mDNS Time-since-received

Status update

Ted Lemon <elemon@apple.com>

Why?

- With SRP replication, multiple advertising proxies will publish the same data
- Current mDNS behavior is that if the set of data registered on an owner name on one proxy is different than the set registered on another, this is a conflict
- This results in lots of self-inflicted conflicts, both when data that has been replicated changes due to an update, and also when an update is split across multiple mDNS packets
- We need a way to know:
  - that there is an actual conflict, or
  - that there is stale data, or
  - that we just have incomplete but current data

Published -05 before IETF deadline

- Document now represents current implementation:
  - TSR is now represented as an EDNS(0) option
  - Timestamp handled the same
    - in flight, it is represented as time relative to mDNS message transmission
    - at rest, it is an absolute time
      - implementation dependent
      - in our case absolute time relative to system start
    - Include a key tag for the key that identifies the authority for an owner name

Stale data/conflict detection and resolution

- Have data with TSR, get data with no TSR, that's a conflict
- Have data with no TSR, get data with a TSR, that's a conflict
- Have data with TSR, get data with a TSR with a different key hash, that's a conflict
- Have data with TSR, get data with TSR with same key hash, that's never a conflict, BUT:
  - If new data has more recent timestamp, our data is stale and is flushed
  - If new data has less recent timestamp, we ignore it because it's stale

Cache vs authority database

- Handling of cache versus authority is necessarily different
- If we get a local registration that conflicts with cache, we flush the cache and probe
- This can result in a new “error” response: “Stale Data”
- Similarly, if we have published data and we subsequently receive fresher data via mDNS, we cache the fresh data and fail the registration with “Stale Data”

Results

- No longer seeing spurious (self-inflicted) conflicts
- Substantial reduction in multicast traffic
- Current API is somewhat inadequate

API issues

- With SRP replication, we can update the timestamp at the same time we receive an mDNS packet with the new TSR
- In this case we get a “Stale Data” response even though we have just registered current data
- API doesn’t provide a way to address this

Proposed API approach

- A more transactional approach:
  - data (all records on owner name) is registered in transaction A
  - we get an update, register new data (all records on owner name) in transaction B
  - commit transaction B
  - old transaction is canceled with “Stale Data” error
  - Transactions are required because we might register multiple records on the same owner name; we don’t want remove events to be generated when the only thing that’s changing is the timestamp.
  - This is a WIP—we don’t actually know the answer yet, but something like this will likely be required for all APIs that implement TSR.

Additional mDNS changes?

- If we are publishing replicated data, the data has already been probed and announced
  - Currently we have a way to skip probing, but not announcing
- When removing replicated data that is still valid, would be best not to send Goodbye packets for the removed data
  - Doing so can cause problems if there is a persistent query on some name/RRtype
    - The querier will get a remove event even though the device is still there
    - No subsequent query will be sent for a long time, so the name/RRset will appear to be gone

Next Steps

- TSR is a prerequisite for Advertising Proxy
  - Can we adopt it?
- Should we document API changes? In TSR doc or elsewhere?