6.033

Andres Romero – andresr@mit.edu

Julián González - jugonz97@mit.edu

Andrew Moran - andrewmo@mit.edu

Instructor: Srini Devadas

Tuesday/Thursday 12PM

05/12/2013

[CHRONICLE]

An Emergency Ad-Hoc Wireless Network

INTRODUCTION

This report outlines the design for Chronicle, an ad-hoc network of emergency first responders that can report their location coordinates and images to a central base station. Chronicle is designed for evacuation and rescue operations prompted by major disasters. Therefore, Chronicle is designed to create and update robust network paths to the base under frequently changing conditions, and to ensure that each first responder's location information reaches the base at least once every five minutes.

DESIGN

DESIGN OVERVIEW

Chronicle incorporates a modified Path Vector protocol to form the network topology, using distance from the base obtained with GPS as a fallback and during initial network setup. As they are time sensitive, first responder's locations are propagated through a layer similar to TCP to ensure their delivery. As image delivery is not as crucial, images are transported by a UDP layer. Queues at Chronicle nodes prioritize location packets over image packets. Finally, the use of hashing and encryption to protect packet headers and packet content respectively combined with limited packet bookkeeping prevent harmful replay attacks.

TRADE-OFFS AND MAJOR DECISIONS

Chronicle is intended to be a temporary, yet robust and persistent solution for a first responder network. Certain trade-offs were considered in terms of routing protocols and security to ensure simplicity and durability. For a more permanent design however, major features such as path metrics and security could be expanded upon.

ENCRYPTION

For a packet, the header is hashed and the payload is encrypted. By hashing the packet header instead of encrypting it, we preserve performance much more so than encrypting the entire packet. Encryption is favored for the payload in order to preserve the original input data much more effectively. There is always the vulnerability of a hacker guessing the hash function or encryption algorithm with our solution, however.

DISTANCE-VECTOR PROTOCOL VS. LINK-STATE PROTOCOL

In a distance-vector protocol, each entry in the routing table contains the outgoing node for a destination. There is the possibility of slow network convergence due to the change in topology or failed links. However, this seems more favorable than link-state. Link-state's convergence can be much faster, but the routing table of the network can grow very large. Each router knows the entire topology, which can produce much greater overhead.

PATH-VECTOR VS. DISTANCE-VECTOR

Distance vector algorithms advertise a distance metric for each reachable destination within the network. While this protocol guarantees connectivity, it suffers from the counting-to-infinity problem, where a node's estimate of distance can balloon very quickly. Although path vector may also suffer from slow convergence, it eliminates counting-to-infinity and the potential problem of cycles in the paths chosen by individual nodes.

ASSUMPTIONS

First responders (FRs) in this network are assumed to have mobile devices that are 802.11 compatible, and are capable of sending and receiving packets to and from arbitrary hosts. These devices come with unique user identification (UUID) numbers, are GPS-enabled, and are equipped with cameras. They aim to send their own GPS location, and when possible photographs to a base station, a computer with a fixed location that is pre-programmed into each FR device. FRs are also responsible for forwarding location data and photos of other FRs to the base station. The base station has a record of all UUIDs of first responders, and it is assumed to be stable and available to first responders at all times. Finally, the first responder and all nodes are assumed to share a synchronized clock, so that latency between two points can be measured by finding the difference of timestamps.

DEFINITIONS

NODE:

A FR mobile device equipped with an 802.11n card, GPS receiver, and camera.

IDLE TIME:

The amount of time that at least one data queue at a node is not empty

NETWORK OVERVIEW

Chronicle uses a combination of existing protocols to create a unique network suite that matches its needs. These are handled by these layers:

- Link Layer: 802.11n protocol, handled by the devices
- IP Layer: Path Vector with modifications (see: Network Connectivity)
- Transport Layer: UDP and modified TCP (see: Network Protocols)

NETWORK CONNECTIVITY

As it is important for FR location data to reliably reach the base station within five minutes, nodes must maintain an accurate picture of the network within the range of their wireless transmitter. To help do so, each node maintains the following information:

BASE STATION PATH TABLE (BSPT)

This table holds paths from the current node to the base station. A path is represented by a sequential list of the UUIDs of all nodes from the first hop to the base. All paths are freed of any routing loops when they are inserted into this table.

For each path, we also store the estimated probability that a packet can successfully be transmitted over it. As our FRs can query the loss probability p_{loss} for the link to every immediate neighbor, the success probability for a path of N nodes is simply $\prod_i^N (1 - p_{loss_i})$for all nodes i along the path.

Table 1. The Base Station Path Table at node G (a neighbor of node C) for a small network.

Path	Latency	Pr(success)	Idle %
{C, A, Base}	10 ms	0.684	1
{C, D, E, Base}	300 ms	0.457	0.5

It is important to recognize that picking a particular path to send packets down can potentially create a bottleneck, slowing down the performance of a large portion of the network. To that end, we store both an estimate of the latency (in milliseconds) of that path and the percentage of the last 30 seconds that all data queues at the first node along that path were empty (called the idle percentage). This information is transmitted to neighboring nodes during network setup.

NEIGHBOR LOCATION TABLE (NLT)

The *Neighbor Location Table* is a hash table mapping the UUID of neighboring nodes to an estimate of that neighbor's Euclidean distance to the base station (at a granularity of 1 meter). This table is only updated when there are no paths in the *Base Station Path Table* and only then is used to route packets.

QUEUES

As a node needs to both send packets and route received packets from neighbors, nodes maintain queues of outgoing messages. Because Chronicle prioritizes the delivery of location information, images are stored in a separate queue than location and routing information. Location information is also annotated with a timestamp of when it was sent from its original source.

NETWORK SETUP

PACKET FORMAT

In Chronicle, all messages between the base station and nodes or between nodes themselves use packets with a common header, described in Figure 2. In the header we include the origin of the message, the node the packet was actually received from, the desired recipient of the packet and the desired next hop of the packet. Unless this field is set to "Broadcast" (indicating that the packet should be forwarded to all neighbors, which is used during network setup) the next hop field is understood to be the preferred immediate destination of the packet, which is subject to alteration for load-balancing, when links fail, or when the current node simply has a better path.

This is because the eventual destination of all location and image packets is understood to be the base station. The recipient hop field can also be set to broadcast (again, this is used during network setup).

Table 2. The Chronicle packet header, shown for a message sent from node C.

Source	Prev Hop	Recipient Hop	Next Hop	Timestamp	Seq #	Hash(Source // Prev // Recp // Next // Time // Seq #)
C	D	E	Base	10:15:05:02	2	3476AED8C

INITIAL PROCEDURE

First responder devices initially wait for contact from the base station. The base station starts off by broadcasting an empty packet (an *announcement* containing, in this case, only a header) to all nodes within range.

After sending an acknowledgment back to the base station, neighboring nodes that hear the base's announcement store their path to the base station in the BSPT (namely, the direct path). They also compute and store the estimated latency of that path by finding the difference between the timestamp in the announcement's packet header and the current time. The queue idle percentage for the base station is customarily set to 1. Immediately afterwards, each neighbor scans for the loss probability of their link to the base station and saves the corresponding success probability to their BSPT.

Following the integration of information into each node's BSPT, each neighbor broadcasts an announcement to all of their neighbors. This announcement is a packet containing a list of tuples (*path*, *success_rate*) where *path* is a list of all the nodes on the path back to the base station in order and *success_rate* is the probability that a packet transmitted on this path will reach the base (without taking into account link failures). Each recipient of this announcement will send back an acknowledgement and then integrate each *path* from the message into their BSPT as before, except now paths containing routing loops are ignored.

As every node, *x*, that neighbors at least one other node, *y*, that has a path to the base, will integrate a path to the base into its owns BSPT, this "flooding" setup procedure guarantees connectivity in a network where FRs do not move a great deal. This is a step towards our goal. However, this "flooding" unfortunately leaves nodes that are farther from the base station idle until path information reaches them.

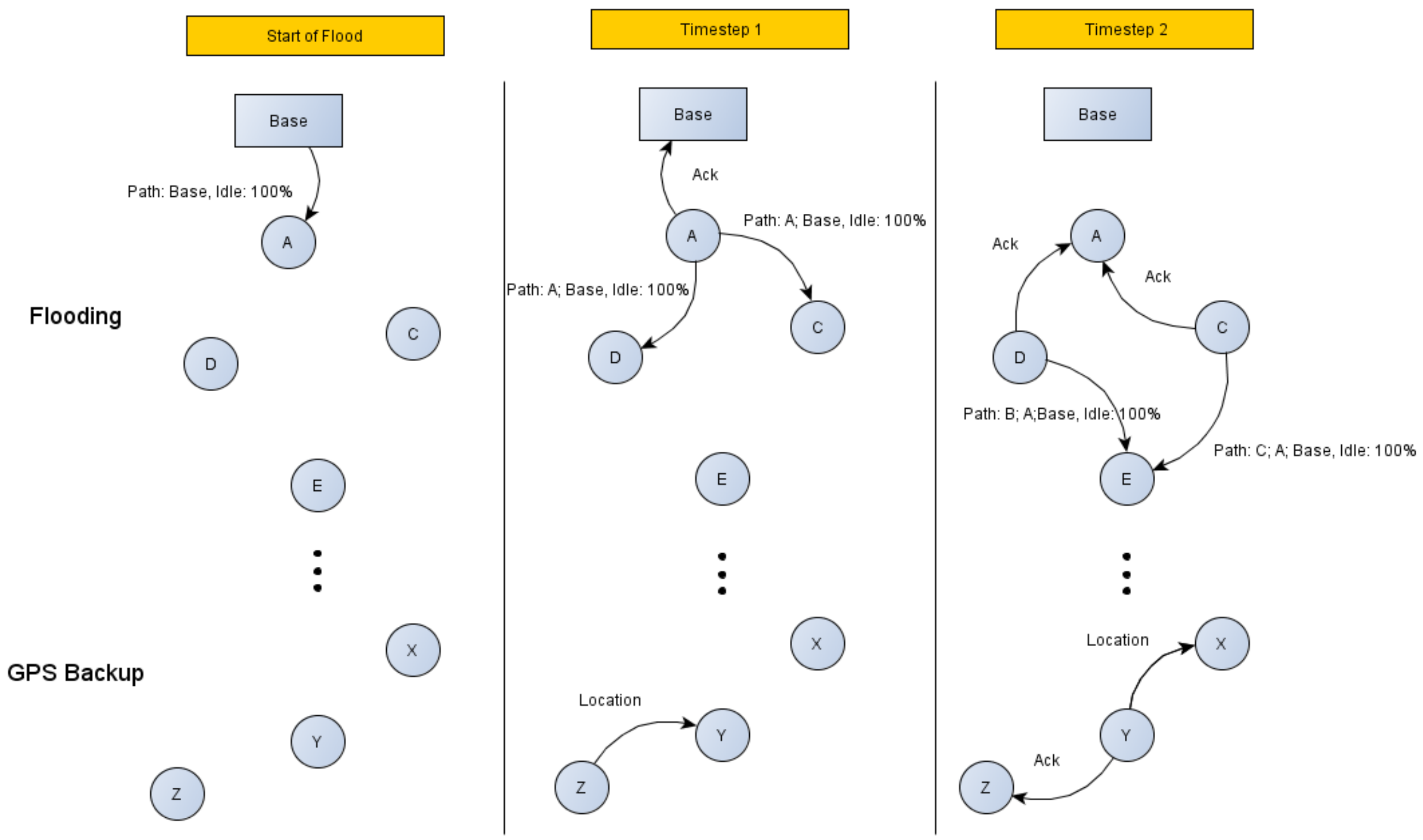

Figure 1. An illustration of the "flooding" procedure that is used to generate network topology.

Chronicle harnesses this unused network bandwidth by taking advantage of the fact that the base station's location is pre-programmed into all nodes. After startup, every node waits a period of time, τ, inversely proportional to its Euclidean distance to the base station. If a path to the base has not been received from a neighbor before τ has elapsed, the node will send its distance from the base station to its neighbors, who will store it in their *NLT*. Those neighbors will use this information to forward packets when the BSLT is empty, which will be the case until path information is received from nodes closer to the base station. Nodes that receive path information before τ elapses do not send GPS information to their neighbors, but may receive it.

In a large network, on the order of 1,000 nodes, where nodes are in an unfavorable topology for our flooding procedure, like a linear topology the node with the longest path to the base (in terms of number of hops) can wait up to a few hundredths of a second if link bandwidth is limited to a few tens of Mbps. As an example, if flooding information $f = 100\ bytes$ needs to propagate through $N = 1000$ nodes that each have a transmission rate of $r = 20\ Mbps$, it will take $\frac{Nf}{r} = \frac{(100\ bytes)(1000\ nodes)}{(20\ mbps)} = 0.0381\ seconds$ for the most distant node to receive location information. Therefore, all nodes should not wait more than a hundredth of that $time$ $(\frac{Nf}{100r} = 0.00381\ seconds)$ to begin transmitting location information to adequately forward an initial set of location information for distant nodes without interfering with the flooding of path information from the base.

PICKING PATHS

As the relative distance of a node to the base as measured by GPS is not a sure-fire measure of connectivity, entire paths (entries in the BSPT) are preferred when choosing outgoing routes.

When the BSPT is not empty, choosing how to send packets takes into account two metrics. The first is for location information packets. Because the goal here is to give the base some information on location within five minutes, we want to minimize the latency for location info. Therefore, we pick the path with the maximum $\frac{p_{sucess}}{latency}$ ratio. All location information is sent down this path to avoid highly successful paths that for any reason (e.g. have a large number of hops) are high latency paths.

For images, the metric Chronicle uses takes into account throughput and the load of the first node along each path. We pick paths probabilistically such that paths with a higher $\frac{p_{success} * idle}{\# \ hops}$ ratio receive more image data.

NETWORK PROTOCOLS

Due to the unique requirements of this network, two modified network protocols will be layered on top of 802.11. These protocols are roughly equivalent to TCP [1] and UDP [2], with modifications to make use of the provided API while ensuring robust performance.

The first protocol is used for all packets except for images, and it is based on TCP. Instead of sending acknowledgements after the successful delivery of a packet through an entire path, acknowledgements are sent at every hop. In this way, on a successful arrival packets become the responsibility of the next node on the path, and packet retransmission becomes localized. However, this also comes at a loss of information about overall paths, as unlike TCP nodes cannot calculate the estimated round-trip time of packets from their source to the base. Instead, a worst-case single hop time constant is used to decide whether packets should be resent. Estimated round-trip times for routing instead come from the path-vector flooding procedure (the estimated latency).

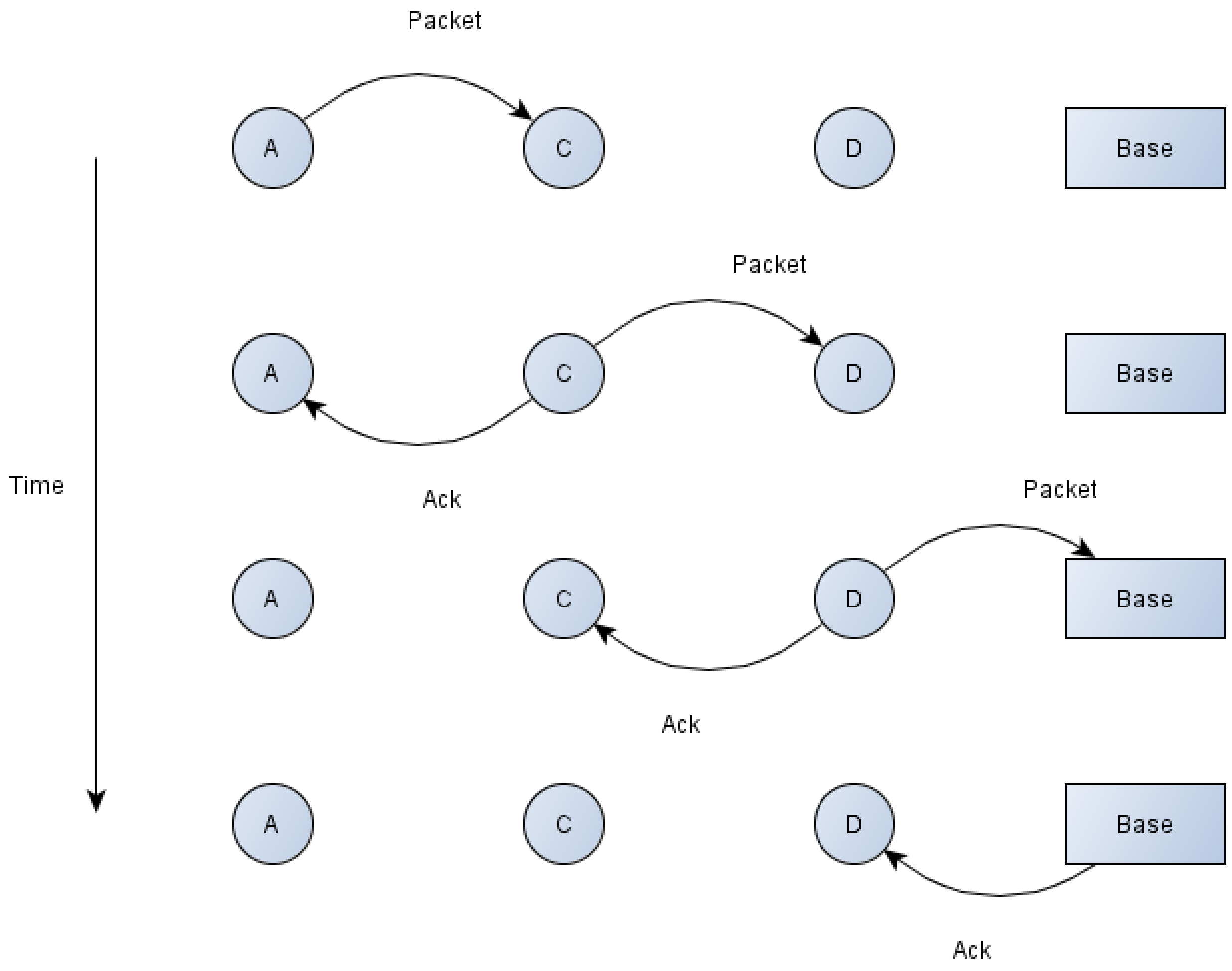

Figure 2: Per-hop acknowledgement and packet transmission responsibility in modified TCP protocol

Additionally, when the TCP protocol would normally resend a packet that was assumed to be lost, we scan for neighbors to check that the receiving node is still present before retrying. This allows links that have failed to be detected after a single send attempt, and lets the first responders alter their routes as necessary (see Network Operation).

Image delivery is not mission critical to the design of Chronicle, and therefore images are sent on the UDP protocol. This protocol remains unchanged from its definition in RFC 768, and therefore is optimized for reduced overhead and latency. However, if guaranteed image delivery was later found to be of more importance, images could also be sent using the our TCP-like protocol as long as they are still kept in a separate data queue at each node.

NETWORK OPERATION

Once the network topology is created, our “flooding” procedure will re-create all paths in the network every 30 seconds, the same interval that new GPS information is available at each node. This will keep all packet loss information up to date and lets nodes continually update their paths. Using this knowledge nodes can send images and GPS locations at will. However, if a connection between two nodes fails between two flooding periods, a node could be left without a path to the base station. In this case, one of two fallback procedures is used.

REQUEST FOR HELP

The first procedure attempts to recover paths from neighbors. If a node detects that its path to the base is no longer valid and that it has no alternative path in its BSPT, it is considered a “lost” node. This lost node will attempt to recover a path to the base station from one of its neighbors.

The “lost” node starts by broadcasting a “help” message through the TCP channel. Any responsive neighboring nodes will first check their path tables, and delete paths through the lost node. If those neighboring nodes have any remaining paths in their tables, they broadcast them to the lost node. The lost now announces its new path to the base to all of its neighbors.

If a neighboring node does not have an alternate path to the base after removing all paths through the lost node it also becomes a lost node, and propagates this “lost” state to any nodes that are connected to it until a path is found.

Assuming the network is a connected graph, this first procedure should reconnect all nodes to some path in the network. Unfortunately, there is no guarantee on this path’s performance. The next path vector flood will update all paths and provide the optimal visible path to the lost node.

GPS BACKUP

If the lost node no longer has a path to the base station after a timeout of 30 seconds, a secondary procedure is activated. This procedure is identical to the GPS based backup that was previously described in Network Formation. Using this procedure allows the cluster of lost nodes to temporarily forward packets to the node closest to the GPS location of the base, in an effort to forward those packets to the base (or a node hopefully close to the base) in the shortest amount of time.

SECURITY

In Chronicle, as with most WiFi-based networks, there is the risk of attacks from outside sources and malicious nodes. In the process of initializing and maintaining the network, other nodes can infiltrate it and issue a series of attacks. These attacks can include (but are not limited to) intercepting information, altering messages, and impersonating other nodes. Necessary measures are needed to prevent related attacks and to ensure malicious nodes do not negatively impact or jeopardize the network. Chronicle uses a combination of the packet header, additional record keeping, hashing, and encryption to help make the network more robust and secure.

Before describing Chronicle's security protocol, we need to describe the threat model of the network. After its initial setup, we will assume that malicious nodes do not have access to our hash function and encryption algorithm. Malicious nodes can, however, fake node IDs and perform frequent replay attacks by broadcasting messages originally sent from nearby first responders. In addition, we will assume that underlying API functions such as `scan() [3]` are secure, meaning that they will not present illegitimate nodes as neighbors.

PACKET STRUCTURE

Malicious nodes can pose a threat to the network by attempting to create new messages. However, this is unlikely to succeed due to the structure of a packet. A packet consists of a header and a payload. In the header, a packet stores the source, previous hop, next hop, timestamp, sequence number, and a hash value (of the remaining header fields). Chronicle will use a private hash function to produce that hash value. It will be difficult for a malicious node to generate an authentic hash value because it will not know the function used. Even if a malicious node fakes another node's ID to send a packet, the hash value of the packet would be hard to accurately replicate. As a result, any FR receiving a packet with an incorrect hash value can simply drop the packet. When the packet is dropped, no acknowledgement is sent back to its source. Additionally, if after reading a packet header a node discovers the packet was not intended for it by examining the Recipient Hop field in the packet header, it is discarded.

The payload is the actual message content corresponding to a packet. To ensure message privacy, messages are encrypted using a private encryption algorithm. Since malicious nodes need to know the encryption algorithm used to generate a message, it is very difficult to alter existing messages. The exact encryption algorithm to use is not specified, but it must have the property that encrypted text is not much larger than its unencrypted version (to not increase packet sizes superfluously).

REPLAY ATTACKS

If a malicious node hears a legitimate message and decides to broadcast it multiple times, the network can easily be flooded. This is a type of replay attack. To help prevent replay attacks, each node will store a table called the *Neighbor-Packet Table* (NPT). This table maps each node's neighbor to the sequence number of the last packet received from that neighbor. There will be two separate NPTs corresponding to location and image packets. Packets live in the NPTs for a designated Time To Live (TTL) that we set to the same time interval that location packets must reach the base station in (five minutes).

At the first attempt of a replay attack, a malicious node sends a packet to the intended nearby neighbor. The first test on this received packet is is to check to see if its timestamp exceeds our TTL. If so, the packet can be dropped and no ACK is sent because a newer legitimate packet for the node whose message was flooded likely exists. Otherwise, the FR will attempt to continue to process the packet. After reading the packet, the node will do a lookup of the suspicious packet's sequence number using the packet's source ID. If the sequence number from the NPT is equal to or greater than the sequence number from the suspicious packet, the packet is dropped. The FR can assume in this case that it has already forwarded the packet. As a result, a malicious node's attempt to repeatedly send messages can be avoided.

Malicious nodes are limited to the amount of threat they can impose on Chronicle. There are no conditions in which a first responder's message is prevented from reaching the base station given our threat model. A malicious node could send a legitimate broadcasted message to its intended receiver ahead of a legitimate node in the case of link failure, however because each keeps little state on its neighbors the only cost is an extra ACK sent from the legitimate node to the malicious node for sending a legitimate packet. However, with the assumption of underlying security, messages cannot be altered.

ANALYSIS

WORST CASE LATENCY

In the following analysis, we assume that Chronicle holds at most 1000 nodes, and that each link is spaced well enough to provide a link throughput of 2 Mbps with a zero loss probability between links. This expected throughput is abnormally low to account for collisions and the average link loss rate in 802.11.

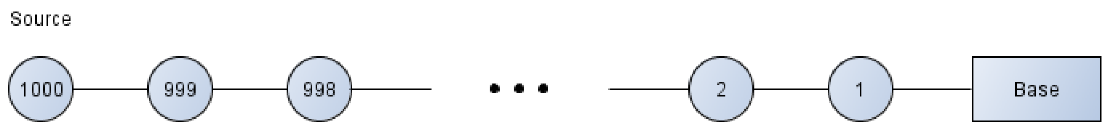

Figure 3: Worst Case Latency Topology

As each first responder may have up to 1000 location packets in their queue, we assume a worst case scenario where the node the largest number of hops away from the base station (labeled 1000 in Figure 5) holds 1000 packets in its queue. Each location packet is 1.5 kilobytes. Therefore, the total time to the base consists of (1.5kB) * (1000 packets)(2 Mbps) = 5.859 seconds in queue, with (1.5kB) * (1000 hops)(2 Mbps) = 5.859 seconds of transmission delay to traverse all hops. This places the total latency at 11.718 seconds, well within the five minute boundary for location information and providing a very strong guarantee that location information will arrive on time in the absence of link failures.

RESOURCE UTILIZATION

Generally, Chronicle has a low network overhead while in operation. However, there are cases in which it uses more resources than required to deliver a single packet. These are:

- Case 1. A link failure causes a packet to routed backwards, towards the source node, in order to reach the base. It is unfortunately very difficult to predict link failures, but Chronicle ensures that the packet will actually reach the base.

- Case 2. A message is sent and heard by a legitimate node farther up a path, that was not the immediate recipient. This node must drop that packet and wait to hear it later because our security scheme makes nodes drop packets not intended for them.
- Case 3. The GPS backup places the location packets further away from the source than they would have been had they stayed at a particular node. The GPS backup is a best-effort protocol as a node's proximity to the base does not guarantee connectivity.
- Case 4. A drastic change in loss rate between two floods. During the time period between those floods, our location data path metric will continue sending packets through paths that may not be optimal.

USE CASES

LIGHT LOAD

Light load conditions offer no threat to Chronicle, and both location and image packets are delivered consistently and quickly. Under light load, the expected throughput of each node is at least 20 Mbps, making the maximum latency of a location packet to be (1.5kB) * 2(20 Mbps) =1.17 seconds.

HEAVY CONGESTION

Under heavy congestion, GPS location and network routing information packets will still arrive to their destination, as they are prioritized to be sent first by nodes. This is evident from the worst-case latency of a fully connected network. Images, however, will suffer from our location prioritization and therefore have a lower throughput rate.

CATASTROPHIC CONGESTION

Under severe congestion (such as a malicious attack on the network operation frequency), Chronicle is unable to make any guarantees about packet delivery, but a good deal of this is inherent in wireless network design.

CONCLUSION

Our system design combines many efficient low level techniques into a robust network with low network overhead. The path vector protocol allows the network to maintain efficient paths to the base node. In addition, our GPS backup protocol minimizes the time path-vector takes to propagate network routing information, a main disadvantage of path-vector. Our unique security solution balances the use of hashing and encryption while avoiding large overheads to produce an adequate solution to the current threat model. These combined techniques minimize the number of false alarms at the base station, while maximizing throughput for images.

In the future, as needed, the network could be adapted to increase security by the use of advanced encryption techniques and handshakes. However, our current threat model deems the security risk not significant enough to warrant such overhead.

ACKNOWLEDGEMENTS AND REFERENCES

[1] V. G. Cerf and R. E. Kahn, "A Protocol for Packet Network Intercommunication," 5 May 1974. [Online]. Available: http://ece.ut.ac.ir/Classpages/F86/ECE571/Papers/CK74.pdf.

[2] J. Postel, "RF 768: User Datagram Protocol," 28 August 1980. [Online]. Available: http://tools.ietf.org/pdf/rfc768.pdf.

[3] 6. Staff, "DP2 Handout," May 2013. [Online].

Word Count: 4,266