Diameter AAA API

IETF 65

David Frascone <dave@frascone.com>
Purpose

• Why was the API Created?
• Implementation independence
  – API is callback based. (Applications register to receive messages)
  – Generic API for initialization, sending messages, etc.
History

- Original version had Java / C++ support.
- 00 -> 01
  - Java support removed.
- 01 -> 04
  - No real changes, no comments from aaa list.
- 04 -> 05
  - Editorial fixes from comments on the list.
- 05 -> dime 00
  - More editorial changes from author(s).
Design

• Callback based
  – Application registers to receive messages via callback function.
  – Other functions (building messages, sending messages) are direct calls (not callbacks).
  – Structures are allocated by diameter server. Application uses references.
Design

- Information Hiding
  - Two structures are used to hold AVPs, one made public to the application, one internally used in the diameter server.
  - This is so applications do not need to know how AVPs are represented internally.
typedef struct avp {
    enum {
        AAA_RADIUS,
        AAA_DIAMETER
    } packetType;
    AAA_AVPCode code;
    uint16_t length;
    AAA_AVPFlag flags;
    AAA_AVPDataType type;
    AAAVendorId vendorId;
    void* data;
} AAA_AVp;

typedef struct xavp {
    AAA_AVP avp;
    struct xavp *next;
    struct xavp *prev;
    int privateFlags;
} Extended_AAA_Avp;
What next?

• Where do we go from here?
  – WG Item?
  – Assign Editors?

• Comments / Flames / Toss spoiled vegetables?