Overview of the “Vulnerability Scanning AI” using the machine learning

2016.01.13

Isao Takaesu, Professional Service Div.

1. Introduction

In November I did a demonstration of Artificial Intelligence that detects vulnerability of Web applications in a cyber security workshop. This demonstration was surprisingly well received, so I decided to explain the overview of our AI, Spider AI Vulnerability Scanner or SAIVS, in this blog post.

Disclaimer: SAIVS uses machine learning algorithms, but I will not go into the details in this post. More information on the machine learning algorithms is in "6. Reference" section, however, so please skip ahead if you would like to read about it.

Alright, let’s begin the overview of SAIVS.

2. Goal

The goal of a vulnerability scanning AI is to perform the following:

- Scan vulnerabilities while crawling the pages of Web applications.
- Report detected vulnerabilities to the developers and site owners of Web applications.

Naturally, the AI must be equipped with the following three abilities:

1. It crawls all the pages of Web applications.
   AI must be able to crawl dynamic pages such as “login” and “create account” pages.
2. It performs various scanning.
   AI must detect vulnerabilities that can be mechanically detected. It also must detect vulnerabilities that require human intelligence to detect.
3. It reports scanning results.
   AI must summarize detected vulnerabilities and countermeasures in a report in a simple, clear manner for the developers of the Web applications.

To perform these functions, the AI needs to have almost the same intelligence level as humans. Among some researchers, AI is predicted to exceed human intelligence in 2045. If this prediction is correct, the completion of the vulnerability scanning AI will also be around 2045.

Nevertheless, I would say that we have reached a certain level with the current technology. We have developed a currently beta SAIVS with the following capabilities:

1. **It can crawl relatively simple Web applications.**
   SAIVS can crawl Web applications that include dynamic pages such as “login,” “create account” and “information search” pages.
2. It can mechanically scan for vulnerabilities.
   SAIVS can detect vulnerabilities that are defined by a pattern.
3. It can output a scanning report.
   SAIVS can output a text-based simple report. The report includes target URLs and

執筆者一覧 (Authors)

○ 寺田 健 (Takeshi Terada)
○ 阪本 達矢 (Tatsuya Sakamoto)
○ 国分 裕 (Yutaka Kokubu)
○ Alice
○ 山谷 晶英 (Akihide Yamatani)
○ 高江洲 勲 (Isao Takaesu)
○ 吉田 裕也 (Yuya Yoshida)
○ 井上 竜馬 (Ryoma Inoue)
○ 小澤 拓海 (Takumi Ozawa)
○ 米山 俊嗣 (Toshitsugu Yoneyama)
○ 諌山 貴由 (Takayoshi Isayama)
○ サイバーインテリジェンスグループ (Cyber Intelligence Group)
○ 廣田 一貴 (Kazuki Hirota)
○ 小河 哲之 (Satoshi Ogawa)

location of the detected vulnerabilities.

SAIVS can also perform the following human-like actions:

"SAIVS recognizes the type of the page. If it crawls the "login" page without having a login credential, it creates one in the "create account" page. After it logs in with the created credentials, it crawls the rest of the pages and scans for vulnerabilities. When it finishes all pages, it outputs a simple report."

Currently our AI uses machine learning algorithms in order to achieve one of the aforementioned capabilities: It can crawl relatively simple Web applications. This post will explain how this ability was made possible by the machine learning algorithms. I will not discuss 2 and 3 because they are not technologically unique.

3. Implementation of SAIVS

SAIVS mechanically scans Web applications while crawling. By the way, what is crawling in the first place? In a nutshell, it is a process in which one follows and browses the links in an HTML document on a Web application. At the first glance, this seems simple enough, but the actual process is surprisingly complicated. For example, what exactly happens if the HTML page you are trying to crawl contains the tag shown in Figure 1?

```
<a href="/cyclone/leaderboard">Leaders</a>
```

Figure 1 Example of ANCHOR tag element

You will access the URL value /cyclone/leaderboard of the anchor tag with the href attribute by using GET method, and you will move to the next page. This is very easy.

How easy then is the case of Figure 2?

```
<form accept-charset="UTF-8" action="/cyclone/sessions" method="post">
    <label for="email">Email</label>
    <input id="email" name="email" size="30" type="text" />
    <label for="password">Password</label>
    <input id="password" name="password" size="30" type="password" />
    <input class="btn btn-large btn-primary" name="commit" type="submit" value="Sign in
</form>
```

Figure 2 Example of FORM tag element

It is safe to say that you need to enter the optimal values in the input forms specified with the INPUT tags to move to the next page. So, you will need to enter the e-mail address and password in the corresponding input forms.

Also, when you see the words "sign in," you can guess this is a "login" form. This means you need to prepare a login credential in advance. If you don't have a credential yet, you will first need to register and create an account from "create account" page.

Moreover, even when you enter appropriate values in the input forms, you may see an error message. In this case, you will recognize that you have failed to move to the next page and that you will need to re-enter new values in the input forms.

This example shows that crawling requires a complex thought process. Therefore, SAIVS must also demonstrate at least the following thinking patterns.

- Recognize the page type
- Learn the optimal parameter values
- Recognize the success or failure of a page transition

Our SAIVS uses three machine learning algorithms and acquired the thinking patterns as shown in Table 1.

No	Thinking pattern	Machine learning algorithm
1	Recognize the page type	Naive Bayes
2	Learn the optimal parameter values	Multilayer Perceptron Q-Learning
3	Recognize the success or failure of a page transition	Naive Bayes

Table 1 Thinking patterns and machine learning algorithms

Brief descriptions of each machine learning algorithm are listed in Table 2.

No	Algorithm	Description
1	Naive Bayes	It is used for text classification. Using some key information that determines the characteristics of a text, it can automatically classify the text into several pre-defined categories. It is used in spam filters.
2	Multilayer Perceptron	It emulates the function of nerve cells of an organism. It models the complex relationships between input and output, and by repeating the learning process, it is possible to reach the optimal output for an input. It is used in image and handwriting recognitions.
3	Q-Learning	It learns the best action of an agent. Using a behavior evaluation value called Q-value, a behavior of the agent is evaluated in a given environment, and by evaluation the agent is able to learn the optimal action. It is used for robots to learn human walking motion.

Table 2 Descriptions of machine learning algorithms

Naive Bayes identifies the page type and page transition’s success/failure using the information on the page (Table 1: No.1 and No. 3). As for learning the optimal behavior (the optimal parameter value) for a page transition, the combination of a multilayer perceptron and Q-learning is used (Table 1: No. 2).

Next, I will explain how to implement each of the thinking patterns.

3.1. Accomplishing "Recognize the page type"

Figure 3 is a page of "Cyclone" in OWASP Broken Web Applications Project. When you look at the page, what do you think this is?

Most people will recognize this is a login page. This is because they look at the information characterizing the page type, such as the texts "Sign in," "Email," and "Password" on the page. Such recognition logic can be achieved by Naive Bayes.

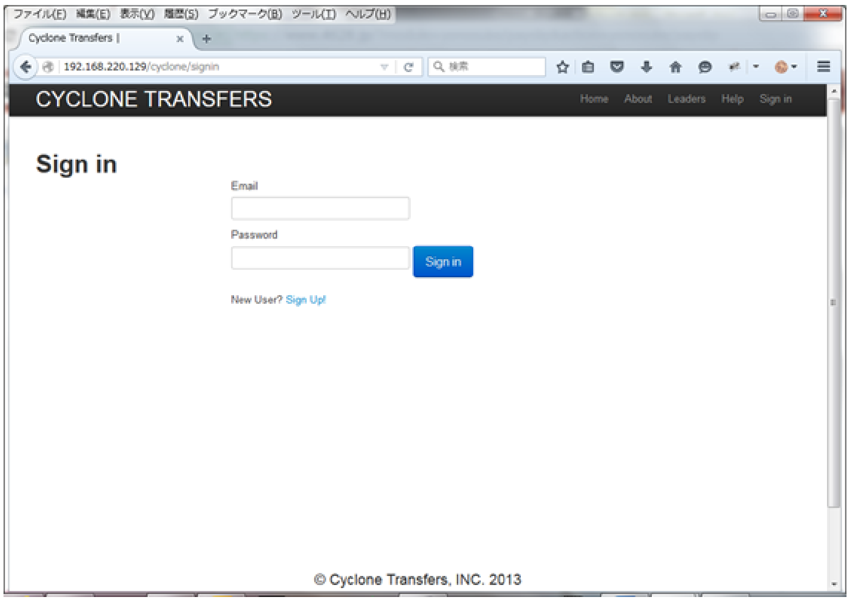

ファイル(F) 編集(E) 表示(V) 履歴(S) ブックマーク(B) ツール(T) ヘルプ(H)
Cyclone Transfers |
192.168.220.129/cyclone/signin
CYCLONE TRANSFERS
Home About Leaders Help Sign in
Sign in
Email
Password
Sign in
New User? Sign Up!
© Cyclone Transfers, INC. 2013

Figure 3 Login page

In order to identify the page type using Naive Bayes, I need to obtain the information that characterizes the page type. This information is available by parsing keywords such as "Signin," "Email," and "Password" in the HTML source of the page (Figure 4: written in red letters).

```
<h1>Sign in</h1>
<form accept-charset="UTF-8" action="/cyclone/sessions" method="post">
 . . .
<label for="session_email">Email</label>
<input id="session_email" name="session[email]" size="30" type="text" />
<label for="session_password">Password</label>
<input id="session_password" name="session[password]" size="30" type="password" />
 . . .
</form>
```

Figure 4 HTML source of the login page

Since I’m trying to identify the page type here, I will first define the categories for the pages as shown in Table 3. For the purpose of this demonstration, I’m only using three categories.

No	Category	Words used for classification
1	Login page	Email, User ID, ID, Password, PW, Pass, Pass Phrase, Sign in…
2	Create account page	User ID, ID, Email, Password, Password Confirm, Sign Up…
3	Search page	Word, Text, String, Search…

Table 3 Category table for the recognition of the page type

I will recognize the page type based on the information in Figure 4, the three keywords, "Sign in," "Email," and "Password." If the probabilities of each category containing one of the keywords are as shown in Table 4 below, the probability of each category containing all three keywords are calculated as the following.

- ”Login” : “0.9×0.5×0.5×100=22.5”
- ”Create account” : “0.2×0.5×0.5×100=5”
- ”Search” : ”0.05×0.2×0.1×100=0.1”

Category	Keywords being contained		
	Sign in	Email	Password
Login page	90%	50%	50%
Create account page	20%	50%	50%

Search page	5%	20%	10%

Table 4 Probabilities of the keywords appearing in each category.

The probability of category "login page" containing all three keywords is the highest. Therefore, the page that contains the texts "Sign in,” Email,” and “Password" is determined as the "login page."

This way I can accomplish the goal "Recognize the page type" using Naive Bayes.

By the way, Naive Bayes disregards the arrangements and correlation of the words (bag of words). Because of this, it can perform classifying and learning at a high speed. When words that do not exist in the category table appear, the "zero-frequency problem” occurs, but we have solved this problem by using "Laplace smoothing" at this time.

3.2. Accomplishing "Recognize the success or failure of a page transition"

Some of you may have realized already, but this ability also uses Naive Bayes. Figure 5 shows a failed login page. Most people will recognize that they have failed to login when they see the text "Invalid email or password" that characterizes a failed page transition.

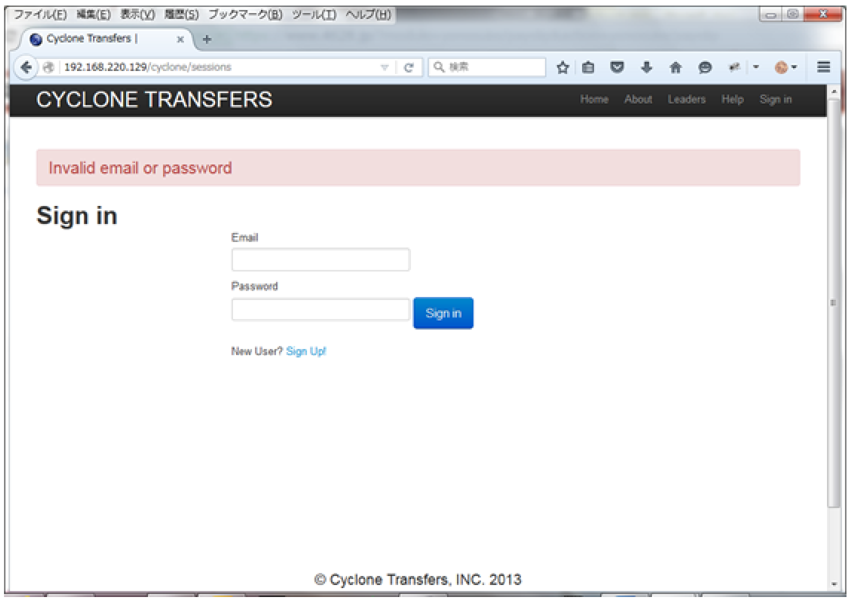

ファイル(F) 編集(E) 表示(V) 履歴(S) ブックマーク(B) ツール(T) ヘルプ(H)
Cyclone Transfers |
192.168.220.129/cyclone/sessions
CYCLONE TRANSFERS
Home About Leaders Help Sign in
Invalid email or password
Sign in
Email
Password
Sign in
New User? Sign Up!
© Cyclone Transfers, INC. 2013

Figure 5 Login failure

When you compare the before and after images of the page transition, you can find the following error message displayed on the HTML source.

```
<div class="alert lead alert-error">Invalid email or password </div>
```

Figure 6 HTML source of a failed login.

By using the texts that characterize the results of a page transition from the HTML source, Naive Bayes can classify the page transitions into "failure" or "success." This way I can accomplish the "Recognize the success or failure of a page transition.” I will not go over this logic because it is the same as 3.1 "Recognize the page type."

3.3. Accomplishing "Learn the optimal parameter values"

Figure 7 is a model that combines a multilayer perceptron and Q-learning and learns the optimal parameter values.

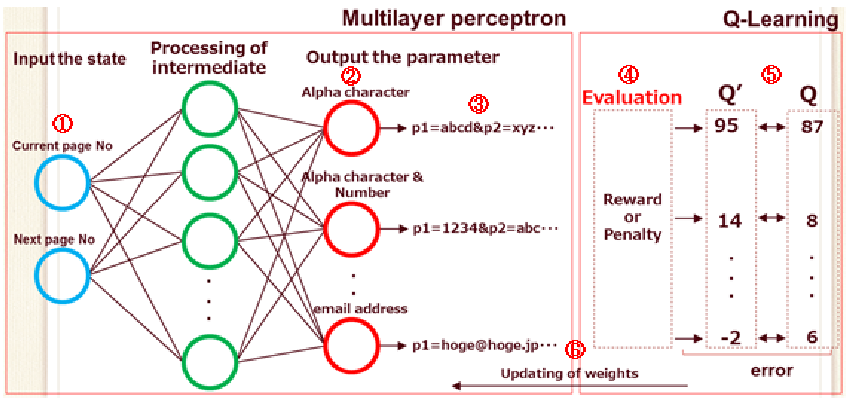


Figure 7 Learning model of the optimal parameter values

In this model, SAIVS learns the optimal behavior patterns (optimal parameter configuration) to transition to the next page by repeating the following steps.

1. Input the current state.
   (Current page No. : Source page, Next page No. : Destination page.)
2. Output some parameter values.
3. Use the output parameter values to transition to the next page.
4. Observe the success or failure of the page transition.(*1)
5. Update the Q-value. (*2)
   Calculate error for the Q-value before and after the update.
6. Let the multilayer perceptron learn, so that the error is minimized.

*1 For the recognition logic of page transitions, see Section 3.2.

*2 Transition is successful: Increase the Q-value.
Transition is failed: Reduce the Q-value.

At the initial stage when this model has not learned yet, it outputs random parameter values, causing SAIVS to fail almost all page transitions and be rarely successful. The model will continue to learn using the results of the "failed" and "successful" transitions and gradually start to output better parameter values by repeating the learning process.

Thus, although the initial learning has a high likelihood of failing page transitions, in the final stage of learning the success rate increases. After the learning process is completed, SAIVS is successful in almost all page transitions. As a side note, the model’s learning time is about 5 minutes per page transition, but we can shorten the time by adjusting the parameters used for the learning process.

SAIVS can obtain the optimal parameter values for input forms by using a model that combines a multilayer perceptron and Q-learning.

In the next section, I will demonstrate the actual work of SAIVS.

4. Demonstration

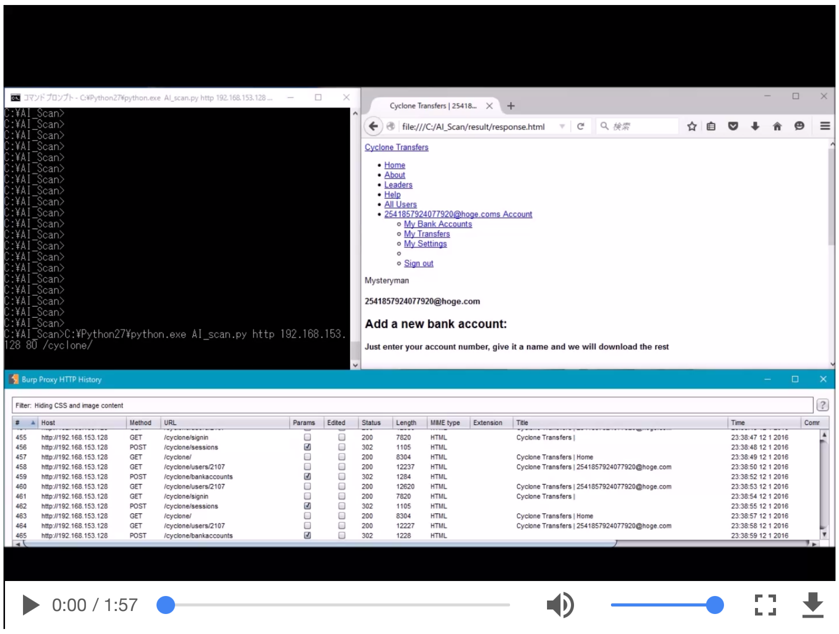
Overview of the demonstration is detailed in Table 5.

Target Web Apps	OWASP Broken Web Applications - Cyclone
Target page	Login : http://192.168.153.128/cyclone/sessions
	Create account : http://192.168.153.128/cyclone/users
	Search : http://192.168.153.128/cyclone/search
Signature	One type of SQL injection. (Insertion of single quotation marks)

Table 5 Overview of the demonstration.

The target Web application is "Cyclone" of OWASP Broken Web Applications. This Web application contains dynamic pages such as “login” page, “create account” page, and “search” page. In order to transition the pages after login, SAIVS needs to create an account in the “create account” page.

In order to shorten the time of the demonstration, I have limited the target to 3 pages. For the same reason, the type of signature is limited to one.



```
コマンド プロンプト - C:¥Python27¥python.exe AI_scan.py http 192.168.153.128 ...
C:¥AI_Scan>
C:¥AI_Scan>
C:¥AI_Scan>
C:¥AI_Scan>
C:¥AI_Scan>
C:¥AI_Scan>
C:¥AI_Scan>
C:¥AI_Scan>
C:¥AI_Scan>
C:¥AI_Scan>
C:¥AI_Scan>
C:¥AI_Scan>
C:¥AI_Scan>
C:¥AI_Scan>
C:¥AI_Scan>
C:¥AI_Scan>
C:¥AI_Scan>
C:¥AI_Scan>
C:¥AI_Scan>
C:¥AI_Scan>
C:¥AI_Scan>C:¥Python27¥python.exe AI_scan.py http 192.168.153.
128 80 /cyclone/
```

Cyclone Transfers | 25418...

file:///C:/AI_Scan/result/response.html

Cyclone Transfers

- Home
- About
- Leaders
- Help
- All Users
- 2541857924077920@hoge.coms Account
  - My Bank Accounts
  - My Transfers
  - My Settings
  -
  - Sign out

Mysteryman

2541857924077920@hoge.com

Add a new bank account:

Just enter your account number, give it a name and we will download the rest

Burp Proxy HTTP History

Filter: Hiding CSS and image content

#	Host	Method	URL	Params	Edited	Status	Length	MIME type	Extension	Title	Time	Comr
455	http://192.168.153.128	GET	/cyclone/signin	☐	☐	200	7820	HTML		Cyclone Transfers |	23:38:47 12 1 2016	
456	http://192.168.153.128	POST	/cyclone/sessions	✓	☐	302	1105	HTML			23:38:48 12 1 2016	
457	http://192.168.153.128	GET	/cyclone/	☐	☐	200	8304	HTML		Cyclone Transfers | Home	23:38:49 12 1 2016	
458	http://192.168.153.128	GET	/cyclone/users/2107	☐	☐	200	12237	HTML		Cyclone Transfers | 2541857924077920@hoge.com	23:38:50 12 1 2016	
459	http://192.168.153.128	POST	/cyclone/bankaccounts	✓	☐	302	1284	HTML			23:38:52 12 1 2016	
460	http://192.168.153.128	GET	/cyclone/users/2107	☐	☐	200	12620	HTML		Cyclone Transfers | 2541857924077920@hoge.com	23:38:53 12 1 2016	
461	http://192.168.153.128	GET	/cyclone/signin	☐	☐	200	7820	HTML		Cyclone Transfers |	23:38:54 12 1 2016	
462	http://192.168.153.128	POST	/cyclone/sessions	✓	☐	302	1105	HTML			23:38:55 12 1 2016	
463	http://192.168.153.128	GET	/cyclone/	☐	☐	200	8304	HTML		Cyclone Transfers | Home	23:38:57 12 1 2016	
464	http://192.168.153.128	GET	/cyclone/users/2107	☐	☐	200	12227	HTML		Cyclone Transfers | 2541857924077920@hoge.com	23:38:58 12 1 2016	
465	http://192.168.153.128	POST	/cyclone/bankaccounts	✓	☐	302	1228	HTML			23:38:59 12 1 2016	

Movie 1 Demonstration of vulnerability scanning

The command prompt at the upper left corner of the video is the console that is running SAIVS. The browser (Firefox) at the upper-right corner renders the page responses to show SAIVS transitioning. "Burp Proxy HTTP History" at the bottom is the access log of SAIVS.

Table 6 is a description of the video demonstration.

Time	Time
0:00	Scanning starts.
0:04	SAIVS skips examining the “Login” page and scans “Create account” page first. (SAIVS does not have a login account.)
0:36	Scanning of "Search" page starts.
0:55	"I found a vulnerability!!" is displayed on the console. Also, SQL error screen is displayed on the browser. These indicate that SAIVS detected an SQL injection vulnerability.
0:58	Scanning of "Login" page starts.
1:29	Scanning ends.
1:42	SAIVS creates a simple report. "Target List" is the list of URLs SAIVS crawled. The detected vulnerability is shown at "Scan Result." This report indicates that SAIVS has detected an SQL injection vulnerability in the search parameter in the "Search" page.

Table 6 Description of the demonstration

As you can see, SAIVS scans Cyclone while crawling it and detected an SQL injection vulnerability in the "Search" page after logging in.

5. Future Prospects

In the future, we will enhance the scanning and page transition abilities to scan various Web applications, and we hope to use SAIVS in Web application vulnerability assessments and bug bounty programs someday. Our goals are:

- Strengthening the page transitioning capability
  - Crawl more complex Web applications
- Strengthening the scanning capability
  - Add more signatures
  - Detect vulnerabilities that require human perception
  - Reduce false detections
- Applying the technology to business
  - Use in actual Web application assessments
  - Participate in bug bounty programs

AI has been gaining a lot of attention since the introduction of Deep Learning to the scene. I believe this trend is not temporary because AI has already spread to industries beyond academic fields and is now used for drones, automated vehicles and controlling robots. In the information security field, AI is used for technologies such as the attack detection in WAF, malware detection in anti-virus software and early detection of APT.

Machine learning will be a norm and used more widely in the near future. If you are interested, I recommend that you start playing with a machine learning algorithm now.

6. Reference

[1] Building Machine Learning Systems with Python/Willi Richert, Luis Pedro Coelho/O’Reilly/2014.10

[2] Evolutionary Computation and deep learning/ IBA Hitoshi/Ohmsha/2015.10

[3] Deep learning/OKATANI Takayuki/KODANSHA/2015.4.7

[4] Neural network and deep learning

[5] Neural network

[6] [Machine learning] The commentary while try the deep learning framework Chainer..

ページトップへ

サイトマップ

サービス

情報漏えい調査
セキュリティ診断
・Webアプリケーション診断
・スマートフォンアプリケーション診断
・組込機器診断
・RIA診断
・ネットワーク診断
・ソーシャルゲーム診断
・標的型攻撃耐性診断
マルウェア解析
・マルウェア解析
・マルウェア感染予兆検知診断
セキュリティ教育
・セキュアwebアプリケーション開発教育
・インフラセキュリティ教育
・セキュアスマートフォンアプリケーション開発教育
AWS™セキュリティ監視 secured by Alert Logic

セキュリティ監視
・WAF監視
・不正アクセス監視
・ログ統合監視
・MBSD Secure Web Gateway
セキュリティ・コンサルティング
・セキュリティ・インテリジェンス
・Splunk導入支援
・ITコンサルティング
・情報セキュリティ推進組織支援
・CSIRT構築支援
・事業継続
・リスクアセスメント
・内部統制
・認証取得支援
・ネットワークトラフィック攻撃解析
シンクタンク・システムズエンジニアリング

セキュリティnews
MBSD Security Insight
イベント＆セミナー
お知らせ
お問い合わせ
メディア掲載
導入事例
採用情報
MBSDについて
・社長のご挨拶
・企業理念
・会社概要（map）
・会社沿革

所在地

本店:
〒103-0013
東京都中央区日本橋人形町
1丁目14番8号　郵船水天宮前ビル6階
Mapはコチラ
TEL：03-5649-1961（代表）

赤坂オフィス:
〒107-0052
東京都港区赤坂2丁目17番7号
赤坂溜池タワー9階
Mapはコチラ
TEL：03-5575-2171

三井物産セキュアディレクション株式会社

Deceptionテクノロジー (illusive networks)