Hijacking Mobile Data Connections

Cristofaro Mune Roberto Gassira' Roberto Piccirillo

by *Mobile Security Lab*
{c.mune,r.gassira,r.piccirillo}@mseclab.com

April, 2008

Abstract

The use of data connections on mobile phones keeps growing at a steady rate. New services are continuously deployed and mobile phones keep getting smarter. Our work shows how an attacker may be able to take complete control of data connections originated by a mobile phone, by remotely re-configuring the handset connection parameters. This can be achieved by leveraging the standard provisioning mechanism, SMS source spoofing and social engineering techniques. The technique, fully described in this paper, can be applied for interception and modification of data sessions, including those that are normally confined into a Mobile Operator Network (data extrusion). As a consequence, it may be possible for an attacker to access session data, inject malicious payload and access internal Mobile Operator network resources.

1 Introduction

Accessing data by using Internet and its services is a primary need in today everyone's life. Accessing data everywhere and whenever desired is becoming a primary need as well. The widespread use of mobile phones, with their continuously improved capabilities, and the availability of newly deployed services, are steadily increasing the number of people making use of mobile data connections. The number of mobile users recently approached 4 billions, and the number of mobile broadband users has been counted in 100 millions [1].

Configuring a phone with parameters, needed for using data connections, can be performed by manually inserting the required settings. But, in order to make the configuration process easier, and to improve users mobility, standard protocols have been created for remotely provisioning the devices with the correct configurations. OMA Provisioning standard describes such process, that relies on WAP specification for building configurations to be sent via SMS. Tools able to send such messages are publicly available, and can be easily purchased at affordable costs.

SMS messages play a great role in the current mobile environment; they can carry simple text messages, or be a transport mechanism for more complex protocols and services. SMS source spoofing is one of the most renowned attack in mobile environment, but the ability to make an SMS appearing as originated by a source of choice may be useful also for business purposes. As an example, company XYZ may choose to send Christmas's messages that appear to users as originated from XYZ rather than a less comprehensible phone number.

Several services are currently available, directly over the Internet, that allow for sending of SMS with the desired content, and, also, to specify the source that will be displayed to the recipient.

In this context, by using knowledge of standard protocols and easily available tools and services, it is possible for an attacker to achieve complete hijacking of data connections origi-

nated by a victim mobile phone.

An attack scenario is described in the following sections, where all the needed elements are analyzed in an attacker's perspective.

2 Provisioning Mechanism

WAP provisioning framework specifies mechanisms to provide devices with connectivity and application access information. This framework allows one or more trusted points of configuration management to tune their respective configurations within Mobile Equipment (ME). The WAP provisioning mechanism includes the use of the WAP stack protocol as well as mechanisms such as WAP Push. The building of provisioning messages involves several WAP stack protocols.

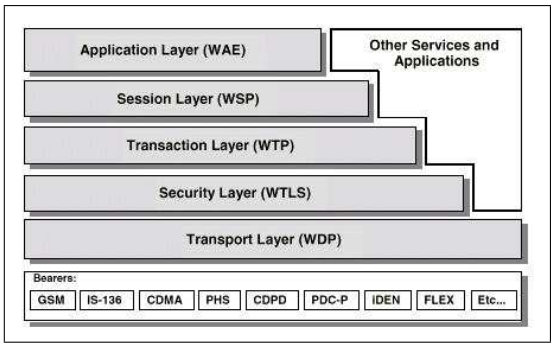

Figure 1: WAP stack protocol

Features provided by layers in the WAP stack protocol will be briefly described in the following:

1. **Bearers Layer :** provides a variety of different bearer services, including short message, circuit-switched data, and packet data
2. **Transport Layer :** offers a set of consistent services, such as UDP, WDP and TCP, to the upper layer protocols and maps those services to the available bearer services
3. **Security Layer :** The Security Service provides mainly privacy, data integrity and authentication services between two communication applications.
4. **Transaction Layer :** Wireless Transaction Protocol (WTP) runs on the top of datagram service and provides a light-weight transaction-oriented protocol that is suitable for implementation in "thin" clients (mobile stations)
5. **Session Layer :** The Wireless Session Protocol (WSP) provides the application layer of WAP with a consistent interface for two session services. The first is connection-oriented service that operates above the transaction layer protocol (WTP). The second is a connectionless service that operate above a secure or non-secure datagram service (WDP).
6. **Application Layer :** The Wireless Application Environment (WAE) is a general-purpose application environment to establish an inter-operable environment that will allow operators and service providers to build applications and services.

3 Building a provisioning message

A step by step approach will be followed in describing the building of a provisioning message. The involved protocols and languages involved are:

- WBXML media type provided by Wireless Application Environment (Application layer)
- Wireless Session Protocol feature (Session layer)
- Wireless Datagram Protocol feature (Transport layer)
- SMS service (Bearer layer)

A provisioning message for remotely adding a new Network Access Point to the mobile phone will be used as a working example.

Configurations are described by XML, as specified in the Provisioning Content Specification [8]:

```
<wap-provisioningdoc>
  <characteristic type="NAPDEF">
    <parm name="NAME" value="NewAPN" />
    <parm name="NAPID" value="NewAPN_NAPID_ME" />
```

```
    <parm name="BEARER" value="GSM-GPRS" />
    <parm name="NAP-ADDRESS" value="apn.new.com" />
    <parm name="NAP-ADDRTYPE" value="APN" />
    <parm name="DNS-ADDRESS" value="x.y.w.z" />
  </characteristic>
  <characteristic type="APPLICATION">
    <parm name="NAME" value="NewAPN" />
    <parm name="APPID" value="w2" />
    <parm name="TO-NAPID" value="NewAPN_NAPID_ME" />
  </characteristic>
</wap-provisioningdoc>
```

The **wap-provisioningdoc** element encapsulates all the provisioned information. The **characteristic** element is needed to group the provisioned information into logical units. It requires a **type** attribute to specify the kind of parameters to configure. This field can assume multiple values, such as PXLOGICAL, PXPHISICAL, PXAUTHINFO, etc. Since we want to configure a new Network Access Point we will use the **NAPDEF** value.

Each type of characteristic element has its own parameters that can be expressed through parm element. The parm element is typically formed by a name and value couple. In Network Access Point definition the following parm elements will be used:

- name=NAME
  value=NewAPN
  Indicates that the new Network Access Point will be called NewAPN
- name=NAPID
  value=NewAPN_NAPID_ME
  Used to link to the TO-NAPID parameter in other characteristic element, for example APPLICATION
- name=BEARER
  value=GSM-GPRS
  Indicates which network the definition is valid for
- name=NAP-ADDRESS
  value=apn.new.com
  Can assume different values, such as phone number of an access router, an SMSC address or, as in our case, DNS resolvable address (APN)
- name=NAP-ADDRTYPE
  value=APN
  Indicates the format of the address presents in NAP-ADDRESS
- name=DNS-ADDRESS
  value="x.y.w.z"
  It is the IP address of the DNS server to be user for resolving DNS names
- name=APPID"
  value="w2"
  This value links this configuration to browser activities

After the XML configuration message has been defined, it is necessary to encode it in a WAP Binary XML format, as specified in WAP Binary XML Content Format specification [9]. There are many free tools that allow for converting an XML file into WBXML.

WBXML message, then, has to be encapsulated into a WSP protocol data unit. A WSP protocol data unit, as described in Wireless Session Protocol Specification [5], is composed by:

- **TID** : (Transaction ID) used to associate requests with replies in the connectionless session service. It must be included in the connectionless PDUs
- **Type** : specifies PDU type and function
- **Type-Specific Contents** : message type specific information

The provisioning message has to be encoded as a Push message by using the Push primitive, defined in Wireless Session Protocol [5].

The Type-Specific Content for a Push primitive is made by:

- **HeadersLen** : length of the ContentType and Headers fields combined
- **ContentType** : contains the content type of data
- **Headers** : contains the push headers
- **Data** : contains the data pushed from the server to the client (our WBXML message)

According to the above information, the WSP protocol data unit for our provisioning message (hex format) is:

wsp_payload:
[e7 06 2f 1f 2d b6 91 81 92 43 42 38 46 44 45 45 46 34 36 43 37 30 46 36 38 34 36 35 43 45 35 37 30 33 43 37 34 42 38 34 36 35 38 35 36 35 42 32 34 00] [WBXML message]
where:

- **TID** = [e7]
- **Type** = [06]
- **Push-Specific Content** = [Remaining Part]

(Note: Headers are not used in such a message)

The Push-Specific Content complies to the Wireless Session Protocol Specification [5], that provides a specific encoding for lengths and parameters:

1. 0x2F reports the length of WSP header
2. 0x1f 0x2d is the length of remaining WSP header given as "Length-quote Length" as defined in WSP specification [5]
3. 0xb6 is the assigned number for the content type **application/vnd.wap.connectivity-wbxml**. Used to declare the message as a Provisioning message
4. 0x91 indicates that SEC parameter and security mechanism will be used
5. 0x81 indicates that USERPIN security mechanism is used
6. 0x92 indicates that MAC parameter is present
7. The next 40 bytes are the MAC value
8. 0x00 End-of-string for the encoded MAC value

Two types of security mechanisms are possible. The first one is called *"Security by means of an out-of-band delivery of a MAC authentication information"* that will not be explored in this paper. The second one is called *"Security by means of shared secret"*. In order to provide security by means of a shared secret, the provisioning message must include MAC and the security method SEC. The SEC parameter can take three different values:

1. **USERPIN**: the user PIN must be a string ASCII encoded decimal digits (i.e. octects with hexadecimal values 30 to 39)
2. **NETWPIN**: the shared secret is based on network specific shared secret
3. **USERNETWPIN** : the shared secret is a network specific shared secret appended with a user PIN

The shared secret will be used to calculate the MAC value in this manner: both, shared secret and provisioning document, encoded in WBXML are input, as key and data respectively, for the HMAC calculation, based on the SHA-1 algorithm, M=HMAC-SHA(K,A). The output message M is encoded as a string of hexadecimal digits where each pair of consecutive digits represent a byte. This hexadecimal encoded output is the MAC value included in the message. This calculation is repeated in the Mobile Equipment when checking the validity of the MAC.

With the USERPIN method, the shared secret is delivered to user in order to calculate the HMAC. The same procedure is applied when USERNETWPIN is used. With NETWPIN method it is not required to send a shared secret to the user because the secret is based on a network specific shared secret.

In order to send our message by SMS we must use the Wireless Datagram Protocol. For GSM SMS bearer the WDP headers structure is defined using User Data Header (UDH) framework as defined in [10].

The User Data Header of our provisioning (hex format) is:
udh_header : [0B 05 04 0B 84 23 F0 00 03 54 02 01]
The User Data Header can be composed of several SMS control functionalities. In our udh_header there are two SMS controls:

- udh_header [05 04 0B 84 23 F0]
  Application port addressing scheme, 16 bit address

- udh_header [00 03 54 02 01]
  Concatenated short message

The first byte, with 0x0B value is the length of whole udh_header. We use Application port addressing to achive a WAP Push connectionless session service:

- 0x05 : Application Port Addressing element identifier
- 0x04 : header data length
- 0x0B 0x84 : destination port 2948/udp
- 0x23 0xF0 : source port 9200/udp

The second SMS control allows to send concatenated sms:

- 0x00 : Concatenated SMS element identifier
- 0x03 : header data length
- 0x54 : Concatenated SMS reference number
- 0x02 : Maximum number of SMS in the concatenated SMS
- 0x01 : Sequence number of the current SMS

In order to complete the provisioning message, an SMS header specifying destination, use of UDH User Data Header presence is added:
SMS_header = [00 41 00 0C 91 xx xx xx xx xx xx 00 F5 8C]
xx xx xx xx xx xx represents the recipient mobile phone number.

4 Provisioning Process and Issues

Mobile operators usually provide handset configuration services to their customers relying on the provisioning mechanisms previously described. Most of them use the USERPIN security mechanism for message integrity verification and authentication. The PIN code required by the USERPIN mechanism is usually provided to the user by means of an Info SMS sent before the provisioning message.
To summarize, the provisioning process is carried out in the following steps:

1. An Info SMS with the PIN code is sent to the user
2. Next the provisioning message is sent
3. The handset requests the user to insert the PIN code to process the message
4. If the PIN code is correct, the new settings overview is shown
5. The user is asked for permission to install the new settings
6. Hopefully, the new settings are installed as the default one.

Test performed on several vendor mobile phones have revealed different issues and weaknesses concerning the provisioning process described above.

First of all, the trustworthiness of the Info SMS is based only on the message sender. This information can be easily spoofed, using for example, one of the many bulk SMS messaging services publicly available, for free or not.
Another issue is the visualization of the provisioning message source. Some mobile phones do not show the sender number but only the parameter NAME, defined in the provisioning message, or a generic sender name such as Service Provider or Message Configuration. However, if the real sender number is displayed, a user will likely accept the provisioning message after receiving an Info SMS.

The next issue regards the parameters installed by the provisioning message that are totally hidden to from the user. This issue may depend on the need for handset User Interfaces to be user-friendly. Displaying many technical details can confuse the user but at the same time, the complete lack of them may hide useful information or improper settings.

Finally, the ability of sending a provisioning message without restrictions can be considered as the most important issue. No filter mechanisms in place were discovered in most mobile operator networks and most mobile phones have no provisioning message filter capabilities.

5 Remote Reconfiguration

By exploiting the security issues identified in the provisioning process, an innovative attack methodology has been designed and tested in order to remotely reconfigure a mobile handset. This attack requires the target mobile phone to be equipped with a standard OMA provisioning client.

By using social engineering techniques, an attacker impersonating the mobile operator tries to persuade the targeted victim user to accept and install a provisioning message.

The attack starts by sending the victim an Info SMS carrying a PIN code. The SMS sender address can be spoofed with the same number or identifier usually used by the mobile operator for delivering a genuine Info SMS. By forging a deceptive message, the victim user can be tricked into thinking that the mobile operator needs to reconfigure his handset by means of an upcoming provisioning message. The reconfiguration can be easily justified, for example by a handset misconfiguration or a network infrastructure parameters update.

Afterward, the attacker sends a malicious provisioning message to the victim. Due to the User Interface issues described above, and to the spoofed Info SMS message received, the user is persuaded to accept and install the new settings carried by the provisioning message.

Depending on the victim phone model, the new settings can be configured as the default ones automatically, upon installation time, or at connection time. For some handsets customized by a mobile operator, configuration settings may not be changed, or user interaction may be required.

However, in many cases the victim accepts the provisioning message and the configuration settings are installed on the victim mobile phone. As previously described, OMA provisioning allows for configuring several network parameters.

From an attacker point of view, the most interesting can be the APN, DNS and Proxy addresses.

The following section will describe how this attack scenario can be performed for hijacking mobile data connections based on HTTP protocol.

6 HTTP Traffic Interception

Most data connections generated by a mobile handset are based on HTTP protocol, for example web browsing, instant messaging and social network clients. Several standards for the mobile environment are also modeled on HTTP transactions as are many services provided by mobile operators. So, the interception of data exchanged during on HTTP transactions may be very attractive for an attacker. This can be achieved by redirecting the victim HTTP traffic toward a transparent proxy.

One possible way to obtain this is by subverting the victim DNS queries, redirecting them toward an attacker controlled DNS server, that answers with the transparent proxy IP address to all queries.

This scenario, however, requires DNS queries to be forwarded outside the mobile operator network and DNS response to be returned. Tests performed in several mobile operator networks have revealed that DNS filtering is not in place, allowing DNS queries to be forwarded.

By using the attack methodology described above, the victim handset can be reconfigured remotely in order to forward the DNS queries to the attacker controlled server. This can be achieved by sending a provisioning message with the same configuration parameters as the victim default access point together with the DNSADDRESS parameter pointing to the attacker DNS server. In this way, all the HTTP transactions are redirected toward the HTTP transparent proxy server in a way invisible to the user.

The HTTP transparent proxy can be implemented with the Apache web server [11] powered by the mod_proxy module [12]. This module normally requires the connections to be forwarded must come from a browser or an application configured to use an HTTP proxy.

In this scenario the connections coming from the victim are not configured to use an HTTP proxy. This issue can be solved using the

mod_rewrite [13]Apache module, that allows for rewriting the victim request in order to be correctly forwarded. The interception of the data connection can finally be achieved using the Audit feature of the ModSecurity apache module [14], able to log all of HTTP requests and responses forwarded by the proxy. The figure 2 shows the attack scenario described.

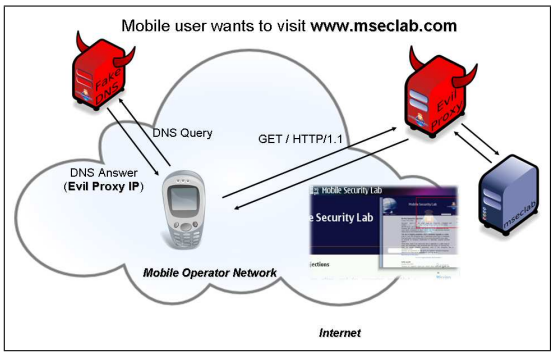

Figure 2: HTTP traffic redirection example

7 Applicability

The described attack provides means for an attacker for taking control of the mobile originated data connections. This possibility can be exploited in several scenarios, achieving in each of them relevant results from an attacker point of view.

7.1 Browsing

Session content can be accessed and login credentials, cookies and SessionID can be stolen and reused by an attacker for performing frauds, identity theft and similar abuses.

Session content can be modified and javascript injection can be easily performed, allowing for the degree of control typical of a Cross Site Scripting attack, but without the need of having an XSS vulnerable site.

"Same origin policy", that restricts cross-domain attacks, is practically bypassed because the full traffic is under attacker control, and scripting code can be injected in the desired context.

Spamming and unsolicited advertisements can be performed by injecting the desired content.

The described attack allows for application of already known methodologies that target SSL sessions [15] to the mobile environment. If poor certificates checking is performed on mobile devices, it could open even easier avenue of attacks, that may not currently be possible with desktop clients.

7.2 Applications

Mobile applications that rely on data connections can have their traffic hijacked also.

Instant messaging, VoIP applications, social networks can be targets of the described attacks allowing access and modification of their traffic sessions. An attacker could steal login credentials, monitor and inject content, and exploit client and server side vulnerabilities.

Encrypted sessions may be also at risks if flaws are present in the design or implementation of the application (eg: poor certificates checking)

7.3 Mobile Operator Networks

Data sessions that are normally confined inside the Mobile Operator network, such as a Mobile Phone accessing a website residing in the Operator Network, could be accessible to an external attacker. Such traffic could used for targeting server side vulnerabilities or for frauds. The ability to inject code that is run on the device side, could allow an external attacker to force the device itself to target network resources, that are not normally reachable from the outside.

7.4 Exploits and Botnet

The ability to fully control the data connections could allow an attacker to easily fingerprint a device and inject exploit code, tailored to the specific victim that leverage an existing vulnerability. By performing these action at a large scale it could be possible for an attacker to build mobile devices botnets.

8 Considerations and Remarks

The attack does not rely on a single vulnerability in a single elements, but exploits several elements that concur in making the attack scenario feasible.

Provisioning USERPIN mechanism is based on the user trusting the received messages. User trust is established only on visual elements.

SMS source spoofing and carefully chosen message trick users into believing that the attackers' messages have been sent by the Operator itself.

Depending on the device, User Interfaces may not provide users with sufficient information at configuration installation time.

If provisioning messages coming from untrusted sources are not subjected to proper filtering policies, the attacker succeed into delivering his own configuration to victim users.

If external DNS servers are reachable by an handset connected to a Mobile Operator, the attacker could be able to hijack all the data connections that rely on DNS server for resolving DNS names. (Attackers could also choose to use the HTTP Proxy address parameter, instead of the DNS server and hijack HTTP sessions, instead of the full range of data connections).

9 Countermeasures

Proper filtering of OMA Provisioning messages would entirely block the attack. Such filtering should be implemented at the network level, because performing it on the mobile device side (eg: whitelist) could be possibly bypassed by performing SMS source spoofing.

Improving information provided by User Interfaces at installation time, such measure could be ineffective in case SMS source spoofing is properly performed by attacker.

Inhibit access to external DNS servers from mobile devices connected to the Mobile Operator Network, but, if implemented alone, this could have unexpected side effects. An attacker could be able to perform massive Denial of Service by reconfiguring a large number of devices that would not be able to contact the attacker DNS server. This would lead to the impossibility for a victim to use data connections at all. Additionally, this countermeasure is ineffective in case the attacker chooses the HTTP Proxy parameter instead of the DNS server address.

References

[1] GSMA,*Mobile World Celebrates Four Billion Connections* . `http://www.gsmworld.com/newsroom/press-releases/2009/2521.htm`

[2] "WAP Architecture". Open Mobile Alliance `http://www.openmobilealliance.org`

[3] "WAP Push Architectural Overview". Open Mobile Alliance `http://www.openmobilealliance.org`

[4] "Wireless Datagram Protocol". WAP Forum `http://www.wapforum.org`

[5] "Wireless Session Protocol". WAP Forum `http://www.apache.org`

[6] "Provisioning Architecture Overview". Open Mobile Alliance `http://www.openmobilealliance.org`

[7] "Provisioning Bootstrap". Open Mobile Alliance `http://www.openmobilealliance.org`

[8] "OMA Provisioning Content Specification". Open Mobile Alliance `http://www.openmobilealliance.org`

[9] "WAP Binary XML Content Format". Open Mobile Alliance `http://www.openmobilealliance.org`

[10] ETSI Digital Cellular Telecommunication Systems (phase 2+); Technical realisation of the Short Message Service (SMS) Point-to-Point (P); `http://www.etsi.org/`

[11] Apache Software Foundation, *Apache HTTP Server Project.* `http://www.apache.org`

[12] Apache Software Foundation, *Apache Module mod_proxy* . http://httpd.apache.org/docs/2.0/mod/mod_proxy.html

[13] Apache Software Foundation, *Apache Module mod_rewrite* . http://httpd.apache.org/docs/2.0/mod/mod_rewrite.html

[14] Breach Security, *ModSecurity, Open Source Web Application Firewall.* http://www.modsecurity.org/

[15] Moxie Marlinspike,*New Techniques for Defeating SSL/TLS.* http://www.blackhat.com/presentations/bh-dc-09/Marlinspike/BlackHat-DC-09-Marlinspike-Defeating-SSL.pdf