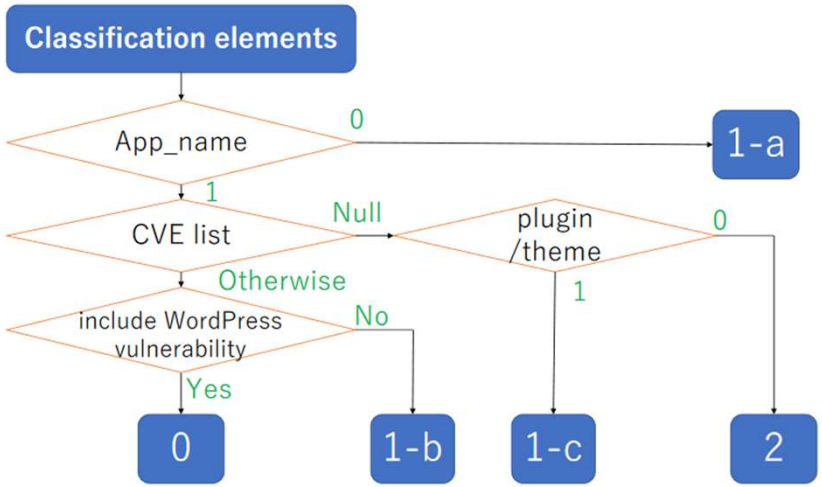
- As a result of the manual labeling, 76 tweets regarding WordPress vulnerabilities have been identified. The remaining 1,040 tweets were mistakenly extracted since its URL referred to the Webpage created using WordPress, for example.

Experiment 3: Method (3/4)

- We filtered tweets on its text context by pattern matching using regular expressions to the tweet's text and web page body which is indicated by URL.
  - The following attributes are extracted from the tweet and stored in JSON format.
    - ID: tweet-id
    - App_name: If the string includes "wordpress" this element is 1, otherwise 0.
    - CVE: Extract the string "CVE-¥d{4}-¥d+" and store it as a list.
    - plugin / theme: If the string includes "plugin" or "theme" this parameter is 1, otherwise 0.
  - These attributes are used to assign an estimated label to tweet

Experiment 3: Method (4/4)

- We automatically assigned an estimated label to tweets according to the flowchart.
  - Each label indicates the following categorization result.
    - 0: WordPress vulnerability information
    - 1-a: Not included the string “wordpress”
    - 1-b: Expected as a WordPress plugin or theme
    - 1-c: No WordPress vulnerability in CVE List
    - 2: Unfiltered by this method



Experiment 3: Result

- This method filtered 597 tweets and 98.5% of them were filtered correctly. On the other hand, 519 were unfiltered.
  - By using the pattern matching approach, the number of objects of analysis could be reduced by half.
- Out of 7 tweets that have been failed to be correctly filtered, 6 tweets were supposed to be labeled as 0 whereas it was mistakenly labeled as 1-b.
  - They contained information about WordPress and its plugins at the same time. Therefore, the filter misjudged them.

		estimated label					
		0	1-a	1-b	1-c	2	total
correct label	0	13	0	6	0	57	76
	1	1	94	292	191	462	1,040
	total	14	94	298	191	519	1,116

Consideration of Additional Sources

- Currently, the real-time information source is only Twitter and it could be prone to be disinformation.

- So, we discussed the possibility of using other information sources.
  - Stack Overflow
  - Reddit
  - teratail
  - Security StackExchange

Explore 1: Serious vulnerability (1/2)

- We explored discussions about following three WordPress vulnerabilities in those communities.
  - CVE-2018-20148 Published: December 14, 2018
  - CVE-2019-17669 Published: October 17, 2019
  - CVE-2019-20041 Published: December 27, 2019
    - These vulnerabilities have a high Common Vulnerability Scoring System (CVSS) score
- We searched the vulnerabilities by Google search.
  - Query: “WordPress” and “vulnerability”
  - Duration: one month before and after the vulnerability announcement

Exploration 1: Serious vulnerability (2/2)

- Results

	2018-20148	2019-17669	2019-20041
Stack Overflow	10(-)	6(-)	10(-)
Reddit	2(-)	15(-)	11(-)
teratail	6(-)	6(-)	3(-)
Security StackExchange	2(-)	0(-)	2(-)

- We could not obtain information about these vulnerabilities from these communities in a timely manner.
- Since the results are not suitable for a comparison of each knowledge community, we tried additional exploration.

Exploration 2: Number of search hits (1/2)

- We compared the number of search hits per site using the Custom Search API provided by Google to see the number of discussions about vulnerabilities in each knowledge community.
  - Query: “vulnerability”
  - Duration: 2017/01/01 – 2019/12/31
- Results

knowledge community\Year	2017	2018	2019	total
Stack Overflow	2,166	2,072	1,837	6,075
Reddit[cybersecurity]	31	172	266	469
Reddit[security]	16	60	151	227
teratail	241	196	172	609
Security StackExchange	1,166	1,072	768	4,006

- From the results, Stack Overflow and Security StackExchange can be used as good information sources.

Conclusion

- In this study, we proposed the WAF signature generation system using real-time information on the Internet.
- We conducted three types of experiments as initial studies.
  - We were able to narrow down the required data to half of the total data.
  - We discovered the following challenges in those experiments.
    - How to clearly distinguish vulnerability information of other Web applications which may have a similar name as Web application name
    - How to select the necessary information effectively from vulnerability information

Future Works

- The following challenges may occur when implementing the whole proposed system.
  - How to determine the disinformation
  - What to do with vulnerability information for which version information could not be extracted
  - Block legitimate Web applications which include the name of the target application in their names
- In order to solve these challenges, we will consider and verify the following approaches.
  - Defining reliability based on the account which posted information to the information sources
  - Creating additional signatures to block the target only