Discussing Secure Input Solutions for Web Applications

Angelo P. E. Rosiello
angelo@rosiello.org
• Angelo P. E. Rosiello received the B.S. and M.S. degrees in Computer Science Engineering cum laude from “Politecnico di Milano” in 2004 and 2006, respectively.

• Previously Angelo worked for Accenture in the Security Strategy Service Line and collaborates with Prof. Christopher Kruegel and Prof. Engin Kirda (Technical University of Vienna) in the ICT security field.

• Angelo works for “The European House – Ambrosetti” in the management consulting field. He owns the ITIL Service Management Certification and he is a specialist of IT Strategy&Governance.

• Master in Marketing & Communication Management.
1. Brief introduction to client-side information theft
2. Strategic defense techniques
3. Some recent secure input solutions
4. Conclusions
Statistics from the Anti Phishing Working Group (AWPG) confirm the global nature of crimeware code.

- Breakdown of the websites hosting malicious code -

APWG reports over 170 different types of keyloggers distributed on thousands of websites!
In a typical attack, the aim of the attacker is to take control of the user’s web client in order to manipulate the client’s interaction with the web application (attacking the integrity of the information).

It is possible to distinguish three main phases of a typical attack:

1. The attacker installs the malware in the machine of the victim
2. The attacker controls the business logic of the victim’s web application
3. The attacker manipulates sensitive information to realize a fraud
The attack is not complex, it starts from a simple e-mail sent to the victims and ends with a real fraud.

Send e-mails

E-mails containing a link to a bad malware are sent to thousands of potential victims.

Install the malware

Statistically someone will click the link provided in the e-mails. If the loaded webpage exploits a vulnerability in the browser (e.g. a parsing vulnerability) the malware will silently be installed.

Manipulate information

The malware automatically detects money transfers and silently modifies the target account number.

Fraud

Money is gone!

Remember that when the client has been compromised even security protocols, such as SSL, are completely useless!
Transaction generators (TGs) let the attacker execute “ordinary” transactions.

- Transaction Generator Code for Firefox -

```xml
<?xml version="1.0"?>
<overlay xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul">
<script>
document.getElementById("appcontent").addEventListener("load", function() {
  var currentLocation = getBrowser().selectedBrowser.contentDocument.location;
  if(currentLocation.href.indexOf("www.retailer.com/loggedin") > 0) {
    var xhr = new XMLHttpRequest();
    xhr.open("POST", "https://www.retailer.com/buy");
    xhr.send("item=blender&quantity=10&address=Kansas");
  }
}, true);
</script> </overlay>
```

The TG issues a purchase request to www.retailer.com/buy and orders ten blenders to be sent to some address in Kansas.
1. Brief introduction to client-side information theft

2. **Strategic defense techniques**

3. Some recent secure input solutions

4. Conclusions
Defense solutions can be distinguished in software and hardware based techniques.

Focus of this presentation!
During the last years many software-based solutions were proposed but none is considered as a reference standard, yet.

<table>
<thead>
<tr>
<th>Software-based Solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PwdHash</strong></td>
</tr>
<tr>
<td>PwdHash is a browser extension that <strong>transparently converts a user's password into a domain-specific password</strong>. PwdHash automatically replaces the contents of these password fields with a one-way hash of the pair &lt;password, domain-name&gt;.</td>
</tr>
<tr>
<td><strong>Spoof Guard</strong></td>
</tr>
<tr>
<td>SpoofGuard looks like a toolbar, after it is installed. When a user enters a username and password on a spoofed site that contains some combination of suspicious URLs, misleading domain name, images from an honest site, and a username and password that have previously been used at an honest site, SpoofGuard <strong>will intercept the post and warn the user with a pop-up that stops the attack</strong>.</td>
</tr>
<tr>
<td><strong>Spyblock</strong></td>
</tr>
<tr>
<td>Spyblock aims to protect user passwords against network sniffing and dictionary attacks. It proposes to use a combination of password-authenticated key exchange and SSL. Furthermore, as additional defense against pharming, cookie sniffing, and session hijacking, it proposes a form of transaction confirmation over an authenticated channel. The tool is distributed as a client-side system that consists of a browser extension and an authentication agent that runs in a virtual machine protected environment.</td>
</tr>
<tr>
<td><strong>Virtual Keyboards</strong></td>
</tr>
<tr>
<td>A <strong>virtual keyboard</strong> is a program which <strong>simulates a physical keyboard</strong> and provides some degree of protection against keystroke loggers. To evade this defense mechanism attackers coded new advanced logging softwares which take screenshots of where the mouse pointer is to determine what number was clicked.</td>
</tr>
<tr>
<td><strong>New Authentication Techniques</strong></td>
</tr>
<tr>
<td>New techniques of authentication are under research, such as <strong>using an image during the registration phase</strong> which is shown during every login process</td>
</tr>
</tbody>
</table>
1. Brief introduction to client-side information theft

2. Strategic defense techniques

3. Some recent secure input solutions

4. Conclusions
A Transaction authentication number or TAN is used (typically by banks) as a form of single use passwords (OTP) to authorize transactions.

**TAN Generation**
- A set of unique TANs for the user are generated
- Every TAN can be used only once
- Some banks generate so called indexed TAN (i-TAN) schemes, where the bank server requests a specific i-TAN for each transaction

**Log-on**
- To log on to his/her account, the user must enter user name and password. This may give access to account information but the ability to process transactions is disabled

**Execute Transaction**
- To perform a transaction, the user enters the request and "signs" the transaction by entering an unused TAN

When the bank requests a certain TAN, malicious code can replace the user’s input without invalidating this transaction number!
TAN and i-TAN are vulnerable to integrity attacks, this is why the information should be bind to confirmation tokens.

- **Client side** -
  1. Specify $f(.)$ that only the user knows or can compute.
  2. Send sensitive information “x” to the server (e.g. the bank).
  3. Send $t=f(x)$.

- **Server side** -
  - Receives “x” and $t=f(x)$.
  - Verify that $t=f(x)$.

- Verification: $\langle x, t \rangle$.
Two schemes were proposed in practice to compute $f(.)$ using both a code book.*

**- Description -**

- The code book contains a collection of simple algorithms that can be used by users to manually compute confirmation tokens
- The **server** (e.g. the bank) will randomly choose an algorithm from the user’s code book to let him/her execute the transaction
- The **user will apply the algorithm** to his/her input executing the transaction

**- Token Calculation Scheme -**

- **Token ID 4:**
  - Create a number using the 5th and 7th digits of the target account and add 542 to it.
- **Token ID 5:**
  - Create a number using the 3rd and 5th digits of the target account and add 262 to it.
- **Token ID 6:**
  - Multiply the 4th and 8th digits of the target account and add 17 to the result.
- **Token ID 7:**
  - Create a number using the 3rd, 6th and 7th digits of the target account.

*) Authors: Szydlowski, Kruegel, Kirda.*
Two schemes were proposed in practice to compute \( f(\cdot) \) using both a code book*.

### - Description -

- The code book will consist of a large number of random tokens that are organized in pages.
- The server side (e.g. the bank) and the user previously and secretly agree on which digits of the account number are relevant for choosing the correct page.
- The bank then requests the user to confirm a transaction by asking him/her to enter the value of a specific token on that page.

### - Token Lookup Scheme -

<table>
<thead>
<tr>
<th>User: 980,243,276</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bank: ID = 20</td>
</tr>
<tr>
<td>Page = 82</td>
</tr>
<tr>
<td>20: F2342U</td>
</tr>
<tr>
<td>21: 47B1S3</td>
</tr>
<tr>
<td>22: 5M2F33</td>
</tr>
</tbody>
</table>

*) Authors: Szydlowski, Kruegel, Kirda.
The idea of this solution* is to extend graphical input with CAPTCHAs.

- Description -

- Generate a graphical input field with randomly placed CAPTCHA characters
- The customer uses the mouse to click on the area that corresponds to the first character that should be sent
- Clicking on the image generates a web request that contains the coordinates on the image where the user has clicked with the mouse
- After the first character is transmitted, the web application generates another image with a different placement of the characters, and the process is repeated
- Since the CAPTCHA characters cannot be identified automatically by machines, a malware program has no way to know which information was selected by the user to corrupt its integrity

*) Authors: Szydlowski, Kruegel, Kirda.
As for every IT attack, web application attacks can be prevented, detected and mitigated through server-based and client-based approaches, supported by education and awareness.

Client-based techniques try to protect users implementing local solutions, software and hardware-based ones.

Server based techniques are applied on servers or providers that offer services to customers.
Questions & Answers

Questions?

yes

Answer Known?

no

State that time has run out!

yes

Answer

no

Thank Audience

Leave...


