MWR InfoSecurity Security Advisory

PCSC-Lite: pcscd ATR Handler Buffer Overflow

13th December 2010
PCSC-Lite: pscd ATR Handler Buffer Overflow Vulnerability

<table>
<thead>
<tr>
<th>Package Name:</th>
<th>PCSC-Lite</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date Reported</td>
<td>3rd November 2010</td>
</tr>
<tr>
<td>Affected Versions:</td>
<td>Confirmed in Version 1.5.3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CVE Reference</th>
<th>Not Yet Assigned</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author</td>
<td>Rafael Dominguez Vega</td>
</tr>
<tr>
<td>Severity</td>
<td>Medium Risk</td>
</tr>
<tr>
<td>Vulnerability Class</td>
<td>Buffer overflow</td>
</tr>
<tr>
<td>Exploit Details Included</td>
<td>No</td>
</tr>
</tbody>
</table>

**Overview**

MWR InfoSecurity identified a vulnerability in PCSC-Lite's pscd daemon. The vulnerability can be triggered using a malicious smart card.

**Impact**

An attacker could use this vulnerability to trigger a denial of service condition or potentially execute arbitrary code in the target system. To successfully exploit this vulnerability the attacker will be required to insert a specially crafted smart card in the target system.

**Cause**

A buffer overflow vulnerability was identified in the code handling the smart card’s ATR (atrhandler.c). The vulnerability occurs as the value of the length supplied in the affected memcpy can be incremented to be larger than the destination buffer.

**Interim Workaround**

Disabling the pscd daemon will prevent users from exploiting the vulnerability.

**Solution**


**Dependencies**

In order to successfully exploit the vulnerability described in this advisory, an attacker would need to have physical access to the affected system in order to be able to plug in a malicious smart card.
Detailed Vulnerability Description

The issue is a buffer overflow affecting the code responsible for handling the ATR coming from the card (atrhandler.c).

The affected code is included here. The vulnerability is in the memcpy shown below, as the value of "p" which is used as the length in the memcpy can be incremented to be larger (up to 45 bytes) than the destination buffer.

```c
#define MAX_ATR_SIZE            33
UCHAR Value[MAX_ATR_SIZE];
p = K = TCK = Y1i = T = 0;
Y1i = pucAtr[1] >> 4;
K = pucAtr[1] & 0x0F;
p = 2;
int i = 1;
do {
    short TAi, TBi, TCi, TDi;
    TAi = (Y1i & 0x01) ? pucAtr[p++] : -1;
    TBi = (Y1i & 0x02) ? pucAtr[p++] : -1;
    TCi = (Y1i & 0x04) ? pucAtr[p++] : -1;
    TDi = (Y1i & 0x08) ? pucAtr[p++] : -1;
    ...
    if (p > MAX_ATR_SIZE) return 0;
    i++;
} while (Y1i != 0);
psExtension->ATR.HistoryLength = K;
memcpy(psExtension->ATR.HistoryValue, &pucAtr[p], K);
p = p + K;
memcpy(psExtension->ATR.Value, pucAtr, p);
```

Source Code from atrhandler.c

Acknowledgement

Thanks to Nils for the support and guidance on this research.

Thanks to Ludovic Rousseau for his co-operation in working with the author in regards to this matter and acknowledge his prompt response in implementing a fix.