3920bis Fixes

Peter Saint-Andre

XMPP WG Interim Meeting

February 7, 2011

3920bis has been approved for publication, but several people have found two small issues with the text. We still have time to fix these during "AUTH48" if our responsible Area Director (Gonzalo Camarillo) does not object...

Issue #1: SASL Mechanism Order

3920bis, Section 6.3.3:

Any entity that will act as a SASL client or a SASL server MUST maintain an ordered list of its preferred SASL mechanisms according to the client or server, where the list is ordered according to local policy or user configuration (which SHOULD be in order of perceived strength to enable the strongest authentication possible). The initiating entity MUST maintain its own preference order independent of the preference order of the receiving entity. A server MUST offer and a client MUST try SASL mechanisms in preference order.

Problems:

1. Server preference order is not communicated in other protocols (e.g., IMAP).

2. More importantly, the client ignores the server's preference order, so why communicate it?

From: **A server MUST offer and a client MUST try SASL mechanisms in preference order.**

To: **A client MUST try SASL mechanisms in its preference order.**

Issue #2: Namespace Prefix Enforcement

3920bis, Section 4.8.5:

An implementation MUST NOT generate namespace prefixes for elements qualified by the content namespace (i.e., the default namespace for data sent over the stream) if the content namespace is 'jabber:client' or 'jabber:server'. [...] An XMPP entity MUST NOT accept data that violates this rule (in particular, an XMPP server MUST NOT route or deliver such data to another entity); instead it MUST either ignore the data or close the stream....

Problems:

1. Some XMPP server implementations can in fact accept such data (or the data is provided "clean" by the XML parser).

2. Therefore this text might violate Postel's Law.

Proposed fix...

An XMPP entity **SHOULD** NOT accept data that violates this rule (in particular, an XMPP server MUST NOT route or deliver such data to another entity **without first correcting the error**); instead it **SHOULD** either ignore the data or close the stream with a stream error....