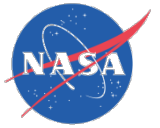
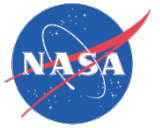


Store, Carry and Forward

Problem Statement, Requirements, Expectations and Testing

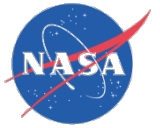
Will Ivancic

william.d.ivancic@nasa.gov



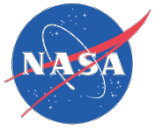
Strategy

1. Develop a Problem Statement
   - The need to standardize Store, Carry and Forward (SCF)
2. Develop SCF Requirements
3. Develop SCF Expectations
4. Define test scenarios and requirements for SCF systems
   - Motivation: testing of DTNs has been very poor at best and not well documented.
5. Develop an SCF protocol (after completing 1-4)



Design Principles

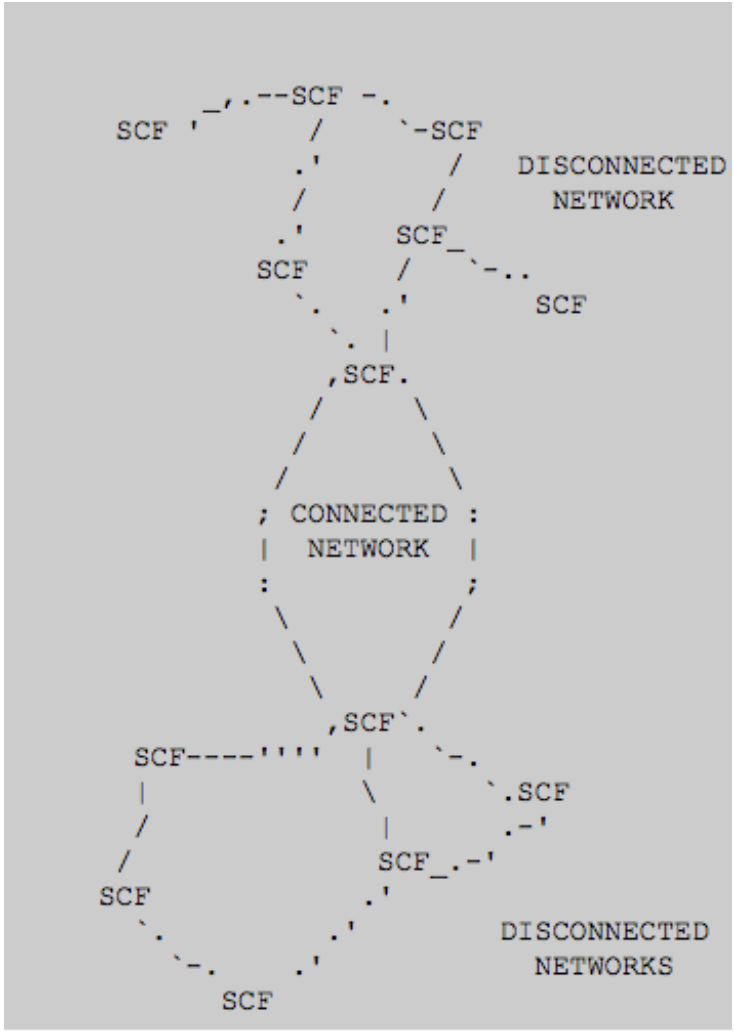
1. KISS
2. Options make interoperability hard
3. Options are often used as a place holder for fixing a bad design.
4. Don't Replace the Internet, Use it
5. Use new terminology. Avoid DTN terminology?
   - “Words make a difference. They affect how we thing about something. The terms chosen to describe a concept are a crucial part of any model. The right concepts with terms that give the wrong connotation can make a problem much more difficult. The right terms can make it much easier. Adopting the mindset of the terms may allow you to see thing you might not otherwise see.” - John Day, Patterns in Network Architect
6. Don’t overload the protocol.
   - Separate Network Management
   - Content Based Networking is different than SCF. SCF can be used to move content, but should not be considered content stores.
   - Separation allows for independent development and/or optimizations.
7. Naming and addressing are key - work to get this right
8. Addressing should be topological.
9. The old paradigm should never be invited to collaborate with the new paradigm.
10. "In anything at all, perfection if finally attained, not when there is no longer anything to add, but when there is no longer anything to take away..." - Antonie de Saint Exupery
11. "A good engineer is a lazy degenerate. He prefers degenerate cases to special cases and will sit around (thinking) until he finds a simple solution, rather that immediately launch into a brute force approach. In other words, the goal of a architect is to use the tools he has to make things simple. (Anyone can make things more complicated)!” - John Day, Patterns in Network Architect

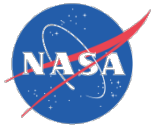


Problem

- This work is for systems that are generally disconnected from other disconnected systems.
  - We are not trying to look at the situation were a smart phone or computer is temporarily off or removed from the Internet and then is reattached to a connected network.
- Directed at non-real-time communication between systems that are generally disconnected, requiring multiple network hops between source and destination, that may never be fully connected end-to-end at any given time.

Architecture



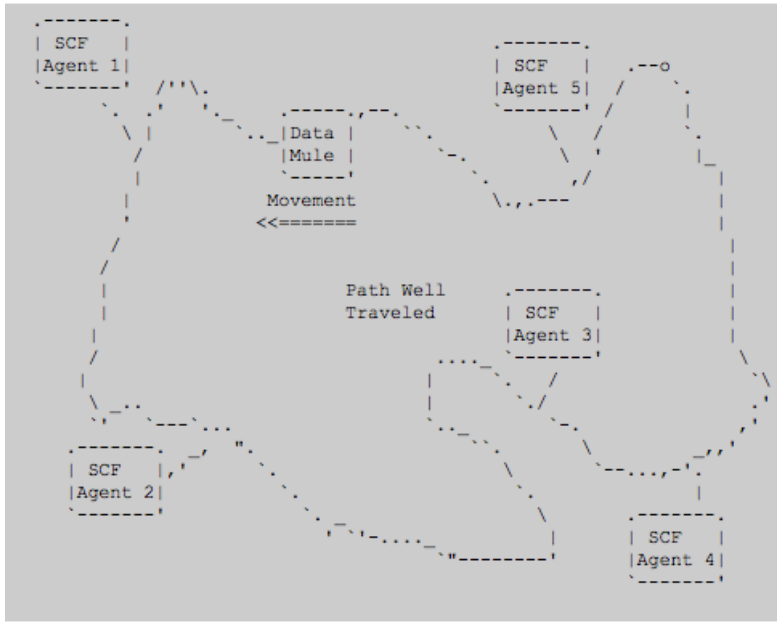


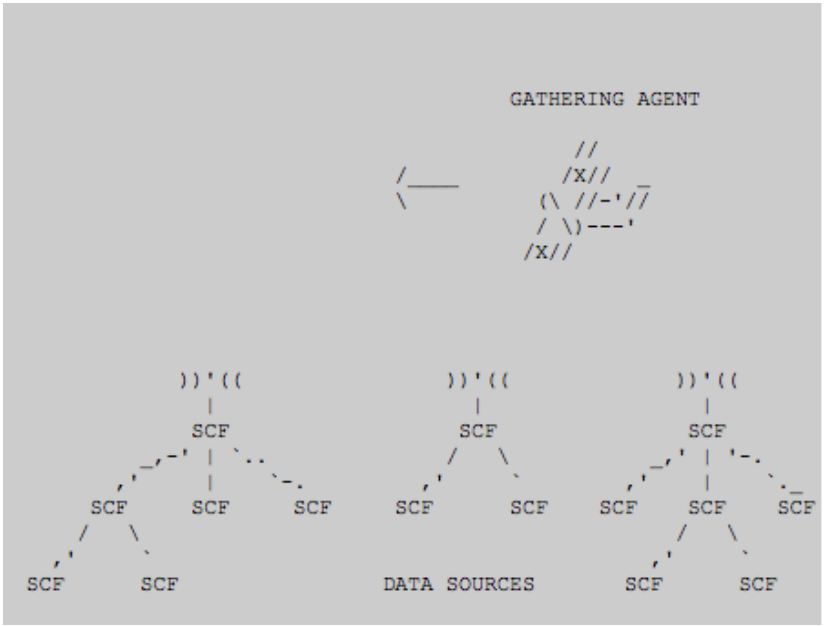
Operational Considerations

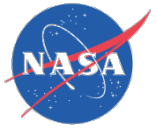
- What types of applications might be suitable to utilize SCF networking?
  - Engineering Telemetry - Accumulated over time for offline monitoring and analysis of some device or system's performance
  - Science Data Gathering - Similar to engineering telemetry, but sensor data is collected at a potentially much larger volume or over a much longer timescale. Due to the size of data sets collected, having multiple copies in-flight within the network may be undesirable, and end-nodes may need to purge old data after it has been sent in order to gather new data.
  - Software Updates - Numerous deployed devices that may never be able to contact an update server in real-time may need to have patches or updates deployed and activated.
- In general, any distributed application where senders and receivers can operate asynchronously in non-real-time, without any real-time requirement on the infrastructure (e.g. to do resolution of DNS names) might be able to function over an SCF service.
- What are the potential deployment scenarios?
- What are the upper layer user/application data set sizes?
- What are the traffic patterns?
  - Movement is not random, even for mobile ad hoc networks.
  - Information flow is not random, even for mobile ad hoc networks.
- What are the processing capabilities of the SCF agents?
- What type of interface between SCF agents and end applications is feasible?

Scenarios

- Data Mule (Distribution/Relay)
- Data Gathering (Retrieval)
- Rapid Disruption
- Traveling the Beaten Path
- Dismounted Soldier (or first responder)
- Low Earth Orbiting Sensor Satellites

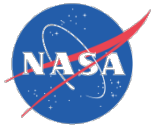






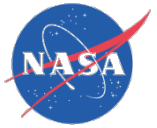
Consideration of Existing Technologies

- DTN technology may represent a basis for developing SCF standards. Several DTN routing protocols exist at varying levels of maturity that can work well for individual SCF scenarios that have been outlined
- Unlike DTN bundles, SCF containers are intended to be aggregatable within the network, even if they are not portions of the same original container from the application. Additionally, some SCF applications (e.g. science data collection) may find (optional) partial reception of subsets of large containers that have been deaggregated into smaller containers, to still be useful, whereas DTN only delivers entire (reassembled) bundles.
- SCF scenarios require some features that are not yet a part of the DTN specifications:
  - The ability to avoid DoS by propagating an application's permit/deny filters to SCF agents.
  - The ability to generate and prove ownership of globally unique application identifiers.
- SCF does not explicitly attempt to operate over long-delay links (though it may end up being possible) since these links are mostly only applicable to deep-space scenarios with small numbers of nodes.
- JMS "messages" transferred between "brokers" and applications are similar to the containers transferred between SCF agents and applications. JMS offers both point-to-point (unicast) and publish-subscribe (multicast) models of communication. JMS uses named "queues" (in the point-to-point model) or "topics" (in the publish-subscribe model) in order to identify destinations.
  - JMS is an API and not a protocol standard. This is the primary hurdle in using JMS to support SCF; as the wire-protocols and other mechanisms used in a particular JMS implementation are not necessarily compatible with others.
- SCF scenarios require some features that are not yet really reflected within the JMS specifications:
  - Multi-hop relaying among brokers and secure propagation of information about the queues/topics present or acceptable is not standardized.
  - Communication of an application's desired permit/deny filters on queues it owns is not standardized.



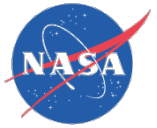
Lessons Learned (At lease we hope)

- SCF systems are generally connected via radio networks. Some radio systems may take far less power to listen than to transmit, though this varies by individual link technology. Wasted transmission is wasted power on a wireless system and can quickly drain a battery. The problem is compounded for devices whose entire lifetime is determined by their battery (e.g. non-rechargeable sensor nodes). Thus, reducing wasted transmissions is high desirable.
  - The ability to reactively fragment large data sets en-route is highly desirable. This has been demonstrated in DTN experiments.
  - Routing loops in the SCF will not be caught by layers below. It is imperative that data dies naturally and quickly so as to not waste bandwidth or transmission power. Such loops have been encountered in early experiments with DTN overlays, and are correctable.
  - It is highly desirable for the sender to know early in a transmission whether or not the receiver will accept the data. This permits a savings in power and optimization of network capacity usage. For instance, in DTN experiments with large bundles, the entire large bundle may be sent, only to be discarded due to security, resource scarcity, or other issues.
- Disconnected networks are difficult, if not impossible, to globally synchronize state across.
- It is highly desirable for a receiving agent to determine early within a transfer whether or not to accept the data. Data sets can be quite large utilize significant processing and storage recourses for data that may end up being discarded due to security, resource constraints, or other policy issues.
- It is highly desirable to keep forwarding tables small, and make forwarding decisions ahead of time for predicted contacts. Book-keeping type of processing while forwarding a large number of small containers can overload the processing system.
- Testing should be thorough and include exercising both the storage and forwarding systems. Failure to do so will lead to erroneous results
- Applications do not really know how to set time.



Information Characteristics
(Its About Time or Not)

- Source & Destination
- Sensitive or restricted (for your eyes only)
  - Not a protocol issue. This security is done at the application
  - Think encrypted email
- Size
- Useful Lifetime
  - Can be very short (seconds)
    - Really looking a forwarding priority
  - Can be very long (days, weeks, months, years)
    - Really looking a storage management
  - Would have to be set by applications
  - Applications don't know how to set this.
  - Useful for use to expire data-at-rest
    - But, really looking at storage priority
  - Very very messy to include as part of a protocol



Requirements

1. Must handle very small and very large data sets
2. Aggregation
3. De-aggregation
4. Reactive Fragmentation
5. Security when needed, but not required
   - Authentication/Authorization
   - Securing MUST work with Fragmentation
6. Separation of the Label from the Container
7. Cannot assume time synchronization (But it is no longer a factor – so who cares)
8. Separate Transport from Network Troubleshooting and Network Management
9. Data should die naturally to avoid routing loops (e.g. Hop Count, something else)
10. Something that specifies the application or content type (e.g. port numbers, mime, application name, etc...)
11. Reliability without security requirement
12. ~~Relative time (vs absolute time)~~
13. Sequence numbers? (There appears to be a desire in many instances to proactively create fixed bundle sizes in DTN and then what the application to put them back in order. With proactive fragmentation, this is possible and there is a mechanism to allow reordering. With straight bundling, this is problematic as there is not such formalized standard sequencing.)
14. Strive to limit the size of forwarding table.
15. The equivalent of late binding (i.e. name to address mapping reasonably often, certainly not once) - it's a mobility thing.

Aggregation (or Not?)

Requiers Topological Addressing

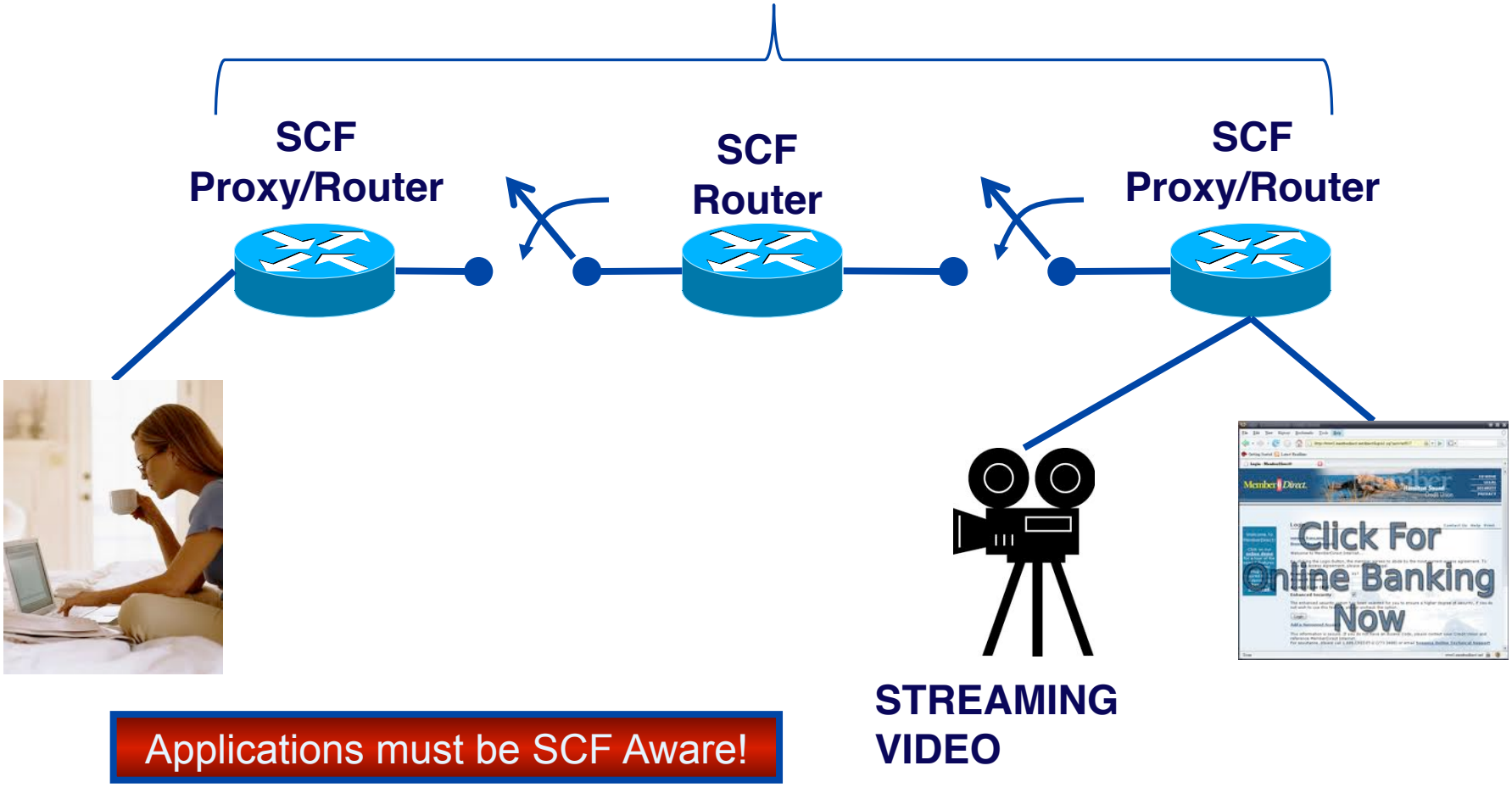
64K Bundles (De-aggregation)

Bundles Table Explosion

64K Bundles (Re-aggregation)

Applications and Testing

All to often these switches have been closed and never open! (Marketing?)



Applications must be SCF Aware!