Event Architecture for EFM

Summary – A Simpler Approach

- Out with the old
  - Event table – gone!
  - Event indication bit – gone !
- In with the new
  - Bad things happen, OAM sends asynchronous event notification
  - Each notification can contain multiple events, each TLV encoded
  - No inherent reliability in event notification
    - Notifications may be lost! Not retried
  - Defining all current events to be periodic or states
    - if a bad condition continues to exist, multiple events generated over time
    - Errored seconds, power level low in an interval, remote failure in an interval

Where we were coming in to Kauai

Self-clearing Event Table

- Procedure
  - Event occurs
  - Event put in event table
    - Event table defined to hold all outstanding events
  - Event notification PDU MAY be sent (optional)
  - Whenever something in event table, AI bit set
  - Events self-clearing after X seconds
- Problems
  - Lose information if not gotten "quick" enough
  - Flag doesn't tell you if there's more than last time
- Key Points
  - Event table defined to hold all events
  - Event table self-clearing
  - Flag represents "state"

Where we are going out of Kauai

Asynchronous Event Notifications

- Procedures
  - Something happens
  - Event is translated to an Event TLV
    - Allows easy future extensibility and possibility of including diagnostic information
  - Event Notification PDU sent containing a set of event TLVs
  - Event notifications are NOT RETRIED and are NOT GUARANTEED to be received by the far end
    - Note: could retry N times but not saying so right now
  - No Event Table needed (I.e. entity that holds set of bad things that happened locally)
  - No Event Flag needed (Event Notification is only indication)

Asynchronous Event Notifications

- Issues
  - Event notification inherently not guaranteed to be received
  - All current events defined to be periodic
    - Errored seconds
    - Temperature below threshold in time period
  - The stateful/periodic nature of the event definitions ensures that sustained bad things are signaled reliably
    - Line degradation
    - Power degradation
  - Short duration self-correcting problems could be lost by far end
    - Glitch in line causing 1 second of errors

Asynchronous Event Notifications

- Simplifications
  - No Event Table (no need to query)
  - No Event Indicator (send everything, don't need flag)
  - No state (send it and forget it)
- Unless its absolutely required, we tried to stay away from it

Asynchronous Event Notifications

- Assumption
  - Proposal assumes that losing an event is acceptable
    - Alternate is to maintain state, acks, retries, etc. to get reliability
  - Seems ok for the events we know about
    - Can model them as "stuff" happening in some "window", so they refresh automatically over time
  - Want a big fat bold ugly editor's note:
    - **The event notification procedures of this document are designed on the assumption that event notification is only reliable as the underlying media, and that an absolute guarantee that every event is received by the peer is NOT a requirement. Comments on this assumption are explicitly solicited, and those that disagree with the assumption are asked to provide technical solutions to provide the requested reliability.**