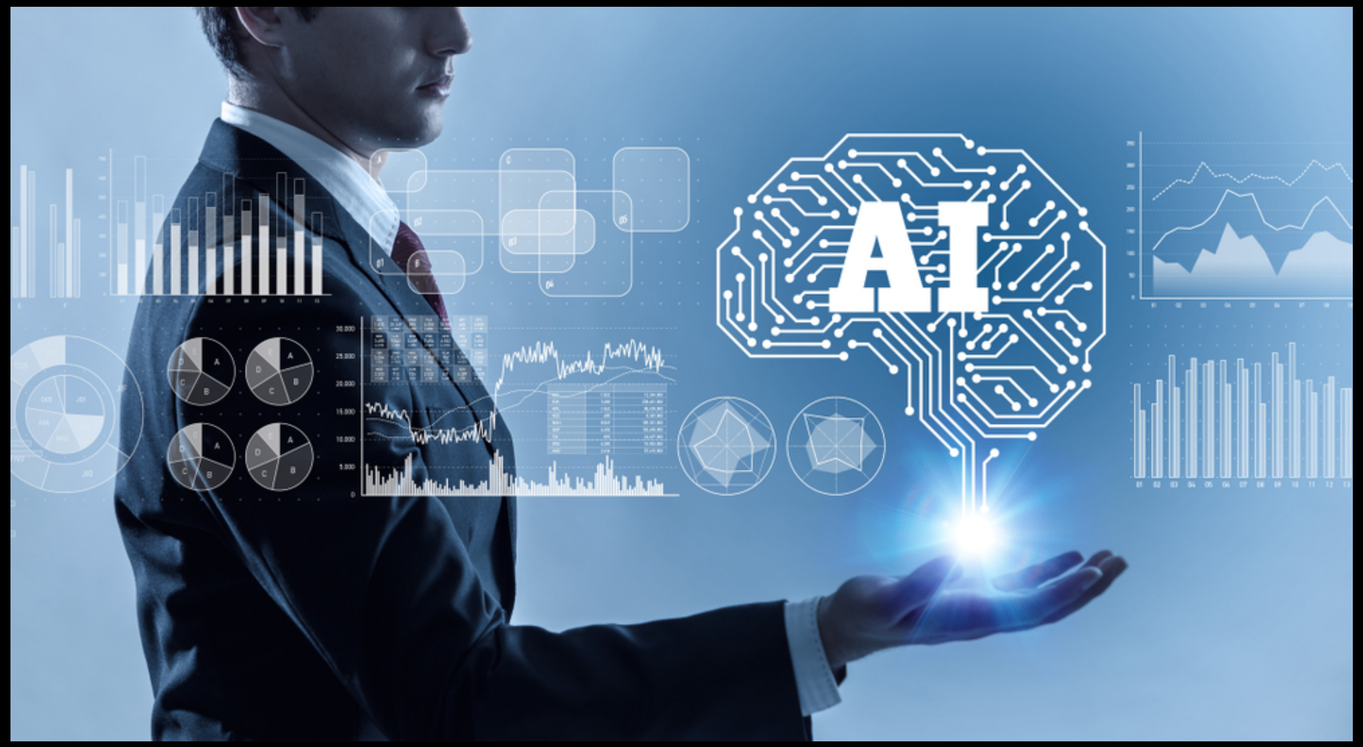
IA en la seguridad informática



Autor: Manuel Marco Sanchez
Linkedin: https://www.linkedin.com/in/mmarcosanchez/

#finger

- Profesional de la Seguridad Informática, experto en hacking ético, linux, blockchain, Inteligencia Artificial, Telecomunicaciones , Voip, redes, programación, administración de servidores, asterisk.

  Soy especialista en desarrollo blockchain , ICO, aplicaciones y módulos de Seguridad Informática, Inteligencia Artificial,Base de datos , Sistemas, CRM, Aplicaciones Vozip para Asterisk ,FreePBX,Elastix,Issabel, así como software de control de call center.

Brechas de Seguridad
en
La Inteligencia Artificial

Seguridad en tus APIs: ¿a qué vulnerabilidades se pueden enfrentar tus APIs?

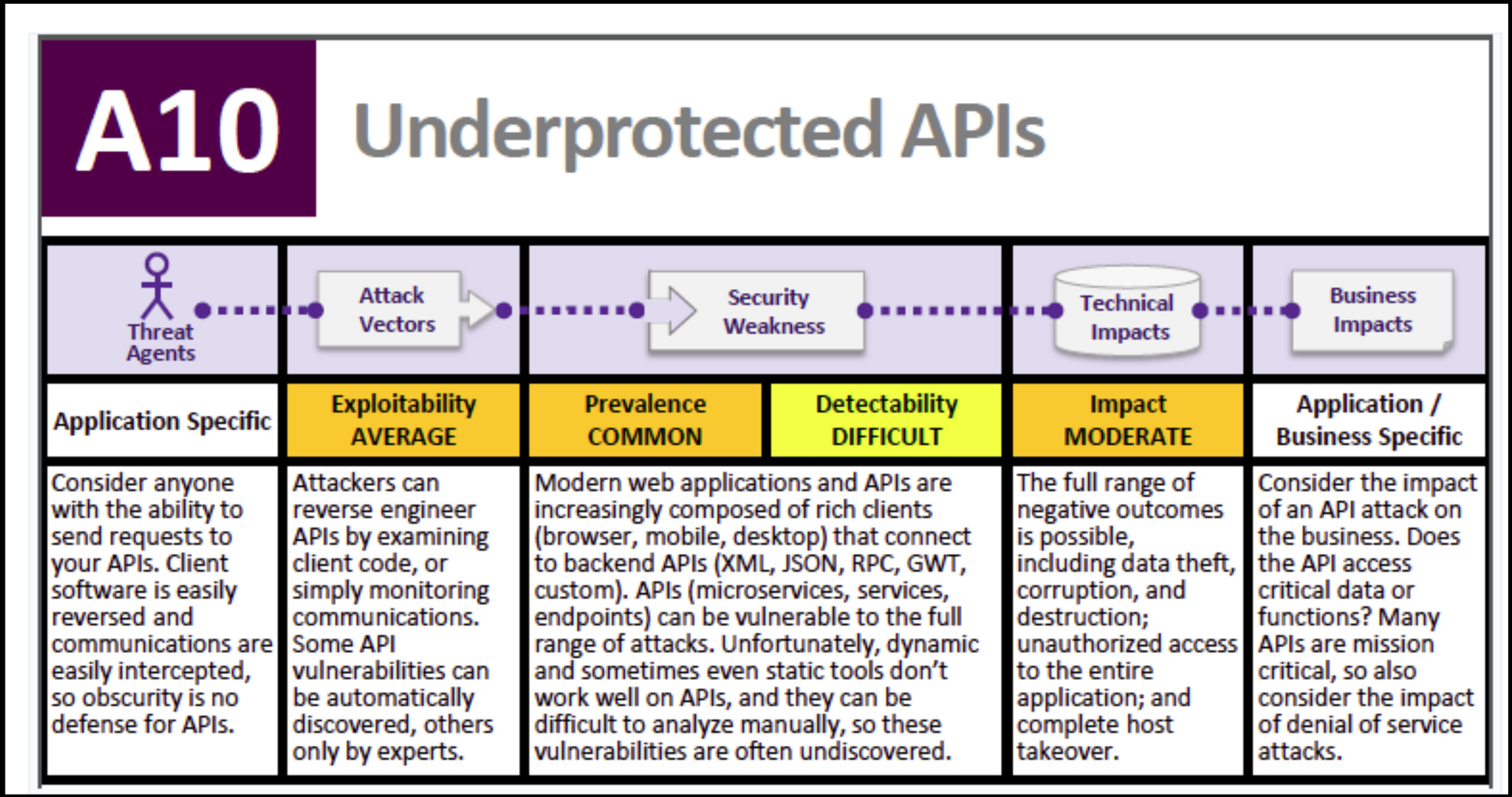
- El uso de APIs y de herramientas para su gestión son una tendencia cada vez más demandada a la hora de usar los modelos de IA.

estudio de Akamai

- El 83% de todo el tráfico web va a través de APIs (Fuente Estadio Akamai del 2019).
- Segun la empresa consultora Gatnert en el año 2022 los abusos derivados de las APIs serán el vector de ataque más frecuente.

En el top 10 OWASP en su versión inicial de 2017 ya lo incluían como amenaza.

Las APIs son el punto de entrada inicial y la superficie de ataque es mayor debido a la mayor demanda de conectividad B2B.



Te Gustaría que tu empresa apareciera

en el Listado del 2020

Top 10 Biggest Breaches in 2018

1. API: Aadhar — 1.1 billion
2. Marriott Starwood hotels — 500 million
3. Exactis — 340 million
4. MyFitnessPal — 150 million
5. Quora — 100 million
6. MyHeritage — 92 million
7. Cambridge Analytica — 87 million
8. API: Google+ — 52.5 million
9. Chegg — 40 million
10. API: Facebook — 29 million

Como Securizar una API

Eres Programador Aunque No sepas Como Se puede Hacer.

- OAuth 2.0 (Autorización abierta)
    - OAuth 2.0 es un método de autorización utilizado por compañías como Google, Facebook, Twitter, Amazon, Microsoft, etc. Su propósito es permitir a otros proveedores, servicios o aplicaciones, el acceso a la información sin facilitar directamente las credenciales de los usuarios. Pero tranquilo, únicamente accederán bajo la confirmación del usuario, validando la información a la que se le autorizara acceder.

Una vez aprobada la autorización, esta aplicación de terceros podrá acceder a la información permitida mediante una autentificación con un token de acceso.

PHP

```php
<?php
define('OAUTH2_CLIENT_ID', '');
define('OAUTH2_CLIENT_SECRET', '');

$authorizeURL = 'https://github.com/login/oauth/authorize';
$tokenURL = 'https://github.com/login/oauth/access_token';
$apiURLBase = 'https://api.github.com/';

session_start();

// Start the login process by sending the user to Github's authorization page
if(get('action') == 'login') {
// Generate a random hash and store in the session for security
$_SESSION['state'] = hash('sha256', microtime(TRUE).rand().$_SERVER['REMOTE_ADDR']);
unset($_SESSION['access_token']);

$params = array(
'client_id' => OAUTH2_CLIENT_ID,
'redirect_uri' => 'http://' . $_SERVER['SERVER_NAME'] . $_SERVER['PHP_SELF'],
'scope' => 'user',
'state' => $_SESSION['state']
);

// Redirect the user to Github's authorization page
header('Location: ' . $authorizeURL . '?' . http_build_query($params));
die();
}

// When Github redirects the user back here, there will be a "code" and "state" parameter in the query string
if(get('code')) {
// Verify the state matches our stored state
if(!get('state') || $_SESSION['state'] != get('state')) {
header('Location: ' . $_SERVER['PHP_SELF']);
die();
}
```

```
// Exchange the auth code for a token
$token = apiRequest($tokenURL, array(
'client_id' => OAUTH2_CLIENT_ID,
'client_secret' => OAUTH2_CLIENT_SECRET,
'redirect_uri' => 'http://' . $_SERVER['SERVER_NAME'] . $_SERVER['PHP_SELF'],
'state' => $_SESSION['state'],
'code' => get('code')
));
$_SESSION['access_token'] = $token->access_token;

header('Location: ' . $_SERVER['PHP_SELF']);
}

if(session('access_token')) {
$user = apiRequest($apiURLBase . 'user');

echo '<h3>Logged In</h3>';
echo '<h4>' . $user->name . '</h4>';
echo '<pre>';
print_r($user);
echo '</pre>';

} else {
echo '<h3>Not logged in</h3>';
echo '<p><a href="?action=login">Log In</a></p>';
}



function apiRequest($url, $post=FALSE, $headers=array()) {
$ch = curl_init($url);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, TRUE);
```

```
if($post)
curl_setopt($ch, CURLOPT_POSTFIELDS, http_build_query($post));

$headers[] = 'Accept: application/json';

if(session('access_token'))
$headers[] = 'Authorization: Bearer ' . session('access_token');

curl_setopt($ch, CURLOPT_HTTPHEADER, $headers);

$response = curl_exec($ch);
return json_decode($response);
}

function get($key, $default=NULL) {
return array_key_exists($key, $_GET) ? $_GET[$key] : $default;
}

function session($key, $default=NULL) {
return array_key_exists($key, $_SESSION) ? $_SESSION[$key] : $default;
}
```

https://gist.github.com/aaronpk/3612742

Puedes usar las bibliotecas cliente a continuación para implementar OAuth 2.0 en tu aplicación. Recomendamos usar una biblioteca de cliente en lugar de escribir tu propio código. El uso de estas bibliotecas de cliente estándar es importante para la seguridad y la seguridad de los usuarios y tu aplicación.

- Biblioteca cliente de API de Google para Java
- Biblioteca cliente de API de Google para JavaScript
- Biblioteca cliente de API de Google para Python
- Biblioteca cliente de API de Google para .NET
- Biblioteca cliente de API de Google para Ruby
- Biblioteca cliente de API de Google para PHP
- Biblioteca de OAuth 2.0 para Google Web Toolkit
- Controladores de OAuth 2.0 para Google Toolbox para Mac

También puedes seguir las instrucciones en la sección Invocación de YouTube Data API para modificar el código y así configurar correctamente los valores de token de OAuth 2.0.

Uso de técnicas de IA en Seguridad Informática

- Una de las técnicas fundamentales de la Inteligencia Artificial aplicadas a áreas sensibles de la Seguridad Informática son los sistemas detectores de intrusos (IDS). La detección de intrusos se ha convertido en parte integral de los procesos de seguridad de la información desde que ellos pueden implementar y administrar controles identificados de la seguridad de la información. Son variadas las técnicas de Inteligencia Artificial las que se han aplicado en estos sistemas informáticos, en todos los casos se busca la optimización y detección más eficaz de intrusiones. La Inteligencia Artificial puede reducir el esfuerzo humano requerido para construir Sistemas Detectores de Intrusos y puede mejorar su rendimiento.

- Existen otras técnicas de Inteligencia Artificial aplicada a los IDS como son las Máquinas de Vectores de Soporte (SVMs), Redes Neuronales Artificiales (ANNs), Regresión Multivariada Adaptativa utilizando Splines (MARS) y Programas Genéticos Lineales (LGPs). Se establecen comparaciones críticas basadas en experimentos realizados de parámetros esenciales en los IDS en cuanto a la precisión, tiempo de formación y tiempo de prueba de los mismos, utilizando las técnicas de Inteligencia Artificial anteriormente descritas. Existen resultados que permiten realizar un análisis en torno a la efectividad en la aplicación de las técnicas que se estudiaron:
    - LGPs superó a MARS, SVMs y ANNs en términos de detección precisa a expensas del tiempo.
    - MARS fue superior a SVMs con respecto a la clasificación de las clases más importantes (acceso no autorizado a privilegios de superusuario y acceso no autorizado a máquinas remotas) en términos de severidad del ataque

- SVMs superó a ANNs en importantes aspectos de escalabilidad (SVMs puede trabajar con un gran número de patrones, mientras que ANNs toma un gran tiempo en lograr o fallar en cuanto a convergencia si el número de patrones tiende a crecer); SVMs funciona en orden de magnitud más rápido.

Codigo y Demos Reales

- Con el crecimiento del aprendizaje automático en los últimos años, se están realizando muchas tareas con la ayuda de algoritmos de aprendizaje automático.

- Si pudiéramos detectar una URL maliciosa desde una URL no maliciosa utilizando algún algoritmo de aprendizaje automático.

- Entonces, se reunió alrededor de 400,000 URL de las cuales alrededor de 80,000 eran maliciosas y otras estaban limpias.

Análisis

Utilizaremos la regresión logística, ya que es rápida. La primera parte fue tokenizar las URL. Se cae un funciona propia de tokenizador para esto ya que las URL no son como otro texto de documento.

```
def getTokens(input):
    tokensBySlash = str(input.encode('utf-8')).split('/')   #get tokens after splitting by slash
    allTokens = []
    for i in tokensBySlash:
        tokens = str(i).split('-')  #get tokens after splitting by dash
        tokensByDot = []
        for j in range(0,len(tokens)):
            tempTokens = str(tokens[j]).split('.')  #get tokens after splitting by dot
            tokensByDot = tokensByDot + tempTokens
        allTokens = allTokens + tokens + tokensByDot
    allTokens = list(set(allTokens))    #remove redundant tokens
    if 'com' in allTokens:
        allTokens.remove('com') #removing .com since it occurs a lot of times and it should not be included in our features
return allTokens
```

- El siguiente paso es cargar los datos y almacenarlos en una lista.

```
allurls = 'C:\\Users\\Faizan Ahmad\\Desktop\\Url Classification Project\\Data to Use\\allurls.txt'  #path to our all urls file
allurlscsv = pd.read_csv(allurls,',',error_bad_lines=False) #reading file
allurlsdata = pd.DataFrame(allurlscsv)  #converting to a dataframe

allurlsdata = np.array(allurlsdata) #converting it into an array
random.shuffle(allurlsdata) #shuffling
```

- Ahora que tenemos los datos en nuestra lista, tenemos que vectorizar nuestras URL. Utilizo los puntajes de tf-idf en lugar de usar la clasificación de bolsa de palabras, ya que hay palabras en las URL que son más importantes que otras palabras, por ejemplo, 'virus', '.exe', '. Dat', etc. Permite convertir las URL en una forma vectorial .

```
y = [d[1] for d in allurlsdata] #all labels
corpus = [d[0] for d in allurlsdata]    #all urls corresponding to a label (either good or bad)
vectorizer = TfidfVectorizer(tokenizer=getTokens)   #get a vector for each url but use our customized tokenizer
X = vectorizer.fit_transform(corpus) #get the X vector
```

- Tenemos los vectores Convirtámoslo ahora en datos de prueba y capacitación y veamos cómo realizar una regresión logística en él.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)   #split into training and testing set

lgs = LogisticRegression()  #using logistic regression
lgs.fit(X_train, y_train)
print(lgs.score(X_test, y_test)) #pring the score. It comes out to be 98%
```

- Obtenemos una precisión del 98%. Es un valor muy alto para que una máquina pueda detectar una URL maliciosa .

```
X_predict = ['wikipedia.com','google.com/search=faizanahad','pakistanifacebookforever.com/getpassword.php/','www.radsport-voggel
X_predict = vectorizer.transform(X_predict)
y_Predict = lgs.predict(X_predict)
print y_Predict #printing predicted values
```

- `Los resultados fueron:`

stanifacebookforever.com/getpassword.php/ **(Bad Url)**
www.radsport-voggel.de/wp-admin/includes/log.exe **(Bad Url)**
ahrenhei.without-transfer.ru/nethost.exe **(Bad Url)**
www.itidea.it/centroesteticosothys/img/_notes/gum.exe **(Bad Url)**
El codigo lo tenéis disponible en **Github**.

Ataques adversarios a una red Neuronal

- Un Ataque Adversario es una técnica que usa para engañar **(hackear)** una red neuronal y da un resultado completamente diferente al que se visualiza en la imágen. Por ejemplo, le presentamos una imagen de un edificio y nos dirá que es un perro. En algunos casos podrá reemplazar la indentidad de una persona o para los coches de conducción autónoma podrá confundir señales de tránsito. Ésto se explica a que el output de una red neuronal se expresa como una propobabilidad y la imagen modificada ofrece una certeza de resultado mucho mayor que las otras imágenes por lo que la red optar por la más alta.
- Veamos el código en acción: aqui

Fuentes para esta presentación y enlaces de interes:

https://www.redeszone.net/2019/05/28/cibercrimen-inteligencia-artificial/

https://sensedia.com/es/apis/10-riesgos-seguridad-apis/

https://www.paradigmadigital.com/dev/seguridad-apis-vulnerabilidades/

https://sensedia.com/es/apis/10-riesgos-seguridad-apis/

https://enmilocalfunciona.io/construyendo-una-web-api-rest-segura-con-json-web-token-en-net-parte-i/

https://hackingprofessional.github.io/Security/Aprende-que-es-Deserializacion-Insegura-OWASP-VII/

https://www.paradigmadigital.com/dev/seguridad-apis-como-evitar-ataques-seguridad/

https://www.youtube.com/watch?v=td-2rN4PgRw

https://www.youtube.com/watch?v=51MuGvJTuQc

https://blog.bigbrainta.com/uso-de-tecnicas-de-ia-en-seguridad-informatica/

https://www.ibm.com/es-es/security/artificial-intelligence?p1=Search&p4=43700052451760830&p5=b&cm_mmc=Search_Google-_-1S_1S-_-EP_ES-_-%2Bia%20%2Bseguridad_b&cm_mmca7=71700000065051640&cm_mmca8=aud-328707907826:kwd-890105591409&cm_mmca9=Cj0KCQjw4dr0BRCxARIsAKUNjWQ6ttNBKdvEmwbOi86CgYy0s61KFh9oNsgJY4vPbRiO0ZfPKcmqOzgaAvGkEALw_wcB&cm_mmca10=426097856836&cm_mmca11=b&gclid=Cj0KCQjw4dr0BRCxARIsAKUNjWQ6ttNBKdvEmwbOi86CgYy0s61KFh9oNsgJY4vPbRiO0ZfPKcmqOzgaAvGkEALw_wcB&gclsrc=aw.ds

https://tecnohotelnews.com/2020/02/05/2020-medidas-seguridad-ciberataques/

https://expansion.mx/tecnologia/2020/02/27/los-ciberataques-se-vuelven-un-asunto-de-inteligencia-artificial

https://docs.microsoft.com/es-es/azure/machine-learning/studio/execute-python-scripts

MUCHAS GRACIAS

La información contenida y comentada se usa con fines educativos
El uso de la misma para fines no lícitos, no es mi responsabilidad.