Registrant Credentials Security at .br

Frederico A. C. Neves
ccNSO TechDay – ICANN Beijing – 08/04/2013
Registration System

- .br registry

- NIR for Brazil - ASNs, IPv[46] blocks

Problem Size
3.1M domains
1.9K ASNs
100K IP blocks

2.7M objects administered directly through 1.4M system accounts (IDs)

500K domains (16%) administered through 60 registrars (EPP)
Increased as a high value target (or why we are such a low hanging fruit?)

Abril/2007 – PlayStation Network
http://en.wikipedia.org/wiki/PlayStation_Network_outage

Junho/2012 – LinkedIn
http://en.wikipedia.org/wiki/2012_LinkedIn_hack

Outubro/2012 – google.ie Hijack/ nic.pe compromise
http://www.lucidity.ie/blog/166-google-ie-hijacked-not-hacked
http://www.cyberwarnews.info/2012/10/20/peru-pe-domain-service-hacked-96000-credentials-leaked/

Novembro/2012 – [google|yahoo|apple|microsoft].[ro|pk] Hijack
http://www.computerworld.com/s/article/9234089/
Attackers_hijack_the_ro_domains_of_Google_Microsoft_Yahoo_others
http://www.theverge.com/2012/11/24/3685334/pakistani-domains-hacked

Novembro/2012 – nic.gp compromise
Credentials Storage

Clear text
abcd1234

Cryptographic Hash
61ee8b5601a84d5154387578466c8998848ba089
Trivial to explore with pre-computed dictionary on these days Laptop CPUs (2M hashes/s)

Salted Hash
xyzh-7be44f960a49c4f7f4ad862be96904dbb91b20b7
Possible to explore with GPUs (350G hashes/s)
Reports on 90% of the Linkedin hashes

Salted Adaptative Hash
010d9f3283ff3dff-86cbd8fced5f199d2afc0d4aba165041c0fa98b5
Difficult to implement using GPUs – (PBKDF2, Bcrypt, Scrypt)

Encrypted Salted Adaptative Hash
Symmetrical Key, OFB mode, Good IV

Great care needs to be taken on the choice of adaptative hash functions for public authentication services. Some of them are very expensive and could be turned in a DOS vector.

Promote good passwords practices. Passphrases of moderated sizes make brute force attacks impracticable
http://cartilha.cert.br/senhas/
Two Factor Authentication – 2FA

Something you know
Password/Passphrase

Something you have
Token with OTPs

Unencumbered available technology - IETF
HOTP/TOTP (RFCs 4226 and 6238)
HMAC – Hash Based Message Authentication Code
Shared Key
HOTP sequential number
TOTP sequential number based on a temporal interval
Origin (epoch 1/1/1970), intervals of 30s
State of last sequential used numbers
Authentication Security “Module” - ASM

Total decoupling from the frontend systems
RestFull Interface

<table>
<thead>
<tr>
<th>2FA</th>
<th>Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>PUT /otp/</td>
<td>PUT /pwd/&lt;auth&gt;</td>
</tr>
<tr>
<td>GET /secret/&lt;id&gt;</td>
<td>GET /pwd/&lt;id&gt;/&lt;auth&gt;</td>
</tr>
<tr>
<td>GET /otp/&lt;id&gt;/&lt;auth&gt;</td>
<td>DELETE /pwd/&lt;id&gt;</td>
</tr>
<tr>
<td>GET /htop/&lt;id&gt;</td>
<td></td>
</tr>
<tr>
<td>DELETE /otp/&lt;id&gt;</td>
<td></td>
</tr>
</tbody>
</table>

After the provisioning no more direct access to shared secrets or pwd credentials

State stored on normal RDBMS
Shared Secrets derived using HMAC, a Master Secret and the <id>
Pwd credentials protected using a Symmetric Key
Master Secret and Symmetric Key protected by SSSS generated at initialization and required to activate the ASM
Rate Limit

All authentication operations rate limited
Source Address ID

Using Token Bucket Algorithm
http://en.wikipedia.org/wiki/Token_bucket

State stored on Redis
Token App – Google Authenticator

Open Source high quality implementation
  Android
  iOS
  Windows Phone (Authenticator)
Provisioning of the Shared Secret

QR Code
Activation followed by HOTPs
Thanks
Comments/Questions?