VoCCN: Voice Over Content-Centric Networks

Van Jacobson D. K. Smetters Nick Briggs Michael Plass Paul Stewart
James D. Thornton and Rebecca Braynard
PARC
3333 Coyote Hill Road
Palo Alto, CA, USA, 94304
{van,smetters,briggs,plass,stewart,jthornton,rbraynar}@parc.com

ABSTRACT

A variety of proposals call for a new Internet architecture focused on retrieving content by name, but it has not been clear that any of these approaches can offer the generality to support Internet applications like real-time streaming or email. We present a detailed description of a prototype implementation of one such application – Voice over IP (VoIP) – in a content-based paradigm. This serves as a good example to show how content-based networking can offer advantages for the full range of Internet applications, if the architecture has certain key properties.

1. INTRODUCTION

There is widespread agreement that *content* – what a user wants – should have a more central role in future network architectures than it does in the Internet's current host-to-host conversation model [8, 4, 22, 3, 1, 13, 5, 17, 18, 21, 6, 2, 7, 14, 15, 16]. But while it is clear that architectures based on Pub-Sub and similar data-oriented abstractions provide a good fit to the massive amounts of static content exchanged via the World Wide Web and various P2P overlay networks, it is less clear how well they fit more conversational traffic such as email, e-commerce transactions or VoIP.

To investigate this question we have implemented VoCCN — a real-time, conversational, telephony application over Content-Centric Networking (CCN) [14, 15, 16] and found it to be simpler, more secure and more scalable than its VoIP (Voice-over-IP) equivalent. Our implementation uses standard SIP [10] and RTP [20] payloads which gives it complete *and secure* interoperability with standard-conforming VoIP implementations via a simple, stateless, IP-to-CCN gateway.

A version of this paper to appear in ACM ReArch'09, December 1, 2009, Rome, ITALY.

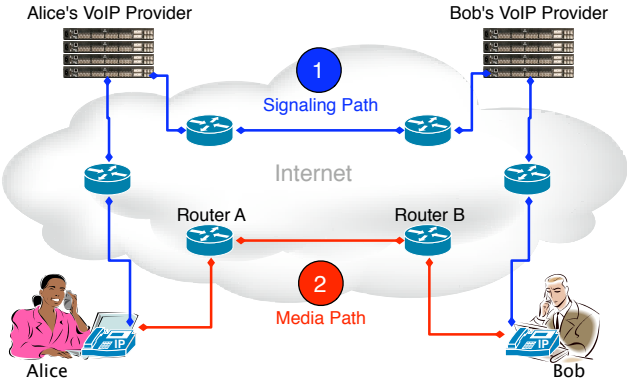

Figure 1: Voice-over-IP data flows

This paper describes how to map the existing VoIP architecture into CCN in a way which preserves security, interoperability, and performance. The mapping techniques are not unique to VoIP, but are examples of general transformations that we believe can be applied to almost any conversational Internet protocol. Through the example of VoCCN, therefore, we explore the properties that can enable networking models focused on content to be more general than traditional conversational models. These new architectures can therefore deliver benefits for both static content and the full range of conversational communication applications that are important in the Internet today.

2. VOIP BACKGROUND

Voice over Internet Protocol (VoIP) is the dominant open protocol for Internet telephony. Figure 1 depicts the components of a standard VoIP exchange. When Alice and Bob wish to make a phone call, their VoIP phones set up an audio link using the Session Initiation Protocol (SIP) [10] via what is termed the *signaling path*. As VoIP endpoints are often mobile or located on dynamic IP addresses, signaling path exchanges are mediated by *proxies* – service providers or corporate VoIP signaling gateways that receive and forward messages on behalf of their client endpoints. To place a

call to Bob, Alice's endpoint will first contact her SIP proxy who will forward the call invitation to Bob's SIP proxy, as only Bob's proxy knows the current IP address of Bob's endpoint. The body of the invitation contains both information about Alice and the RTP [20] address where she expects to receive audio (or other media streams) from Bob, should he accept the call. Bob's accept of the invite contains the RTP address of where he expects to receive audio from Alice which allows a direct, bi-directional *media path* between their endpoints.

VoIP media (voice, video, etc.) are typically secured and authenticated using either an encrypted form of RTP (SRTP [11]) or by tunneling RTP inside another secure network protocol (*e.g.,* DTLS [9]). The encryption keys are either set up via the signaling path, which must then itself be encrypted, or in-band in the media path (ZRTP [23]). Signaling path authentication and encryption is typically done via wrapping the signaling exchange in DTLS and relying on a Public Key Infrastructure (PKI) to authenticate the exchange or using a key agreement protocol such as Multimedia Internet Keying (MIKEY [12]) embedded in the signaling messages. MIKEY has the advantage of providing end-to-end security and authenticating its own messages while minimizing signaling path roundtrips, but does not provide confidentiality of the signaling pathway.

3. ARCHITECTURE

The complex data paths of Figure 1 result from a mismatch between the user's goal and the network's means of achieving it: Alice simply wants to talk to Bob but the network requires that the communication be addressed to the IP address of Bob's phone. All the infrastructure in the Figure, together with the several services, devices and DNS name registrations that are not shown, exist solely to map from the user/application's world view into the network's world view. One strong driver for content-oriented networking is that this translation (typically referred to as *middleware*) is not needed. Ideally data should flow directly from producer to interested consumer, as shown in Figure 2. Our VoCCN prototype achieves this.

There are a couple of problems that must be solved in order to map conversational protocols like SIP and RTP into a content-oriented model. First, we must support *service rendezvous*. To initiate a call, the caller's phone must be able to request a connection with the callee, and get a confirmation response. This requires the callee's phone to offer a service contact point. In standard IP, a port number serves as such a service contact point at which a process can receive requests, to which it dynamically generates responses. To translate this into a content-oriented model, we need *on-demand publishing*: the ability to request content that has not yet been published, route that request to potential publishers, and have them create, and then publish, the desired content in response.

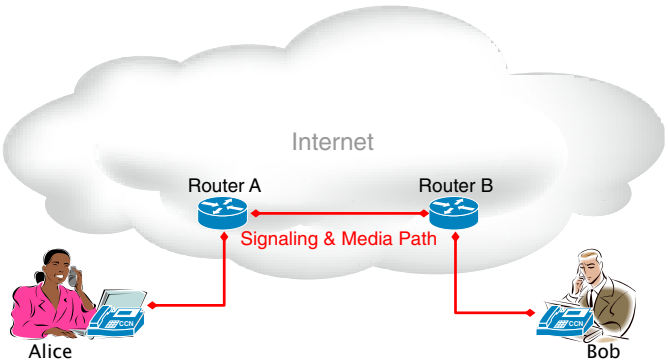

Figure 2: Voice-over-CCN data flows

Second, we must have a way to transition from this initial rendezvous to a bi-directional flow of conversational data. In standard IP, there are packets (either designated packets in the rendezvous sequence, or all TCP or UDP packets) that contain the information needed to name the destination to which replies should be sent. For example, a TCP packet header contains a source IP address and port plus a protocol identifier. In a SIP exchange, the SDP content of the SIP message (shown in Figure 3) identifies the address to use for the media conversation. To translate this into a content-oriented model, we need *constructable names*: it must be possible to construct the name of a desired piece of content *a priori*, without having been given the name up front or having previously seen the content – the service consumer must be able to figure out how to a request that will reach the service provider. This can be done if:

- There is a deterministic algorithm by which both the data provider and data consumer arrive at the same (routable) name based on data available to both.
- Consumers can retrieve content based on partially-specified names.

The former requirement guarantees that consumer and producer will arrive at the same name, and that names will not depend on data neither has (such as naming content by its cryptographic digest, impossible for data the consumer has not seen, or that does not yet exist). The latter deals with the fact that without significant prearrangement to allow for a source of shared randomness, such constructed names will not be unique. By allowing flexibility in the query mechanism, we can allow for uniquely named content, while matching it to deterministically generated queries. For example, generating a query for a structured name that matches only

```
INVITE sip:bob@bigco.com SIP/2.0
Via:  SIP/2.0/CCN bigco.com:5060
From:  Alice Briggs <sip:alice@otherco.com>
To:  Bob Jacobs <sip:bob@bigco.com>
Call-ID: 1911287229
CSeq:  20 INVITE
Content-Type:  application/sdp
Max-Forwards:  70
User-Agent:  Linphone/3.0.0 (eXosip2/3.1.0)
Subject:  Phone call
Expires:  120
Content-Length:  1477
[...
o=alice 123456 654321 IN IP4 13.2.117.34
c=IN IP4 13.2.117.34
a=key-mgmt:  mikey AQQFgE3dV+ACAA...
m=audio 7078 RTP/AVP 111 110 0 3 8 101
...]
```

Figure 3: Example of SIP INVITE message

the prefix of that name, . This latter facility also supports on-demand publishing,as described below.

In CCN, each fragment of content that may be published in the network has a hierarchically structured name, and requests for content are expressed in *Interest* packets, which specify the prefix of the name of the desired content and a set of rules by which to determine what of the content under that prefix to return. CCN routing tables use prefix matching to directly route interests based on their name prefixes towards content sources that have registered availability of content by prefix. CCN does not require that data be published and registered with the infrastructure before it can be retrieved; a request merely needs to start with a prefix registered with intervening routers to make it to an interested publisher, who can then look at the request and create content dynamically in response. The network forwards matching Data packets back along the path taken by Interests so content reaches the requester and is never sent where it was not requested.

In the case of a SIP rendezvous, each phone endpoint is configured at provisioning time with an identity, and registers to offer data in a namespace derived from that identity and the name of the SIP service. A caller maps a SIP INVITE such as that shown in Figure 3 into an interest packet asking for new content from the callee. The network routes the interest to the callee, which can generate a piece of data with the requested name containing the SIP response. This exchange is illustrated in Figure 4. The use of structured names in CCN allows a simple mapping of the callee identity and service name into the first part of the content name (used for the hierarchical, prefix-based routing) while unique identifiers for the request are added to distinguish the desired content from anything existing. Note that the entire SIP INVITE message is included in the requested content name. The callee receiving the interest can unpack the name and generate a the SIP response as the content of the Data packet that will satisfy the Interest.

To transition from the SIP rendezvous into the media conversation with RTP, each phone takes information exchanged in the rendezvous and uses it to construct a sequence of names for the individual packets of media data. This is also illustrated in Figure 4, where we see that the `call-id` from the SIP exchange, together with the identity of the other party and a sequence number is used to construct the name of each fragment of media. Sequence numbering provides a simple way for data provider and consumer to arrive at the same unique name for each piece of content.

In the content delivery architecture of CCN, Interests and Data flow in lock-step, each Interest retrieving a single data packet. In a dispersed or high-latency network, the round-trip latency can easily be large enough to delay reception times of media packets to the point where they become unplayable. To solve this problem, we employed pipelining by sending Interests in multiple packets at the same time. The CCN media receiver maintains some number of outstanding interests in a media stream; when the stream is opened (or as network conditions change) it generates a number of Interests. Each time it receives content for that stream, it produces a new Interest, thereby restoring the number of outstanding interests in the pipeline.

3.1 Advantages

Besides simply supporting an existing conversational protocol, the content approach adds a few appealing properties:

- Content networking infrastructures support multipoint routing – for example, routing a call request to all likely places where it might be answered. This supports multipoint calling directly in the infrastructure, and removes the requirement that endpoints register their IP address every time they move, at least within the routing domain.
- The "identity" of an endpoint in a content infrastructure is represented by credentials located on that endpoint – *i.e.,* a signing key, identifying content (*e.g.,* voice packets) that it creates. Management in a voice content system is minimized, as provisioning a new endpoint consists only of giving that endpoint a credential.
- Advanced services (voicemail, call logging and recording, conference calling) can be built easily on top of a voice conference system as additional components that follow call requests or copy and process call contents.

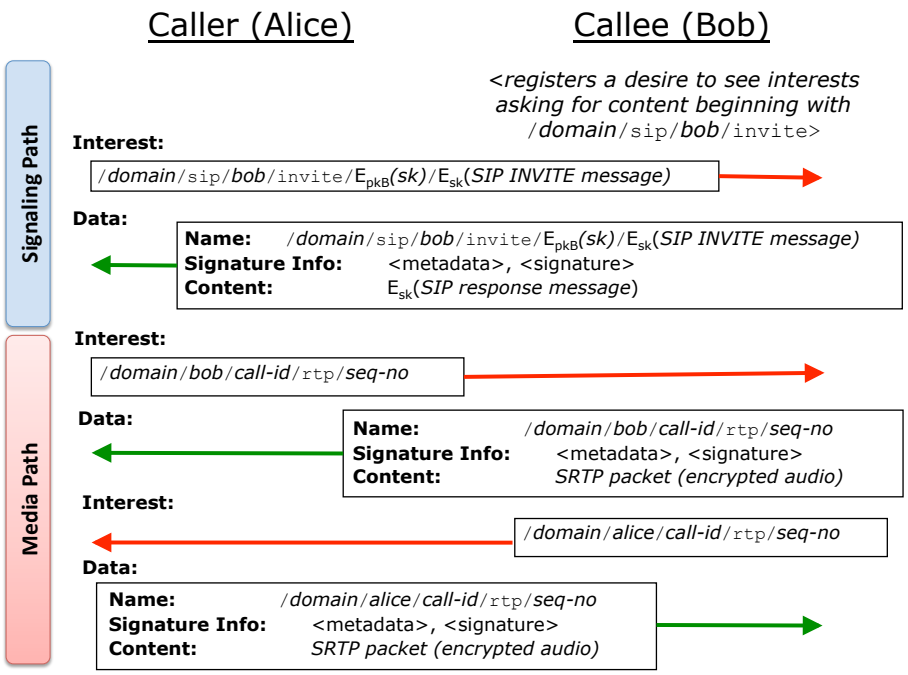

Figure 4: Protocol exchange.

3.2 VoCCN/VoIP Interoperability

In order to achieve interoperability between VoIP and VoCCN, we have designed (but not yet implemented) a stateless VoCCN-VoIP gateway. This gateway, which also serves as the SIP proxy for VoIP calls, translates from VoIP packets (SIP and SRTP) to VoCCN packets (which again merely encapsulate SIP and SRTP for this application), and vice versa.

On recieving a SIP or SRTP packet, the proxy merely examines that packet, and generates a corresponding CCN data packet whose name is determined based on information in the original inbound packet header (see below). It then forwards it into the CCN routing fabric in response to an interest from the other endpoint involved in the call. The proxy then performs the CCN-specific parts of the call on behalf of the legacy VoIP client – generating and sending an interest in the next packet of the exchange, where the name used in the interest is also computed as a function of information in the inbound VoIP packet.

The proxy retains no state about the call; responding only to received (CCN or VoIP) packets – the call "state" is contained at the endpoints and in the interests noted in the forwarding tables along the path to the content source. The proxy also has limited participation in call security. Key exchange and media path encryption (if supported by the VoIP client) is end-to-end. SIP signaling security is protected by legacy mechanisms between the VoIP client and the VoCCN-VoIP gateway/SIP proxy, and all messages translated by the proxy are digitally signed (by the proxy) before sending them into the CCN infrastructure. The proxy can also encrypt the SIP messages between itself and the CCN endpoint.

4. IMPLEMENTATION

We have implemented a proof-of-concept VoCCN client as an extension to an open source Linux VoIP phone, Linphone (version 3.0). We made Linphone exchange data over CCN by taking advantage of the ability to plug new transports into libeXoSIP and liboRTP, the libraries it uses for SIP and RTP.

To implement our content-based network substrate, we used the open source Content-Centric Networking (CCN) Toolkit, available from [19]. This toolkit provides both a content router, which runs on every CCN-aware node, and a set of interface libraries to simplify the process of writing content-based applications. We ran content routers on endpoint nodes, which sent CCN packets via an overlay consisting of UDP sent over pre-configured point-to-point or multicast links.

4.1 Security

All VoCCN packets in both the signaling and the media paths were digitally signed, using per-user key pairs. The corresponding public keys were also distributed via CCN. A simple extension would have these keys further signed (again as CCN data) by an organizational root

key, effectively building a corporate PKI.

To provide media path security, we ran all calls over SRTP,[1] using pre-existing hooks to integrate `libsrtp`, an open SRTP implementation, with Linphone. We implemented a MIKEY [12] library to perform key exchange in the signaling path. We selected MIKEY over DTLS for its ability to perform a complete SIP exchange and key setup in a single round trip.

To provide signaling path security, we implemented a simple inline message encryption and authentication scheme (shown in Figure 4). The caller, after generating a SIP invite message, would encrypt and cryptographically authenticate (using a symmetric-key MAC, HMAC) that message using a randomly-generated symmetric key, *sk*. The caller would then encrypt that key under the public key of the callee (*pkB* in Figure 4). When constructing a call-initiating interest message, the caller would include both the encrypted key block ($E_{pkB}(sk)$ in Figure 4) and the encrypted and authenticated SIP message ($E_{sk}(SIP\ INVITE\ message)$ in Figure 4) as components of the name it was expressing interest in. The callee, on receiving the interest, could decrypt the key block with its private key, recover *sk*, and use it to verify and decrypt the SIP INVITE (which itself contained the first, separately authenticated, MIKEY key exchange message). The caller would then use *sk* to encrypt its SIP response message.

The net result offers much better security than most common production VoIP deployments, which are often unencrypted and unauthenticated; when they are protected it is usually only in a hop-by-hop fashion by secure tunnels between proxies. Our VoCCN implementation instead provides true end-to-end security.

4.2 Performance

Subjective voice call quality of our VoCCN implementation was quite good (as measured between two reasonably fast workstations, on either 100Mbs or 1Gbs switched networks).

To evaluate CCN's ability to support timely delivery of realtime data we looked at the arrival times of packets in our VoCCN implementation. Figure 5 shows the distribution of inter-packet intervals, effectively packet *jitter*, for a 10-minute voice call made using stock Linphone (solid line) over UDP RTP, and our Linphone-based VoCCN client (dashed line) (expected interval 20 msec). Steplike appearance is due to the Linux kernel scheduling quantum (for a single process in the case of stock Linphone, and 3 processes for VoCCN). The VoCCN call has slightly fewer packets at or below the expected inter-packet interval, and a small number of long-interval packets at the tail. No packets were lost

[1]This provided both encryption and content authentication; though the latter was redundant given CCN's digital signature on each packet.

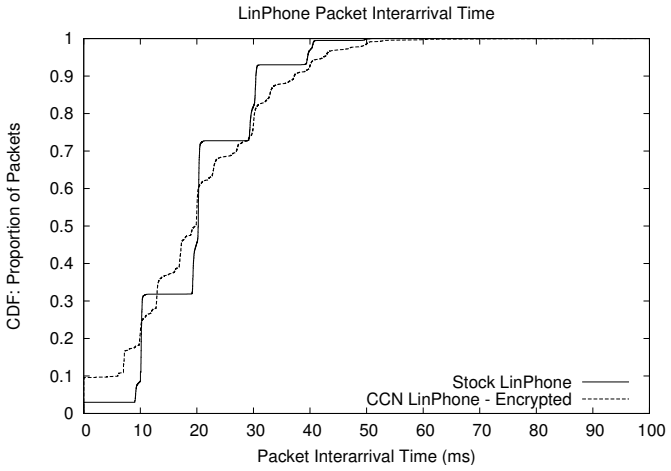

Figure 5: Cumulative distribution of inter-packet intervals, or jitter, for a 10-minute voice call.

by either stock Linphone or our VoCCN client, however a small number of VoCCN packets (less than 0.1%) were dropped by Linphone for having arrived too late.

5. CONCLUSIONS

Content-oriented network architectures not only move content scalably and efficiently, they can also implement IP-like conversational services like voice calls, email or transactions. To demonstrate this we have implemented and tested a Voice-over-CCN prototype. The result is functionally and performance equivalent to Voice-over-IP but substantially simpler in architecture, implementation and configuration. Because it does not require VoIP's inbound SIP proxy (with the associated signaling state concentration), it is intrinsically more scalable. Because CCN secures content rather than the connections it travels over, VoCCN does not require delegation of either trust or keys to proxies or other network intermediaries and thus is far more secure than VoIP. Finally, thanks to certain affordances offered by CCN's structured naming, VoCCN is completely interoperable with VoIP via simple, stateless gateways.

6. REFERENCES

[1] W. Adjie-Winoto, E. Schwartz, H. Balakrishnan, and J. Lilley. The Design and Implementation of an Intentional Naming System. *SIGOPS Oper. Syst. Rev.*, 33(5):186–201, 1999.

[2] B. Ahlgren, M. D'Ambrosio, M. Marchisio, I. Marsh, C. Dannewitz, B. Ohlman, K. Pentikousis, O. Strandberg, R. Rembarz, and V. Vercellone. Design considerations for a network of information. In *CONEXT*, 2008.

[3] H. Balakrishnan, K. Lakshminarayanan, S. Ratnasamy, S. Shenker, I. Stoica, and

M. Walfish. A Layered Naming Architecture for the Internet. In *SIGCOMM*, 2004.

[4] H. Balakrishnan, S. Shenker, and M. Walfish. Semantic-Free Referencing in Linked Distributed Systems. In *IPTPS*, February 2003.

[5] M. Caesar, T. Condie, J. Kannan, K. Lakshminarayanan, I. Stoica, and S. Shenker. ROFL: Routing on Flat Labels. In *SIGCOMM*, 2006.

[6] C. Dannewitz, K. Pentikousis, R. Rembarz, E. Renault, O. Strandberg, and J. Ubillos. Scenarios and research issues for a network of information. In *Proc. 4th Int. Mobile Multimedia Communications Conf.*, July 2008.

[7] C. Esteve, F. L. Verdi, and M. F. Magalhães. Towards a new generation of information-oriented internetworking architectures. In *CONEXT*, 2008.

[8] M. Gritter and D. R. Cheriton. An architecture for content routing support in the internet. In *Usenix Symposium on Internet Technologies and Systems (USITS)*, 2001.

[9] IETF. Datagram transport layer security.

[10] IETF. RFC 3261 – SIP: Session initiation protocol.

[11] IETF. RFC 3711 – The Secure Real-time Transport Protocol (SRTP).

[12] IETF. RFC 3830 – MIKEY: Multimedia Internet KEYing.

[13] V. Jacobson. A new way to look at networking, August 2006. `http://video.google.com/videoplay?docid=-6972678839686672840&ei=iUx3SajYAZPiqQLwjIS7BQ&q=tech+talks+van+jacobson`.

[14] V. Jacobson. Making the case for content-centric networking: An interview with Van Jacobson. *ACM Queue*, January 2009.

[15] V. Jacobson. Special plenary invited short course: (CCN) content-centric networking. In *Future Internet Summer School*, August 2009.

[16] V. Jacobson, D. K. Smetters, J. D. Thornton, M. Plass, N. Briggs, and R. Braynard. Networking named content. In *In submission to CoNext '09*, 2009.

[17] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica. A Data-Oriented (and Beyond) Network Architecture. In *SIGCOMM*, 2007.

[18] B. Ohlman et al. First NetInf architecture description, April 2009. `http://www.4ward-project.eu/index.php?s=file_download&id=39`.

[19] PARC. CCN toolkit. `http://⟨TBD⟩`, September 2009. URL will be available by final version of paper.

[20] RTP. RFC 1889 – RTP: A transport protocol for real-time applications.

[21] D. Trossen. Conceptual architecture of PSIRP including subcomponent descriptions (D2.2), June 2008. `http://psirp.org/publications`.

[22] M. Walfish, H. Balakrishnan, and S. Shenker. Untangling the Web from DNS. In *NSDI*, 2004.

[23] P. Zimmermann, A. Johnston, and J. Callas. ZRTP: Media Path Key Agreement for Secure RTP. `http://www.philzimmermann.com/zfone/draft-zimmermann-avt-zrtp-15.html`.