An Enhanced VoIP Emergency Services Prototype

Jong Yul Kim, Wonsang Song, and Henning Schulzrinne
Department of Computer Science, Columbia University
{jk2520, ws2131, hgs} @ cs.columbia.edu

ABSTRACT

In this document we describe enhancements made to the prototype for emergency services in VoIP originally proposed and implemented in (Mintz-Habib, Rawat, Schulzrinne and Wu, 2005). In particular, we describe alternative methods of acquiring the physical location of an emergency caller and a novel way of using location information to determine call destination. We also introduce psapd, an enhanced third party call controller at the public safety answering point (PSAP), and discuss new features made possible by psapd. Preparations are underway in Texas and Virginia to install and test the enhanced prototype.

Keywords

SIP, VoIP, emergency service.

INTRODUCTION

As more people around the world choose VoIP services as their main means of voice communication, it is imperative that emergency services work in VoIP networks. In the United States, the National Emergency Number Association (NENA) has taken initiative to lay out the future steps for the evolution of Enhanced 9-1-1 (E9-1-1). Next Generation E9-1-1 (NG E9-1-1) is the name for the long term solution that is completely IP based. NG E9-1-1 has to support both current requirements and new features made feasible by transitioning to IP (NENA, 2005).

In (Mintz-Habib, Rawat, Schulzrinne and Wu, 2005), the authors proposed and implemented a working emergency services architecture and prototype for VoIP networks based on the Session Initiation Protocol (SIP). Session Initiation Protocol (SIP) is an application-layer signaling protocol for Internet telephony (Rosenberg, Schulzrinne, Camarillo, Johnston, Peterson, Sparks, Handley and Schooler, 2002). Figure 1 shows the architecture and Figure 2 shows the four steps of emergency call handling.

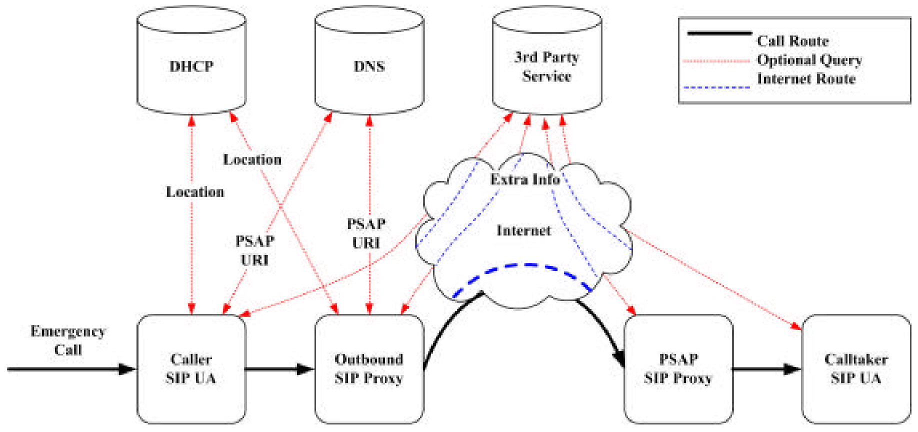

Figure 1. Emergency services system architecture (Mintz-Habib, et al., 2005)

Each component in the system architecture takes part in one or more steps of emergency call handling. The caller SIP UA is a soft or hard phone that initiates the emergency call. This component has to distinguish an emergency call from other calls by looking at the number or the SIP URI dialed. It may also determine the physical location of the caller using various methods including but not limited to DHCP. The call is forwarded to an outbound SIP proxy, which routes the call to the most appropriate PSAP based on the location of the caller. If the caller's location is not supplied by the caller SIP UA, then the proxy tries to determine the location by querying DHCP. The destination

PSAP is found by querying DNS server using a technique called DNS SOS. DNS SOS provides a mapping between civic address or geographic coordinates to PSAP URIs (Rosen, 2005). Once the PSAP URI is determined and the call is routed to that PSAP, it is then forwarded to a call taker SIP UA, which displays not only the caller's contact number or SIP URI and geographic location on a map, but also other related information such as pointers to the floor plan or the caller's medical history if provided. 3rd party services can also provide extra information. These steps are summarized in Figure 2 below.

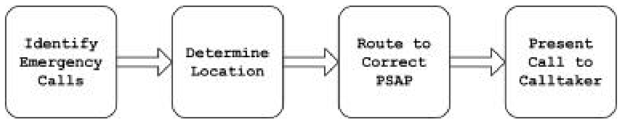

Figure 2. Control Flow for Emergency Call Handling (Mintz-Habib, et al., 2005)

In their implementation, they used the Columbia SIP User Agent (SIPc) and other hardware UAs for the caller SIP UA, Columbia InterNet Extensible Multimedia Architecture (CINEMA) for the SIP proxy, and SIPc for the call taker SIP UA.

While continuing to follow the architecture and the steps of call handling, we integrated novel solutions, modified components, and added new features. We are guided in our efforts by "NENA IP Capable PSAP Features And Capabilities Standard" (NENA, 2005).

In the remainder of our paper, we describe in detail the novel solutions and implementation details of the work.

DETERMINING THE LOCATION OF AN EMERGENCY CALLER

There are many ways to determine the location of an emergency caller and there is no single solution that works in every situation. (Mintz-Habib, et al., 2005) proposed and implemented methods using Dynamic Host Configuration Protocol (DHCP), Global Positioning System (GPS), and manual entry by users. Here we introduce another solution using Cisco Discovery Protocol (CDP).

Cisco Discovery Protocol

Cisco Discovery Protocol (CDP) is a protocol used for discovering devices on a network (Cisco, 2004). All CDP-enabled Cisco switches send periodic messages to a well-known multicast address (Cisco, 2004). Message interval is usually set to once per minute. CDP messages contain the name of the switch and the ID of the port that is sending the message (Cisco, 2004).

We believe that the area covered by a switch/port is small enough to be useful for emergency calling purposes. Switches cover a floor or half a floor, but each port leads to a jack in a specific room, so port information yields room-level accuracy. To get location data from switch and port information, we use a database located in each administrative domain that contains the mapping between the two.

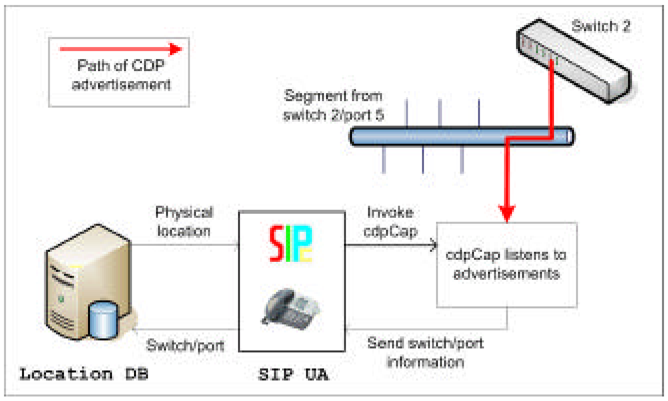
Path of CDP advertisement
Switch 2
Segment from switch 2/port 5
Physical location
Invoke cdpCap
cdpCap listens to advertisements
Switch/port
Send switch/port information
Location DB
SIP UA

Figure 3. Location determination using CDP

Figure 3 shows our implementation of this method. At startup, SIPc invokes cdpCap, which listens on the network and sends the switch/port information to SIPc whenever it captures a CDP advertisement. Then SIPc queries the central database for the location using switch/port information. The location is discarded after 65 seconds unless the same switch/port is advertised a minute later. If the switch/port information changes, then a new query is sent to the database and the location is updated.

However, this method requires network administrators to manage a central database that contains mappings between switch/port and the physical location. The additional burden imposed on network administrators would depend on how frequently switches are replaced or moved around.

We must mention here that while CDP is Cisco-specific, many other Ethernet switch vendors have similar protocols. Also, there is a generic solution called Link Layer Discovery Protocol-Media Endpoint Discovery (LLDP-MED) which will be standardized very soon. LLDP-MED explicitly supports location identification of endpoints (LLDP-MED, 2005).

Comparison of Different Methods

	CDP	DHCP	GPS	Manual Entry
Merits	??Cisco devices are ubiquitous ??Less burden for administrators than DHCP	??DHCP is ubiquitous ??Applicable to both SIP UA and SIP proxy	??Delivers precise location ??No work for administrators	??Is always a backup method
Drawbacks	??Only works with Cisco switches and access points ??Administrators have to enter switch – location mapping	??No good for wireless connections ??Administrators have to enter machine – location mapping for each machine	??GPS does not work indoors or when a significant portion of the sky is blocked from view.	??No guarantee of timely update ??Prone to human error
Useful Situation	In organizations that use Cisco devices	In organizations where computers are fixed in one place	Outdoors	When all else fails

Table 1. Different Methods of Location Determination

Table 1 shows the merits and drawbacks of different methods of determining the caller's location that we've implemented. Each method has varying degrees of usefulness in different situations. There are other methods such as using bluetooth (Mintz-Habib, et al., 2005) or compiling mappings of MAC addresses and locations (Plazes, 2004).

ROUTING EMERGENCY CALLS

An emergency call must be routed to an appropriate PSAP based on the location of the caller. (Mintz-Habib, et al., 2005) implemented a solution based on DNS SOS, which uses the DNS infrastructure to translate civic address or geospatial coordinates into PSAP URLs (Rosen, 2005). The fundamental problem with DNS SOS is that DNS forces a hierarchical structure upon its resource records whereas a geospatial coordinate does not imply any hierarchy. Here, we explain a new solution based on Location-to-URL Mapping Protocol (LUMP) proposed by (Schulzrinne, 2005).

LUMP is a protocol which maps a location to one or more URLs depending on the type of service requested. Location information is presented as either a civic address or geospatial coordinates. A service URN identifies the type of mapping service requested by the querying party. (Schulzrinne, 2005) proposed a URN namespace that allows to define context-dependent services. For example, emergency services can be defined as a list of URNs with the top-level service identifier 'sos', such as urn:service:sos, urn:service:sos:fire, and urn:service:sos:police.

The LUMP system consists of queriers which request a mapping service and resolvers which answer a particular query containing the civic address or the geospatial coordinates covered by the resolver. For redundancy and load-balancing, a set of resolvers can form a cluster that shares the same mapping information and returns the same results. LUMP is not bound to a structure that reflects the hierarchical nature of civic addresses. However, we organized LUMP nodes into a tree structure in which the root node serves as a nation-wide resolver, sub-nodes serve as state-wide resolvers, and so forth. Each LUMP node is assigned a LUMP URL such as "lump://leonia.nj.us". When a query comes in, a non-leaf node returns the LUMP URL of one of its child nodes that can resolve the requested location with finer granularity. A leaf node returns the URL of the PSAP that covers the requested location and the query stops.

LUMP uses web services for Remote Procedure Calls and we implemented each LUMP server using Apache Axis, which is a Java-based SOAP implementation (The Apache Software Foundation, 2005). We also used PostgreSQL with PostGIS extension to store and lookup the geospatial dataset (Refractions Research, 2005).

To determine the appropriate PSAP URL, the SIP proxy sends to the root resolver a LUMP request which includes the emergency caller's location information and the service URN such as 'urn:service:sos', indicating that this request is for mapping the location to a PSAP. The query will continue iteratively down the tree until it gets the final PSAP URL. Once the SIP proxy receives the PSAP URL, it will route the call to that PSAP.

PSAPD - AN ENHANCED THIRD-PARTY CALL CONTROLLER

The third party call controller (called psapd) is the main component inside our IP PSAP architecture, shown in Figure 4 below. It is a back-to-back SIP user agent that automatically handles all calls that are routed into the PSAP and supports advanced functions not possible in one-to-one communication. In our architecture, the capabilities of an IP PSAP are limited by what psapd can support. Therefore we had to make sure at the design phase that implementing additional functions to psapd would be easy.

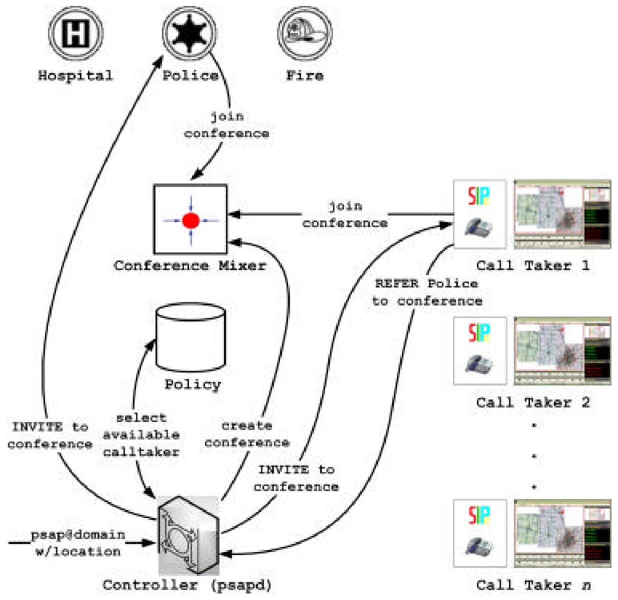

Figure 4. IP PSAP architecture (Mintz-Habib, et al. 2005)

Psapd is written in C++ and uses the Columbia InterNet Extensible Multimedia Architecture (CINEMA) library. The library provides SIP functions and state machines that comply with RFC 3261, the SIP standard document. With the library, we were able to design and implement sophisticated call handling logic with relative ease. It is also more reliable and robust partly because the CINEMA library has been thoroughly tested and used in a wide variety of applications for four years and partly because we designed and scrutinized call handling logic carefully before writing codes.

Figure 5 below shows the basic call handling state diagram. A normal call would start from state 1, pass through states 2, 3, 4, 6, 8, and terminate in state 9. When a call comes into a PSAP, it is received and accepted by psapd and all media streams are directed to the conference server. This makes it easy to add more participants later. So, every emergency call is a conference call and participants are not directly connected to each other - call signaling is handled by psapd and media streams are handled by the conference server. After accepting the incoming call, psapd selects a call taker and sends a SIP INVITE request to that call taker along with the caller's information and the conference server IP and port for media negotiations. The selection logic is written as a script which is invoked by psapd. This way, the logic can be customized easily to accommodate different policies of PSAPs. When the call taker accepts the call, he or she automatically joins the conference and a communication channel between the caller and the call taker is established. On the call taker side, the caller's location and other information are presented graphically using mapping software such as Geolynx or online mapping services such as Google Maps (GeoComm, 2005; Google, 2005). The conference is terminated and released when the call taker hangs up.

One important enhancement of psapd is faster notification of an incoming call. In the original prototype, the caller was bridged into the conference before the call taker was notified of the incoming call. With psapd, we changed the order so that the call taker is notified first, bridged into the conference upon acceptance of call, and then bridging the caller to the conference. This way, the call taker is alerted quickly and the caller is not confused by being in the conference all by oneself until the call taker is bridged in.

There can be many exceptions to the call. For example, all call takers might be busy answering other calls. In this case, psapd sends a "486 Busy Here" response. Then the SIP proxy server may try again at a later time or route the call to another PSAP. The other alternative is to queue the call within psapd but that is left as future work. Another exception is failed conference setup. For example, the conference server could go down and psapd has to deal with that situation. In this case, psapd returns a "500 Server Internal Error" response. Again, the SIP proxy may try to route the call to another PSAP.

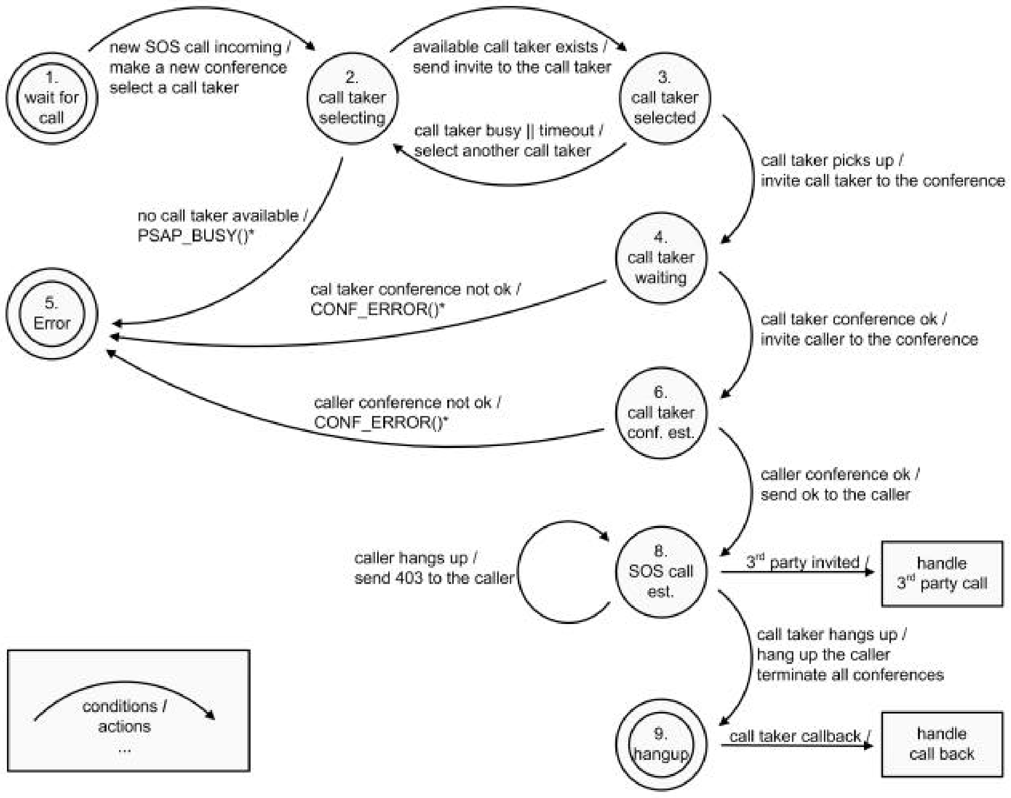
1. wait for call
new SOS call incoming / make a new conference select a call taker
2. call taker selecting
available call taker exists / send invite to the call taker
3. call taker selected
call taker busy || timeout / select another call taker
call taker picks up / invite call taker to the conference
no call taker available / PSAP_BUSY()*
4. call taker waiting
5. Error
cal taker conference not ok / CONF_ERROR()*
call taker conference ok / invite caller to the conference
caller conference not ok / CONF_ERROR()*
6. call taker conf. est.
caller conference ok / send ok to the caller
caller hangs up / send 403 to the caller
8. SOS call est.
3rd party invited /
handle 3rd party call
call taker hangs up / hang up the caller terminate all conferences
conditions / actions ...
9. hangup
call taker callback /
handle call back

Figure 5. Basic Call Handling State Diagram

In addition to call handling, we implemented new features for psapd such as language-based call distribution, better logging, call back of abandoned or disconnected calls, and text messaging among call participants. The following subsections describe implementation details of each feature.

New Features

Text Messaging

SIP supports text messaging with the SIP method MESSAGE described in RFC 3428 (Campbell, et al., 2002).

Text messaging is important in many aspects of IP-based emergency services. First, it provides a communication channel for the speech and hard of hearing community. Members of this community benefit by being able to use various devices that can run SIP clients, such as PDAs or laptops, instead of using TTY/TDD (Teletypewriter / Telecommunications Device for the Deaf) which are essentially modems. When text messaging is coupled with video using these devices, call takers will get a better understanding of the situation. Second, it can be used in situations where the caller is afraid of making noise and cannot speak up. Third, Instant Messaging (IM) users may also be able to access emergency services using their favorite IM client.

To support text messaging, psapd acts like a multicast router. When one participant sends a text message it must be copied to all other participants in the call. All participants are assumed to have text messaging capability. When psapd receives a MESSAGE request, it distributes the MESSAGE request to all other participants. Participants with text messaging capability typically return "200 OK" response. If delivery failed, then a 4xx or 5xx response is returned by the receiver and psapd notifies the sender of the failure with the reason.

Psapd does not send failure responses from participants other than the caller or the primary call taker. One reason behind this is that it is critical for the caller and the primary call taker to receive text messages but not as critical for other participants. Other participants can communicate with the primary call taker using voice channels. Another reason is to prevent the text window from becoming flooded with failure responses if there are multiple third party participants and most of them fail to get text messages.

Currently, psapd does not distinguish between permanent failures and temporary failures. Therefore, it continues to send text messages to participants that return failure responses. Adding 'group management' capability to psapd is left as future work.

Language-based Call Routing

Psapd is able to route calls based on the emergency caller's language preference. In our prototype, there is a database that contains each call taker's language capabilities and the relative preference for that language. Preference is displayed as a value between 0 and 1 where 1 means 'most preferred'. All call takers are assumed to have a preference of 1 for the default language of the particular country, e.g. English in the United States. A call taker who can communicate in two languages with ease can select both languages with a preference value of 1. In reality, it does not make sense to have a beginner in Spanish select the language and give it a preference of 0.2. Even if the call came through, the call taker would not be very helpful to the caller. However, that is beyond the technical aspect of this feature.

The caller also must select languages and give preference values. SIPc has a language selection window in which the caller can choose languages in the order of their preference. When a call is made, a SIP INVITE request with an Accept-Language header is sent. We follow the syntax and semantics described in Hypertext Transfer Protocol (HTTP/1.1) (Fielding, et al., 1997). The values of this header are ISO 639 2-letter language codes that correspond to the languages that the caller selected and the q-values which give order to the languages according to the caller's preference. An example is "Accept-Language: en, ko;q=0.8".

When psapd receives the INVITE request, it invokes the call routing script and supplies a list of languages in order of preference. The script goes through the list and queries the database for available call takers who are able to communicate in the language. If one is found, the call is routed to that call taker. Otherwise, the call is routed to any available call taker and both parties have to communicate in the default language.

Ability to Call Back Disconnected Calls

Emergency calls can be disconnected for a variety of reasons. When such an event occurs, the call taker may have to call back the emergency caller. This is supported in psapd by accepting calls from inside the PSAP. The call taker's SIP client has a 'call back' button which when clicked will initiate a call to psapd with a special header "In-Reply-To". The value of this header is the instance number of the disconnected call. Using that instance number, psapd queries the call log for the emergency caller's SIP URI and routes the call to the caller. This way, the call is established as a conference call just like incoming emergency calls. All call back calls are logged separately but have a link that refers back to the original disconnected call.

Ensuring that Callers Cannot Hang Up

Emergency situations require that calls are not terminated until the call taker decides to do so. Psapd fulfills this requirement by sending a "403 Forbidden" response whenever the caller sends a SIP BYE message. We modified SIPc so that it does not hang up when it receives "403 Forbidden" response. However, this behavior cannot be guaranteed in other SIP clients because RFC 3261, the SIP standard document, states that "the UAC MUST consider the session terminated (and therefore stop sending or listening or media) as soon as the BYE request is passed to the client transaction." Thus, this method currently works only for SIPc, but may be standardized as emergency call behavior in the future.

Enhanced Logging

Psapd supports better logging by recording data of interests directly to the database. Data such as the caller SIP URI, the call taker SIP URI, the time that the call was received and the time that the call taker answered the call are vital and are recorded. All incoming calls are logged regardless of whether they succeeded or failed in getting through. A separate web interface of the call log displays this information and allows call takers to append incident codes and comments to the log. It also calculates and displays call response time, calls per hour, week, or month, and number of calls received by a particular call taker.

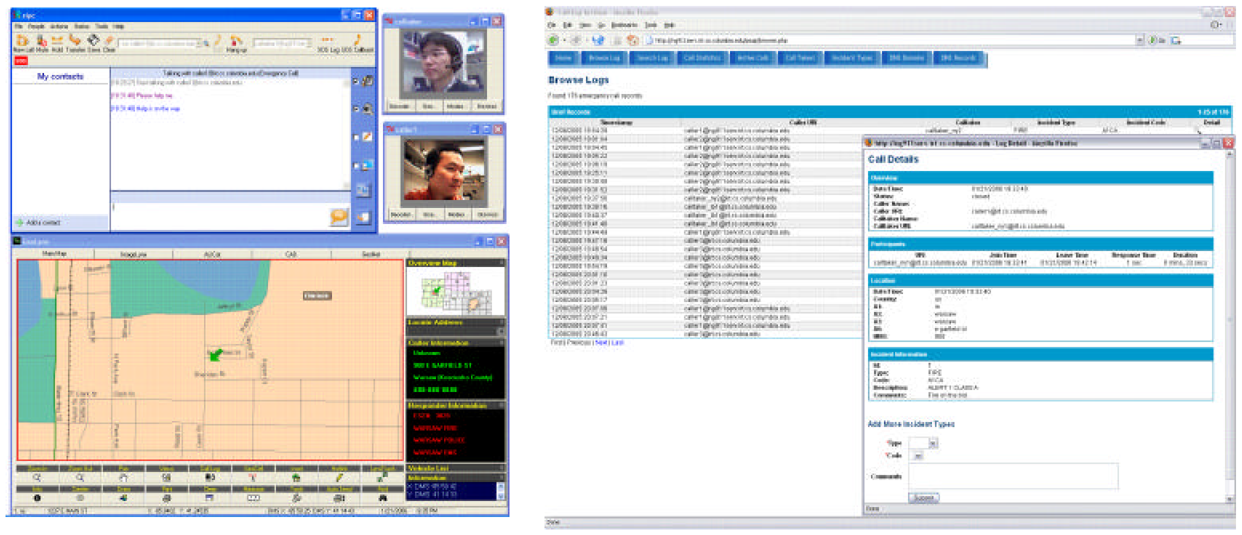

Figure 6. Screenshots of a call taker's monitor

Figure 6 shows what a call taker would see while answering an emergency call. This screenshot shows a SIP client with video support on the top left, a mapping software underneath the SIP client, and logs of calls on the right.

CONCLUSION

We have presented new ideas and improvements in design and implementation of the emergency services prototype first proposed in (Mintz-Habib, et al., 2005). A new method for determination of the caller's location was presented with implementation details. It was compared with other methods of location determination also implemented in the prototype. We described Location-to-URL Mapping Protocol and implemented it as a solution to the location-based call routing problem. Lastly, we explained the details of third party call controller in PSAP (psapd) and new features added to the system such as text messaging, language-based routing, call back, and enhanced logging. The improved psapd not only supports more features but is more reliable and robust.

The prototype system is being deployed in a live PSAP in Texas and Virginia for testing and feedback from professional 9-1-1 operators. The prototype will evolve to address the needs of the operators and to become more robust as we conduct multiple test runs.

We still see a lot of room for improvement in terms of performance and features. NENA Technical Information Document for NG E9-1-1 states that "call setup time (dialing of last digit to ring at the PSAP), under expected peak load shall be less than 2 seconds.” We used the logs of our outbound SIP proxy to measure the latency of our prototype. Specifically, we looked at the time when the SIP proxy first received a SIP INVITE request from the caller and subtracted it from the time when the same proxy received “180 Ringing” response from the call taker. The latency on average is 1.6 seconds. We found that the SIP proxy takes around 1.4 seconds to determine the appropriate PSAP. Reducing latency is very important and will be one of our main tasks in the future.

Also, some features in the requirements of IP PSAP, such as the queuing of incoming calls and transfer of calls between PSAPs, are missing. We intend to implement these features soon.

ACKNOWLEDGMENTS

The work described in this paper was supported in part under a grant by SIPquest and a Technology Opportunities Program (TOP) grant by the National Telecommunications and Information Administration (NTIA).

REFERENCES

1. Campbell, B., Rosenberg, J., Schulzrinne, H., Huitema, C., Gurle, D. (2002) Session Initiation Protocol (SIP) Extension for Instant Messaging, RFC 3428
2. CINEMA (2005) Columbia InterNet Extensible Multimedia Architecture, [Online]. Available: http://www.cs.columbia.edu/IRT/cinema
3. Cisco (2004) Configuring Cisco Discovery Protocol, [Online]. Available: http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122cgcr/ffun_c/fcfprt3/fcf015.htm
4. Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., Berners-Lee, T. (1999) Hypertext Transfer Protocol -- HTTP/1.1, RFC 2616
5. GeoComm Corporation (2005) GeoLynx Dispatch Mapping System, [Online]. Available: http://www.geo-comm.com
6. Google (2005) Google Maps, [Online]. Available: http://maps.google.com
7. Mintz-Habib, M., Rawat, A., Schulzrinne, H. and Wu, X. (2005) A VoIP Emergency Services Architecture and Prototype, *Fourteenth International Conference on Computer Communications and Networks (ICCCN'05)*, San Diego, California USA.
8. NENA, National Emergency Number Association (2005) NENA IP Capable PSAP Features And Capabilities Standard, Document 58-001 [Online]. Available: http://www.nena.org/VoIP_IP/
9. NENA, National Emergency Number Association (2005) Short Descriptions of E9-1-1 Evolution Steps, [Online]. Available: http://www.nena.org/VoIP_IP/
10. Plazes (2004) What’s Plazes, [Online]. Available: http://www.plazes.com/info/whatis/
11. Refractions Research (2005) What is PostGIS, [Online]. Available: http://postgis.refractions.net/
12. Rosen, B. (2005) Emergency Call Information in the Domain Name System, draft-rosen-dns-sos-03.txt, Internet Draft, work in progress.
13. Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A. R., Peterson, J., Sparks, R., Handley, M., and Schooler, E. (2002) SIP: Session initiation protocol, RFC 3261
14. Schulzrinne, H. (2005) A Uniform Resource Name (URN) for Services, draft-schulzrinne-sipping-service, Internet Draft, work in progress.
15. Schulzrinne, H. (2005) Location-to-URL Mapping Protocol (LUMP), draft-schulzrinne-ecrit-lump-01, Internet Draft, work in progress.
16. SIPc (2005) Columbia SIP User Agent, [Online]. Available: http://www.cs.columbia.edu/~xiaotaow/sipc
17. The Apache Software Foundation (2005) Apache Axis, [Online]. Available: http://ws.apache.org/axis/index.html
18. TIA, Telecommunications Industry Association (2005), Link Level Discovery Protocol – Media Endpoint Discovery (LLDP-MED) draft 08 (final), Project Number: PN-3-0185, [Online]. Available: http://www.tiaonline.org