Journal of Basic and Applied Engineering Research
Print ISSN: 2350-0077; Online ISSN: 2350-0255; Volume 1, Number 6; October, 2014 pp. 67-72

http://www.krishisanskriti.org/jbaer.html

Command Injection Threats in Mobile Applications

Jyothy Joseph

Al-Ameen College Edathala, Aluva

Abstract: **Mobile devices have very huge involvement in daily activates. Many of them are relaying mobile applications for their day to day activities like banking, shopping, communication, etc. But all of them are not having good idea on its security threats. Command injection is one of serious vulnerability involved in many of unsecured mobile applications. Many of them downloading and using multiple mobile applications from unsecured sources. Command injection threat utilizing the secured information in unauthorized way and making financial and data loses. This paper trying to list out various types of command injections involved in mobile applications and advisable precautionary measures.**

1. INTRODUCTION

Nowadays mobile devices offer an extreme level of convenience. It became a mandatory object in many of them day to day life. Extended and convenient features of mobile operating systems bring great acceptance of mobile devices in all categories of people. In the communication media, mobile phones are keep maintain its first position. Now communication methodologies are growing in different level. Many of them uses different type of applications for phone calls like Face time, Skype, Tango, Viber, Lion, etc. Few applications like Magic Jack provide contact numbers, which can be reachable anywhere in the world through internet. Previous days SMS were commonly used for mobile messaging. Now the trends are switched to different mobile applications like WhatsApp, Kik, WeChat, Wrap-up, etc. Previously text messages were commonly used but now changed to multimedia messages. Social media like Facebook, twitter, etc are doing a key role in communication. Many of mobile device users are using social media applications also.

As grows the convenient usage of mobile applications, the security threats also grows. Command injection is one of the security threat faces in mobile applications and websites. This threat aims to execute arbitrary commands on the host operating system through a vulnerable application or website. This kind of attacks is possible when the application or website passes user supplied data without proper security measures. The hacker passes the malicious operating system level commands or database level commands to access the secured information in unauthorized way. Normally hackers are using the cookies, session, forms, HTTP headers, etc. for passing the crooked commands.

Using the cracked applications or mobile sites, the hackers are collecting the secured information like bank account credentials, personal account credentials, personal identification details etc. These types of critical information theft might leads to financial loses and other issues. In mobile operating systems, the Command injection majorly comes in picture in two ways

- Mobile Apps
- Mobile Web apps

Both mobile Apps and Web apps are accessed on mobile devices. A Mobile website is very much similar as other websites, it consist of HTML based pages which can accessed through a Wi-Fi and 2G/3G/4G networks. Many of the websites are specifically designed to access from mobile; it is giving very good mobile view. Separate style sheets are maintained for mobile site. Mobile sites are additionally provide few features like click to call (dial to the display number), location based mapping, etc.

Mobile apps very much similar are other desktop software which can be installed in the mobile devices. Google play store, iTunes App sore, Blackberry App World, etc. are providing thousands of mobile applications in different categories. Mobile apps also called as native apps. The apps might be pull the content and data from the internet, in similar fashion to a website, or it may download the content, so that it can be accessed without an Internet connection.

How to decide the mobile app or mobile web app is the best one? That will be depends on the goal and the user convenience. Normally a mobile website should be the first step in developing a mobile web presence, whereas a mobile app is useful for developing an application for a very specific purpose that might be difficult to achieve through via a web browser. Latest HTML technologies are giving a great view to mobile applications.

Many of mobile applications are available in market from different sources. But many of the users are not having much idea on its reliability. Improper application developments lead to command injection threats.

2. RESEARCH STRUCTURE

Many methodologies are available to define the workflow of a research topic. This study followed the analytical methodology as mentioned in below structure (Fig 1). Here analysis the role of command injection in mobile applications and website. Also list out the various types of command injections threats noticed in mobile.

This study tries to suggest the best practices to come out from command injection threats.

Fig 1. Research Methodology

2.1 Define the research Structure: Here defining the study boundary and the flow. This study mainly focused the command injection threats in mobile applications and mobile websites. This also explains different types of mobile command injections and its precautionary measures.

2.2 Analysis mobile Apps and mobile Sites: Android and iOS mobile operating systems are providing exiting features to smart phone. Recent year's, numbers of smart phone users are drastically increased. A new mobile application rapidly changes the shopping modes. It is providing new beneficial services to change the user shopping experience. New mobile apps provide verity options to compare the products before purchase. The apps notify the new deals and features to users on timely manner. The new mobile application trends give great advertising platform to many of the enterprises for notifying their services to millions of consumers. The mobile apps facilitate the different types of online shopping options. Google play store and iOS App store are currently offer thousands of mobile applications in different categories. When installing and using many of applications, they are collecting user details which might include secured data like bank details, SSN number, etc. Most of the applications give a confirmation to consumers on its security part. But many of the consumers are not giving enough attention when providing their secured information. The growth of mobile technology, combined with the prevalence of smart mobile devices, it is changing the people shopping experience. Mobile applications offer new beneficial services designed to enhance the consumer shopping challenges. These applications can allow consumers to use their mobile devices in stores to compare prices against the different sellers and pay for purchases.

Similar ways the new mobile devices giving new customer experience. It also keeps opening the different types of threats. The hackers are utilizing the advantages of command injection and trying to make financial benefits. Many of the users are not much familiar about the security features required for a mobile application.

Earlier month of this year, FTC (Federal Trade Commission, USA) released a note on the security failure of two mobile applications. *Fandango and Credit Karma* mobile applications failed to secure the transmission of millions of consumers' sensitive personal information from their mobile apps. The applications disabled a critical default process, known as SSL (Secure Sockets Layer) certificate validation, which would have verified that the applications communications were secure. By overriding the default validation process, Fandango undermined the security of ticket purchases made through its iOS app, exposing consumers' credit card details, including card number, security code, zip code, and expiration date, as well as consumers' email addresses and passwords. Similarly, Credit Karma's apps for iOS and Android disabled the default validation process, exposing consumers' Social Security Numbers, names, dates of birth, home addresses, phone numbers, email addresses and passwords, credit scores, and other credit report details such as account names and balances. Due to that those applications become vulnerable and it opens a way to hackers, to interact the information sent and receive through the application.

When any of applications disabled the SSL secure validation and using the unsecured public Wi-Fi networks which are available in coffee shops, airports, shopping malls, restaurants, etc. causing command injection threats. Later two companies have agreed to settle FTC (Federal Trade Commission) charges that they misrepresented the security of their mobile apps and failed to secure the transmission of millions of consumers' sensitive personal information from their mobile apps. To help secure sensitive transactions, mobile operating systems, including iOS and Android, provides application developers with tools to implement an industry standard known as Secure Sockets Layer (SSL). If properly implemented, SSL certificates secures an app's communications and ensures that an attacker cannot intercept

the sensitive personal information when a consumer submits through the application.

2.3 List out Various types of command Injections: As normal website, mobile applications also impacting commend Injection. The commands are passing in the form of SQLs, XMLs, OS level commands, etc. Below are the major types of injections noticed in mobiles devices

1. SQLite Injection
2. Cross-site Scripting
    - JavaScript Injection
    - HTML script injection
3. Local File Inclusion
4. XML Injection
5. Format String Injection
6. Intent Injection
7. OS command Injection

2.3.1 SQLite Injection: SQLite is a light weight relational database, commonly used in mobile operating systems like iOS, Android, Blackberry, Symbian OS, etc. It is small, compact, and self-contained database. SQLite Mobile client provides the ability to synchronize the data in SQLite databases with applications. As SQL injection impact the web applications, it also impacts the mobile applications too. This type of injection mainly comes due to inefficient coding standards. Most of the applications have an option to enter user inputs. Hackers are using this place to push additional commands to run against the database. This might cause pulling secured information out or it might insert unsecured data into database. This issue is coming due to improper input validation before issue the database requests.

Normal application design gives a login page to enter the user credentials and get it logged into the application. User entered the user details as below and clicked the submit button

If the code is placed as below to fetch the details from database, the SQLite injection will play here

```
$cn = new SQLiteDatabase('filename');
$uname = <Jyothy' OR 1=1 -->;
$pwd = <any value>;
$result= @$cn->query(

    "SELECT * FROM user
     WHERE username='{$uname}'
     AND password = '{$pwd}'");
```

The below query perform with database

SELECT * from USER

WHERE username = 'Jyothy' OR 1=1
-- AND password = 'any value'

This query will allow the user to get inside the application with any value. This query gets injected below two major commands to exclude the proper validation

1. Added *OR 1=1* , this criteria always gives true
2. Added --,this will exclude the rest of the condition

When do the coding, needs to provide proper user validation to filter out the injection commands. The below code structure provide an additional validation function *specialchar()* to takeout the illegal inputs as below code

```
$cn = new SQLiteDatabase('filename');
$uname = <Jyothy' OR 1=1 -->;
$pwd = <any value>;
if (specialcahr())
{

$name = sqlite_escape_string($name);
$pwd = sqlite_escape_string($pwd);
}
$result= @$cn->query(
"SELECT * FROM user

 WHERE username='{$uname}'

 AND password = '{$pwd}'");
```

2.3.2 Cross-site scripting Injection: Crosse site scripting (XSS) in one of common application layer web injecting method. This type vulnerabilities target the client side scripting. XSS is manipulating the client side script to execute exactly similar as the system designed. The manipulated script is embedded with the current page and executed as normal. These types of injections normally target the users not the application.

How the Cross site script is working, below example (Fig 2) explains the Cross site scripting injection.

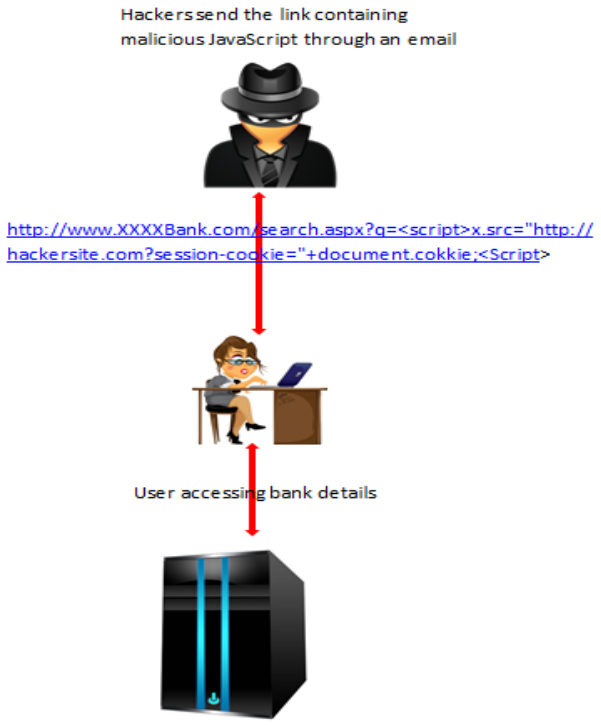

Fig 2: XSS scripting injection

Here unknowinly the user directing to bank account through a link clicked from an email send by a hacker. Here the hacker injected a command to pulling the user details from cokkies. Cross site scriping might infacted through two ways

1. JavaScript Injection
2. HTML script injection

2.3.2.1 Java Script injection: JavaScript is a commonly used technology in websites and web related applications. This is not only using for the good purposes, but also for malicious purposes as well. Many of mobile apps especially Android apps compromise the security and privacy in many of applications. JavaScript injection pushes some additional java scripts in between a normal web call and it causing to loose the secure data. A JavaScript injection allows hacker to do multiple things; below points are the major threats

1. Read the cookie values
2. Update the cookie values
3. Update web forms
4. Redirect to invalid forms or sites
5. Inject tracking or exploit codes

Below mention an example code for a JavaScript injection. Here, when the user click the display buttons it direct the user to hacker's site.

```
<script>
<button onclick=”location.href=’http://
        www.hackersite.com’”id=”1″                value=
”1″/>DisplayButton
</button>
</script>
```

2.3.2 .2 HTML Injection: This issue normally comes when the application is not handled the user supplied data properly. The hacker can supply a valid HTML and mislead the users to hacker control. This vulnerably occurs when the application is allowing inserting a block of HTML code through input parameter or a link. HTML5 is very much popular for mobile application development and many of them widely using this nowadays.

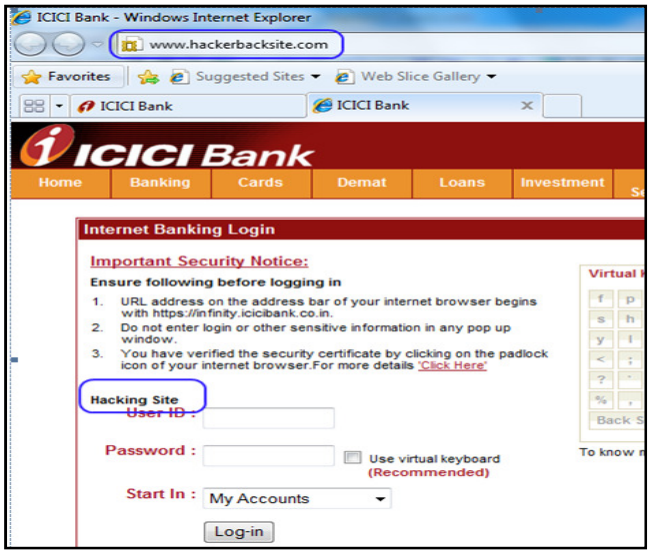

Fig 3 : HTML Injection

Above image(Fig 3) is an example for HTML injection. The haker imitate the Bank site through the HTML code. If the user given their credentils in ths page the hacker capture the user credentials.

2.3.3 Local File Inclusion : This vulerability is existing in many of sites. The hacker get allowed to include and additonal file along with the exisitng system. This issue occuring due to insuffient input field validation. The impact of this injection is, instead of normal application file the hacker giving a different work flow for their own crooked purposes. Below example helps to understand the scanario in better way. Here, instead of local file the hacker refer a remote file.

Main file

```
<?php
$ dir='image/';include_once('header.php');
```

Include file

```
<html>
```

```
<head>
<title>Test Site</title>
<style type=”text/css”>
@import url(“<?php echo $style_dir;>style.css”);
</head>
<body>
Hacker Call
http://sitename.tld/header.php?dir=http://hackSite.tld/css/
```

2.3.4 XML Injection: In this type of injection, the hackers are providing the crooked data in the, input filed in XML format. Which will override the other data and geting placed hackers details into secured place. Below example gives a better understanding

Input XML document format

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<employees>
<emp>
<empname>jyothy</empname>
<empid>123<empid/>
<email>jyothy@testmail.com</email>
</emp>
</employees>
```

Hackers input

```
empname        : alice
empid          : 234
email:
<email>joseph@testmail.com
</email> </emp><emp><empname>Hacker
</empname><empid>333</empid>
<email>hacker@testmail.com
</email>
```

Resulting XML document

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<employees>
<emp>
 <empname>jyothy</empname>
 <empid>123<empid/>
 <email>jyothy@testmail.com</email>
</emp>
<emp>
 <empname>joseph</empname>
 <empid>234</empid>
 <email>joseph@testmail.com</email> </emp>
<emp><empname>Hacker</empname><empid>
  3333</empid><email> hacker@hackermail.com
 </email>
</emp>
</employees>
```

In the above example hacker inserted hacker details into email field. In many cases the second details will override with first one. so the hacker details will get inserted into secured place(database). To prevent XML injection below measures will help

- Check the XML data have proper XML definition
- Validate the input values and make sure it having expeted format
- Include encoding methods

2.3.5 Format String Injection: In this type of injection, hackers are providing formatting characters or functions into a string input field. For example, in certain functions of the C programming languages such as printf, the formatting character *%s* will print the contents of a memory location expecting this location to identify a string. A hacker can read or write into that particular memory locations and can manipulate the values in unexpected ways. Improper access of memory locations might cause to crash the application itself. Below Objective C methods are vulnerable to format string attacks; this is causing to make issues in iOS based mobile devices

- NSLog
- NSString stringWithFormat
- NSString initWithFormat
- NSMutableString appendFormat
- NSAlert informativeTextWithFormat
- NSPredicate predicateWithFormat
- NSException format
- NSRunAlertPanel.

2.3.6 Intent Injection: Majorly Android mobile devices have the Intent Injection issue. When users provide the input data in dynamic way, a hacker can manipulate the code in the form of input data and execute. When dynamic data include in intent needs to provide proper validations. Below are the major intent methods causing this intent injection

- addCategory
- setAction
- setClass
- setClassName
- setComponent
- setData
- setDataAndType

2.3.7 OS Command Injection: This type of injection occurs when the hackers are executing the system level commands to

pull or push the secured data through vulnerable applications. This issue is maximum causing because of improper user input validation. The string input fields' needs to scrutinize with proper user input validation.

2.4 Impact through Mobile apps : The mobile application reliability depends on the different application design components like architecture, configuration, coding, data access mode, etc. The below image (Fig.4) giving a normal mobile application architecture. Few of mobile apps work independently in the mobile device itself. But most of the recent mobile apps are access internet services and getting details from different networks and different servers. The interaction to internet keeps open more opportunities to hackers to access the mobile devices and access the secured details. User input validation is most important point to keep away the command injection attackers. Avoiding shared memory usage also helps to reduce the hacking options.

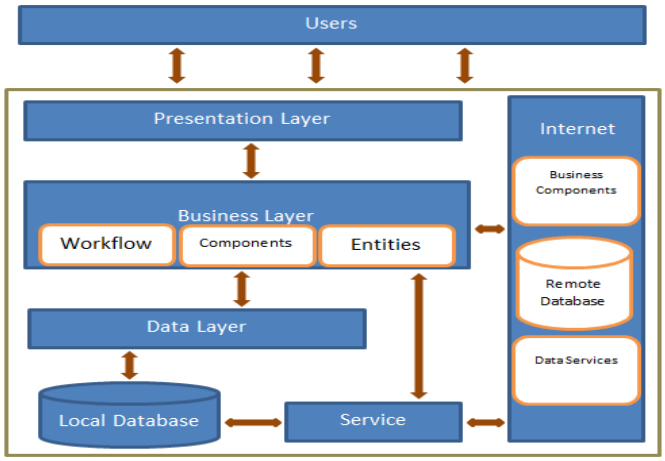

Fig 4. Mobile App Architecture

2.5 Impact through Mobile web apps : Recent days importance of mobile web applications are drastically increased. Many people are accessing multiple websites through mobile devices. Latest technologies like HTML5, jQuery,CSS3, etc. are facilitating very nice user interfaces. Many of new web frameworks are default providing mobile device compatibility. Currently many of business organizations are looking forward multi-channel web applications. As the mobile devices have very less place to give the display, sometimes the developers are excluding secured functionalities. This is giving an addition opportunity to hackers to get the secured data. Latest mobile web applications are trying very well to mimic the native applications characteristics. Latest frameworks like MVC, MVP, etc. are giving better convenience for mobile web application development.

Below image (Fig.5) giving a normal mobile web application architecture.

Fig 5. Mobile Web App Architecture

2.6 Conclusion : Mobile command injection attackes can be controlled based on porper precausionary measure. Below listout the important precausionary measures.

Precautionary measures

- User input Validation - None of UI inputs are execute without validation. Do the specific input validation based on input type and its length
- Script filters - Implement proper java script validation procedures
- SQL queries - Parameterized queries are alwayes advisable for avoiding SQLite injection
- Input file validation - Include operating system based input file validators like NSFileManager
- Format String - Avoid string execution, before validating format string injection. Make sure the validation for the string value getting from other services.
- String functions - Avoid or carefully use command injection reported old string functions like strcat, strcpy, strncat, strncpy, sprint, vsprint, etc.
- Infrastructure - Make sure industry suggested security methodizes like SSL certificate are properly using.
- Avoid unsecured network connections - Carefully connect the mobile devices with public network, possibility of command injection threats are high on public networks
- Application source - Download the mobile applications in secured and trustable sources

REFERENCES

[1] Web Application Security' https://www.owasp.org/
[2] Federal Trade Commission (FTC)' https://www.consumer.ftc.gov/
[3] 'XML Injection' http://projects.webappsec.org
[4] 'Format string injection' http://minsky.gsi.dit.upm.es
[5] 'Internet Injection' http://oasam.org