XCSSA GPG Key Signing Roster & Verification

(take this home for your signing use)   v2008-03-17

Name, email	**Verbal**	**Vis.**	**Signed**
Key Info	**F.prnt**	**ID**	**(from**
Fingerprint	**Check**	**Check**	**home)**
A. Roy Pittman, roy@earthlightscomputers.com			
DSA-1024 / El-Gamal-4096	**[ ]**	**[ ]**	**[ ]**
7D9B 4127 AD2A C8FD FFF8 D044 5AB7 5FFB **B8F9 D75C**			
Alberto Laporte, arlaporte@gmail.com			
DSA-1024 / El-Gamal-2048	**[ ]**	**[ ]**	**[ ]**
9B23 13F8 AB22 7F0C 04B2 C464 48B4 7A17 **863B EFDE**			
Alberto Laporte, alberto.laporte@rackspace.com			
DSA-1024 / El-Gamal-2048	**[ ]**	**[ ]**	**[ ]**
1C17 5A27 0CD6 23C6 7708 BF2A FAC6 9175 **6983 4D44**			
Al Castanoli, al.castanoli@gmail.com			
DSA-1024 / El-Gamal-1024	**[ ]**	**[ ]**	**[ ]**
2F8A 4AC2 E398 C53E 4604 960A 63D3 855E **2D1E 60D1**			
Chris Goldsmith, cdhgold@tecman.com			
DSA-1024 / El-Gamal-4096	**[ ]**	**[ ]**	**[ ]**
A9D4 1D85 9711 3C45 3950 ED3D AB69 AE94 **00C4 B51F**			
Daniel J. Givens, daniel@rugmonster.org			
DSA-1024 / El-Gamal-2048	**[ ]**	**[ ]**	**[ ]**
71E3 CE97 A8E4 8A31 C0D4 7C2B EFC6 F507 **323F 6297**			
David Kowis, dkowis@shlrm.org			
RSA-3072 / RSA-2048	**[ ]**	**[ ]**	**[ ]**
98EC 7077 CB97 00A1 F2E7 3834 C9DF FAF4 **70EB 739B**			
Ed Tillman, spamcatcher93@gmail.com			
DSA-1024 / El-Gamal-4096	**[ ]**	**[ ]**	**[ ]**
2D89 F157 AB1C F490 5104 EE66 D2A3 AACC **CB9C 64D6**			
Sean P. Tompkins, seanp@thetompkins.net			
DSA-1024 / El-Gamal-4096	**[ ]**	**[ ]**	**[ ]**
6263 47E4 DE29 3EE9 0F02 A299 16FA 8CCB **3B1B EACE**			
Thomas W. Weeks, tom@theweeks.org			
DSA-1024 / El-Gamal-4096	**[ ]**	**[ ]**	**[ ]**
5A27 DABA EEBC 63A5 2A46 0D78 2757 662F **7501 52F1**			

Instructions for After the Keysigning Party

Following the party, you'll need to return home with these two sheets to actually sign the other people's keys. Different distros and mail clients have automation tools to make this critical step easier, however we're going to keep it simple and do it by hand. It's a little bit more manual, but it will give you a better understanding of what's actually going on when you DL, sign, and re-upload someone's key(s).

Overview of “doing the signing” from home:

1. Participants go home and retrieve the public keys of all key signing participants by fetching individual keys from public key servers.
2. Participants work through their “Keysigning Roster Sheet”, only downloading the keys that have a check for both the “Verbal” and “Visual” checkoff on the roster. Next, each downloaded key's fingerprint will be compared against the fingerprint on the roster, and will sign each key (details in HOWTO below).
3. Participants then upload each public key they've signed back up to the same public keyserver, and optionally email the signed person a GPG signed email so that the recipient can test/verify when the two way trust is established (the mail client, once configured for GPG use, will display the level of signature trust).
4. Throughout the process, each participant is probably going to re-synchronize with the keyserver several times. Each time synch with the keyserver, they should see more and more signatures appear on their own key as others sign theirs, thus extending the “web of trust”.

XCSSA Basic Command Line Signing HowTo:

1. Find the key ID on the roster sheet (last 8 of the fingerprint) for the key you're about to check & sign. For tweeks' key (for example), it would be “750152F1” from the roster sheet.
2. Fetch the public key using the key ID. If you're running GnuPG on the command line, you can do this by typing

   ```
   $ gpg --keyserver keyserver.ubuntu.com --recv-keys <KeyID>
   ```

   (where KeyID is obviously the ID of the key you want, no spaces).
3. Check that the fingerprint of the key you've just fetched matches the fingerprint on the slip of paper: run

   ```
   $ gpg --fingerprint <KeyID>
   ```

and compare it with the hard copy roster in front of you.

4. If the fingerprints match (screen & roster) and the person showed you sufficient ID (Verbal check and Visual ID from roster), only then should you do the actual 'signing' part of the process:

   ```
   $ gpg --sign-key <KeyID>
   ```

   answering the questions that it asks.
5. Next you need to re-upload the signed copy of their key back to the keyserver:

   ```
   $ gpg --keyserver keyserver.ubuntu.com --send-key <Key_ID>
   ```

   You should get back something like 'gpg: sending key <Key_ID> to hkp server keyserver.ubuntu.com'