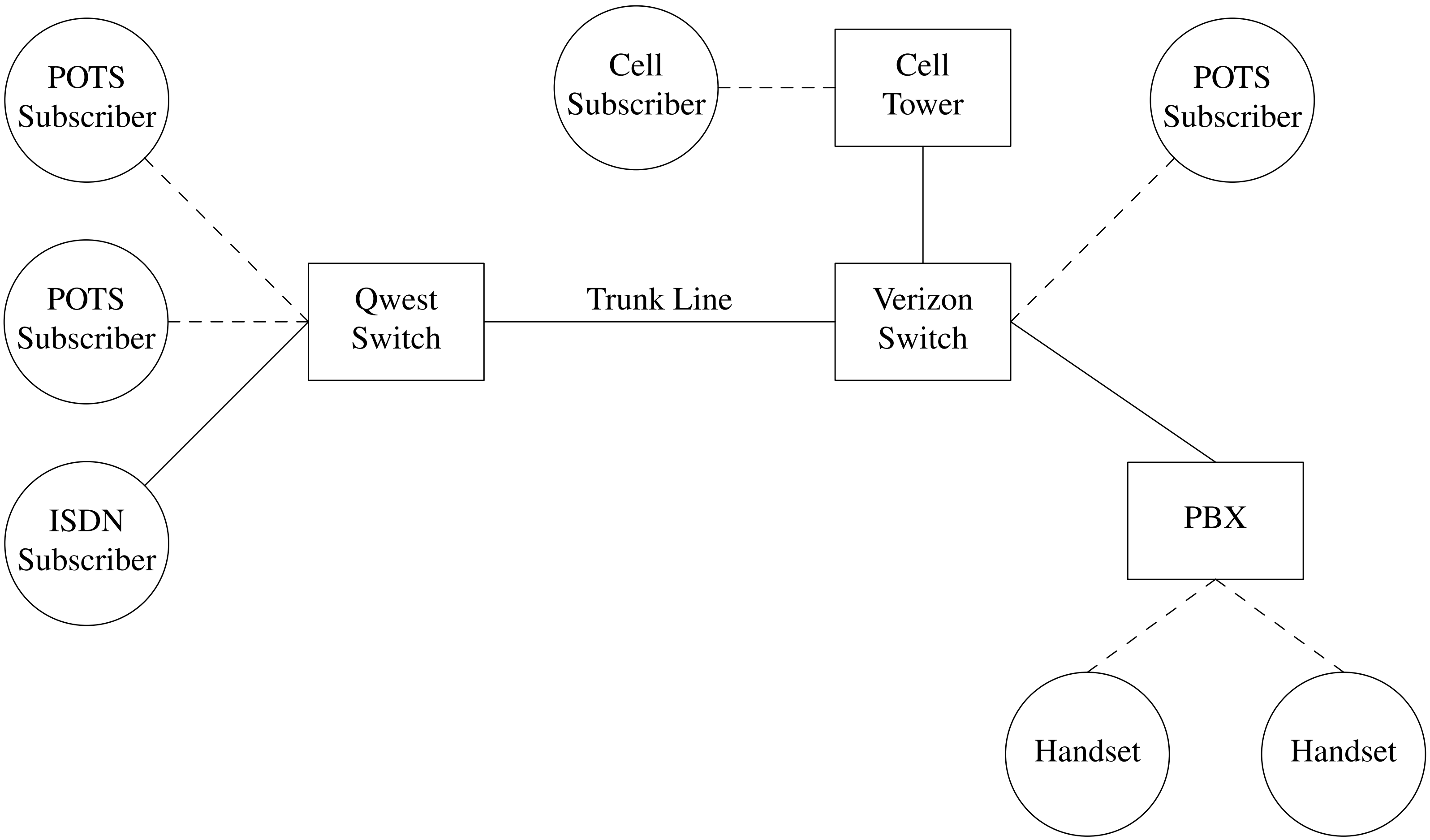
Security in VoIP Systems

Eric Rescorla
RTFM, Inc.
`ekr@rtfm.com`

Background: the PSTN



Plain Old Telephone Service(POTS)

- This is what you probably have
- Analog transmission
  - A pair of copper wires from you to the CO
- All signalling is inband
  - Instructions from you to the switch are DTMF tones
  - From the switch to you is tones (e.g., caller ID)
- Basically no security
  - Wiretapping means a pair of alligator clips and a speaker
  - Hijacking is just as easy

Digital Telephony

- Used for
  - Trunk lines between switches
  - Digital service to subscribers (ISDN BRI)
  - PBXs for enterprised (ISDN PRI, typically)
- Signalling System 7 (SS7)
  - Digital control and signalling protocol
  - Used between the switches
    - Reduced version (Q.931) used for communication with ISDN phones and PBXs
- Security is based on transitive trust
  - If you're on the SS7 network you're trusted
  - Example: Caller ID forgery

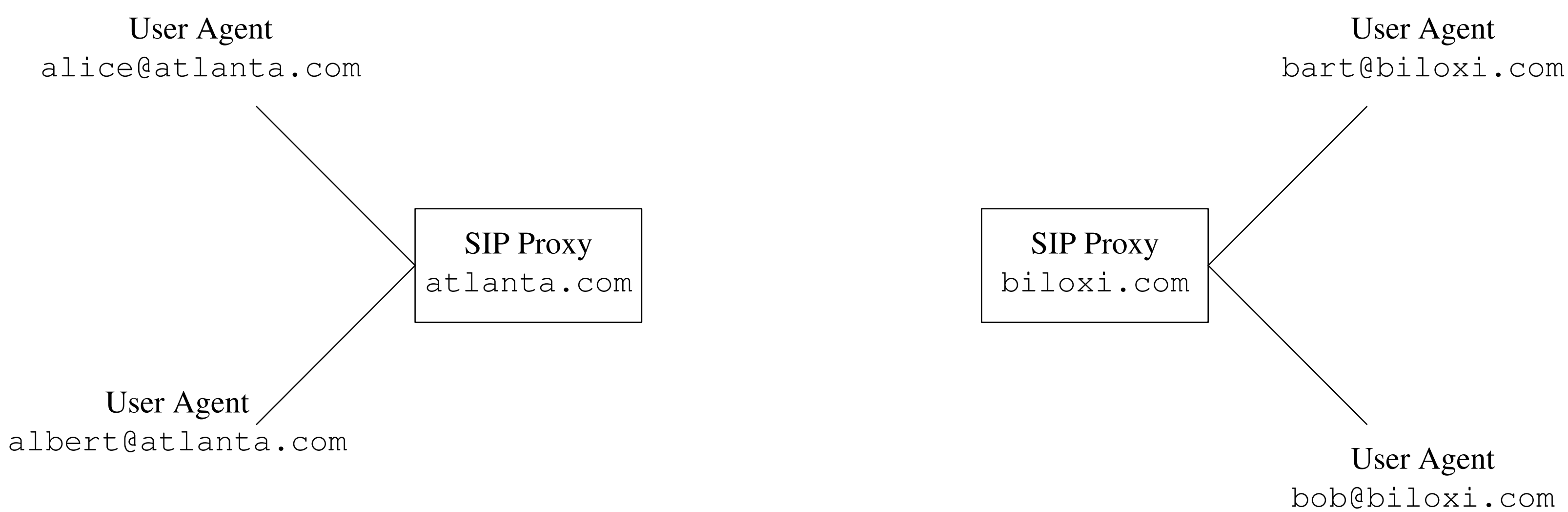
What about cellular?

- Currently: closed system with digital transmission
  - Some weak crypto between handset and base station
  - Phones are not trusted
- Future: IP system running SIP
  - 3GPP Internet Multimedia Subsystem
  - Not really compatible with IETF SIP
  - Not clear if this is going to happen

Why is VoIP Complicated?

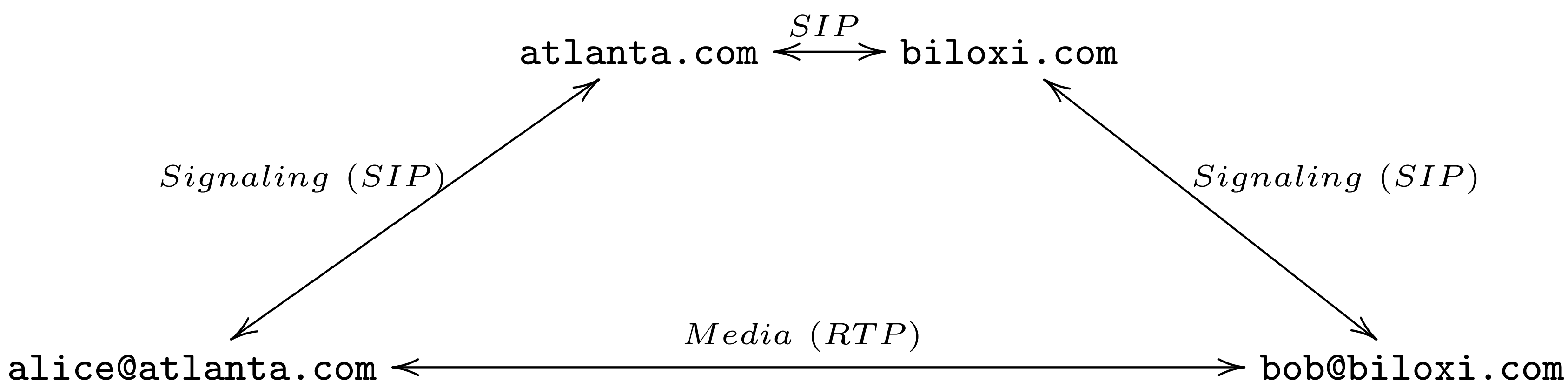
- Just connect to the callee and start talking, right?
  - Not quite so easy
- Challenges
  - Naming
  - Name resolution
  - Rendezvous
  - NAT/Firewall traversal
  - Multiple devices/voice mail
  - Retargeting

SIP [RSC+02] Topology



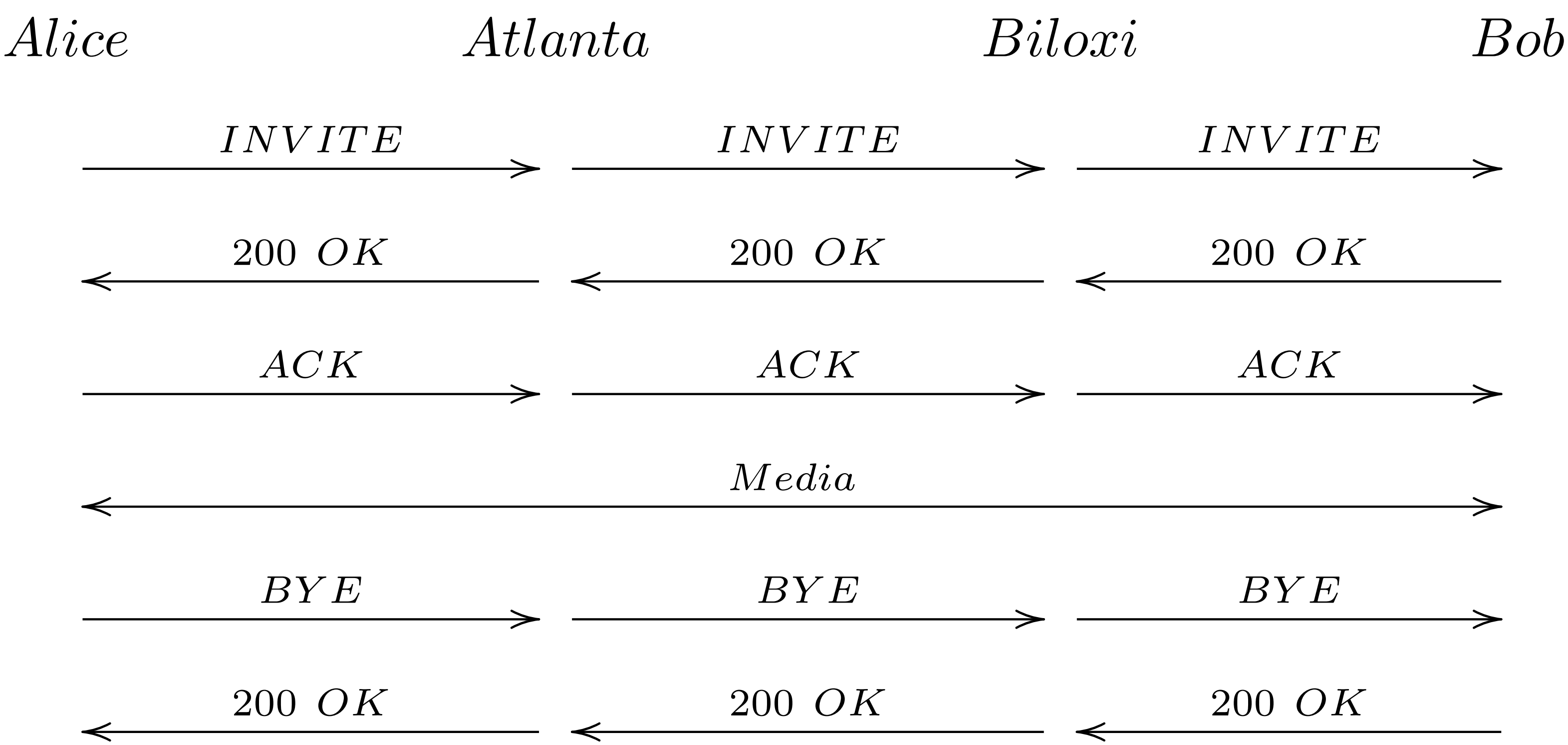
- Each user is associated with a given proxy
  - Like email and email servers
  - To reach `alice` contact `atlanta.com`
- The provider doesn't (necessarily) control the access network

Basic SIP Interaction



- Signalling goes through proxies
  - Rendezvous
  - NAT/Firewall traversal
  - Support for offline user agents
- Media goes directly
  - For performance reasons

Typical SIP Callflow



- INVITE and OK contain the media parameters
  - Ports, codecs, etc.

Example SIP INVITE

```
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bK776asdhds
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710@pc33.atlanta.com
CSeq: 314159 INVITE
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/sdp
Content-Length: XXX

v=0
o=carol 28908764872 28908764872 IN IP4 100.3.6.6
s=-
t=0 0
c=IN IP4 192.0.2.4
m=audio 0 RTP/AVP 0 1 3
a=rtpmap:0 PCMU/8000
a=rtpmap:1 1016/8000
a=rtpmap:3 GSM/8000
```

Security Requirements

1. Allow system provider to know and control who is using the system
2. Calls meant for me are not delivered to other people
3. Allows users to know who they are talking to
4. Only parties you want to be talking to can participate in/hear the conversation.
5. Allow users to hide who or where they are from people they are talking to.
6. Provide ways to mitigate unwanted communications such as telemarketing.

Why control access at all?

- The bits don't cost the provider anything
  - Remember, he doesn't run the access network
  - Note: IMS is different here
- What does cost money?
  - Running the servers (remember, they need to be reliable)
  - Gatewaying to the PSTN
  - Running media relays
- Stop other people from posing as you
- Usual profit motive

User Authentication

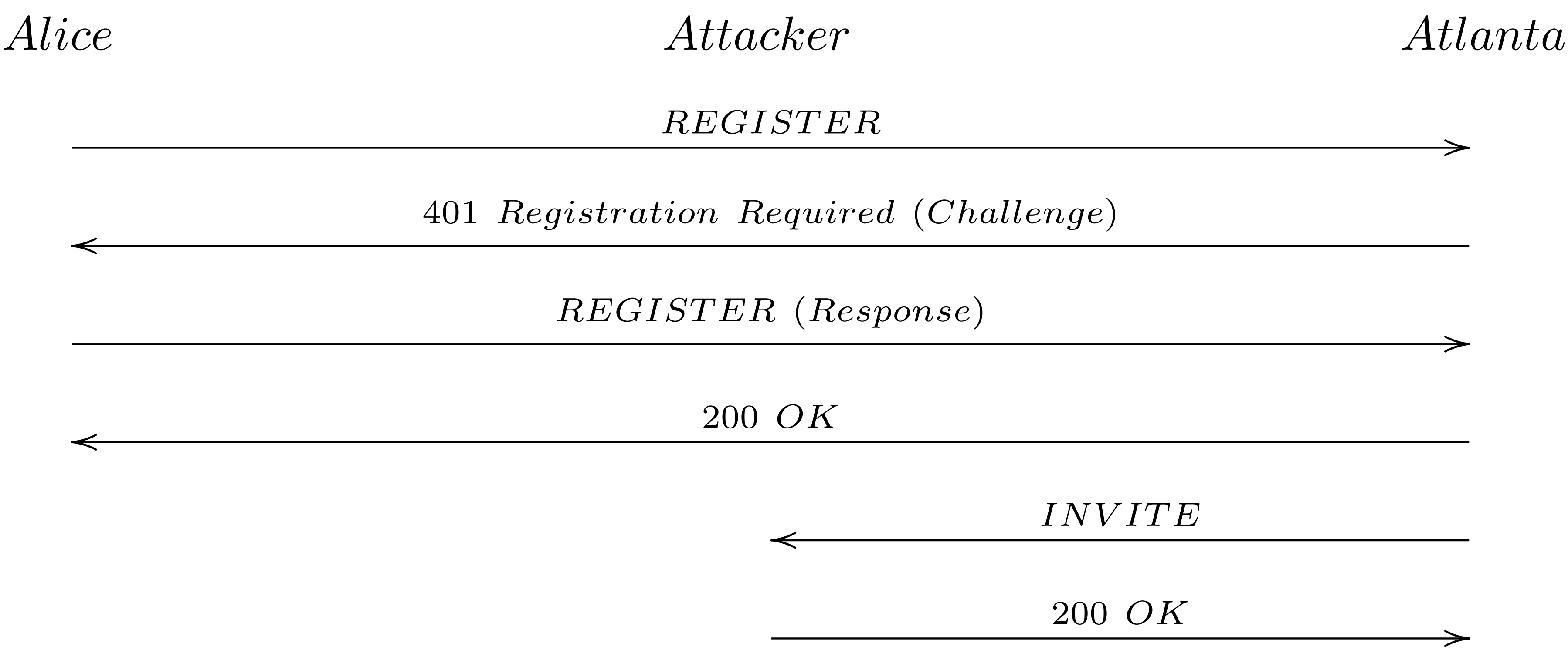
- Every user has an account with a username and password
  - This happens out of band
- User agent authenticates to the proxy (server)
  - This uses "Digest Authentication" (challenge response)
  - The server can challenge any request from the client

Digest Security Properties

- Client authentication only
  - No server
  - And only for requests
- Integrity for the request URI
  - And optionally the body
- Dictionary attacks
- No confidentiality

Insecure Transport

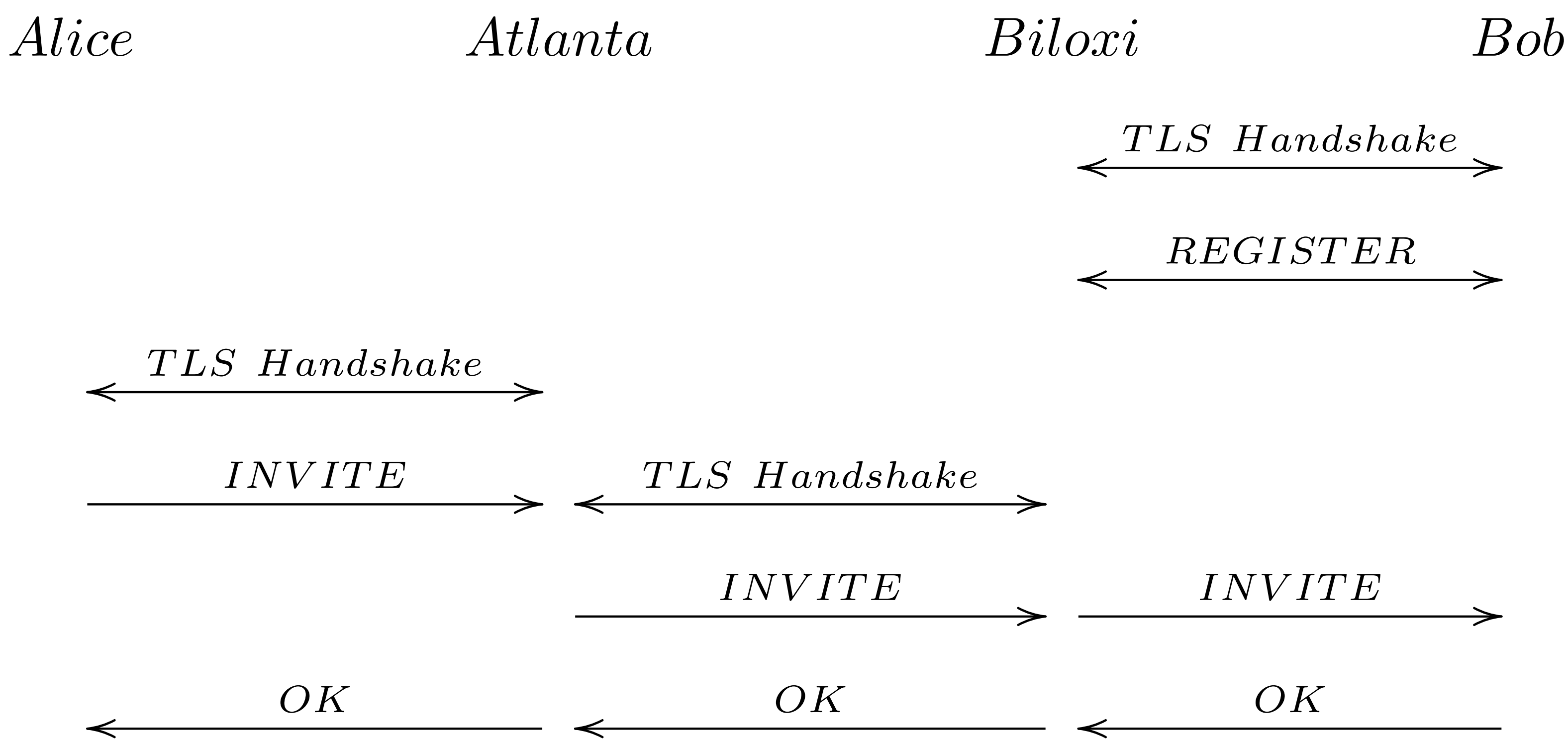
- Digest only provides security for the individual request
  - Any request can be authenticated
  - What about other messages?



Using TLS [DR08] with SIP (Client side)

- Server is issued a certificate
  - Identity is the server's domain name (e.g., `sip.example.com`)
- Client is configured with the name of the server
  - TLS connects to server
  - Compares server certificate to the expected name
- Security properties
  - Client is able to authenticate server
  - Server authenticates client with Digest
    - TLS lets you leverage this authentication across requests

Typical Callflow with TLS



Why not TLS mutual auth?

- TLS offers a mode for certificate client authentication
  - Why bother with passwords, digest, etc.?
- User-level certificate deployment is prohibitive [Gut03]
  - Conceptually complex
  - Bad UI from CAs
  - CA vendor lockin

PKI Structural Mismatch

- Your identity is `vern@cs.berkeley.edu`
- Who assigned that identity?
  - UC Berkeley did
- But VeriSign (or any other CA) wants to issue you a cert with your whole identity
  - How do they know who you are?
  - They need to ask Berkeley somehow (not convenient)
- Berkeley should be a CA
  - CAs don't want this (revenue preservation)
  - CAs are hard to operate
- This basic constraint impacts the rest of the system (more later)

Proxy to Proxy Authentication

- TLS with mutual authentication
- There's an asymmetry here
  - The "client" knows who is trying to connect to
    - Check the certificate against expectations
  - The "server" just knows somebody connected
    - Extract the identity from the certificate
    - Cache to avoid connection in reverse direction
- This is just hop-by-hop

Transitive Trust

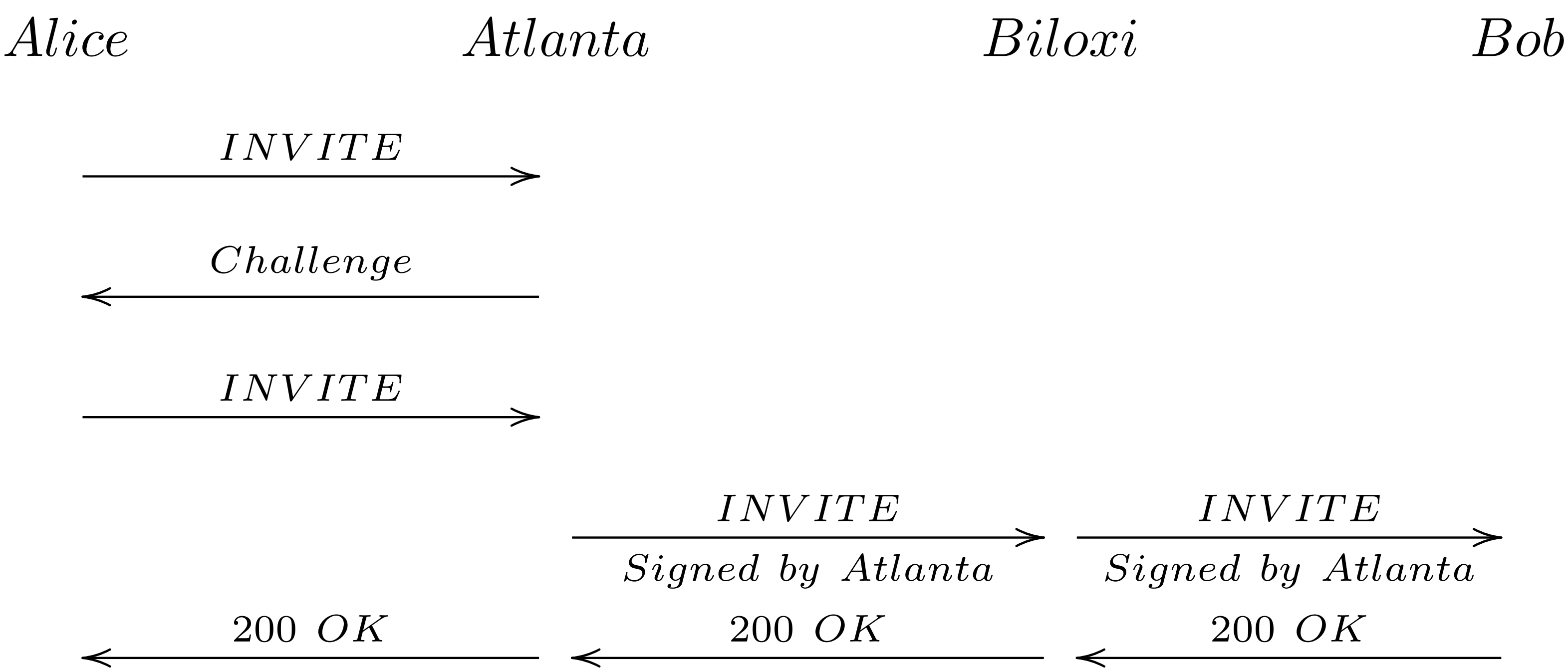
- Bob cannot verify that Alice sent him a message
  - He knows that: Biloxi says that Atlanta says that Alice sent this message
- Malicious proxies can
  - Forge messages
  - Reroute messages
  - Change message contents
- Why is this bad?
  - Topologies with untrusted proxies
  - VSPs with complext internal structure
  - Lawful intercept (bad from end-user's perspective)

Failed Approaches

- These issues were known when SIP was designed
- SIP includes support for end-to-end security using S/MIME, OpenPGP
  - Digital signatures by the UAs on each message
  - Encryption of messages between UAs
- This stuff utterly failed
  - S/MIME required end-user certificates
    - Which nobody has
  - Complicated to implement and understand
  - ... ASN.1 allergy

SIP "Identity"

- End-users don't have certificates
  - But servers do (for TLS)
- We can leverage this
  - User's local server signs an assertion of their identity



Security Properties of SIP Identity

- Signed assertion that user sent this message
- Traceable back to server certificate
  - `alice@example.com` is signed by `example.com`
- Signature covers some of header and all of body (media parameters)
  - Some headers are changed by proxies in transit
- Some replay prevention
  - Timestamps, unique IDs in messages
- You need to trust the signing server
  - But it controls the namespace anyway
- Effectively caller-ID for VoIP
  - Doesn't work well for E.164 numbers

Media Security

- We've just secured the signaling
  - But that just sets up the call
- What about the media?
  - Wiretapping—listen to the media traffic
  - Hijacking—divert traffic somewhere else
- We need security for the media as well
  - Leverage secure signaling to get secure media

Why is encrypting Media hard?

- Very tight performance constraints
  - Custom tuned *Real-time Transfer Protocol* (RTP)
  - Packets are very small (but very frequent)
  - Per-packet overhead is important
- What's wrong with generic protocols (TLS, DTLS, IPsec)
  - Must work with any kind of payload
  - Must carry meta-information (sequence numbers, IVs, etc.)
  - Result: lots of overhead (20-40 bytes per packet)
- We can do a lot better with a custom protocol

Example: a TLS record (CBC mode)

Header (5 bytes)	IV (16 bytes)	Data (variable)	MAC (10-20 bytes)	Padding (1-16 bytes)

- Contents
  - Header: type, version, length
  - IV: per-packet distinguisher
  - MAC: the integrity check
  - Padding: to fill out the cipher block
- Overhead: 21-56 bytes
  - The best generic TLS mode (GCM) has 13 bytes of overhead
  - DTLS has an extra 8 bytes for sequence number

A Split Architecture

- Media transport security protocol (Secure RTP) [BMN+04]
  - This is a well understood problem
  - Optimized for minimal overhead
  - Assumes the existence of some key management protocol
- Key management protocol
  - This is less well understood
  - Several false starts
  - Finally getting traction with DTLS-SRTP
- This is a typical IETF divide and conquer approach

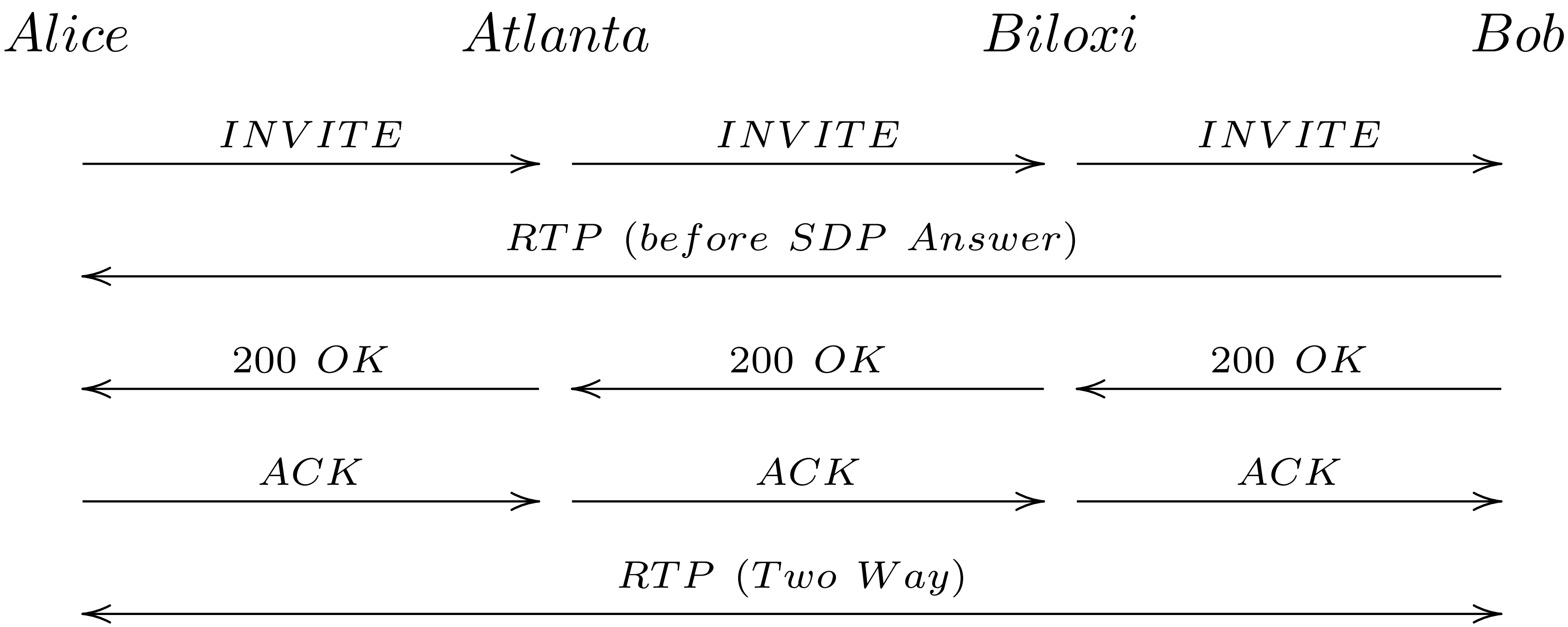
An SRTP Packet

```
     0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<+
  |V=2|P|X|  CC   |M|     PT      |       sequence number         | |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |
  |                           timestamp                           | |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |
  |           synchronization source (SSRC) identifier            | |
  +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+ |
  |            contributing source (CSRC) identifiers             | |
  |                               ....                            | |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |
  |                   RTP extension (OPTIONAL)                    | |
+>+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |
| |                          payload  ...                         | |
| |                               +-------------------------------+ |
| |                               | RTP padding   | RTP pad count | |
+>+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<+
| ~                     SRTP MKI (OPTIONAL)                       ~ |
| +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |
| :                 authentication tag (RECOMMENDED)              : |
| +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |
|                                                                   |
+- Encrypted Portion*                      Authenticated Portion ---+
```

Why is key management for SRTP hard

- We can't completely trust the signaling
  - Identity provides authentication
  - But any proxy can read the SIP messages
- Users won't have their own certificates
  - So classic PKI-based protocols don't work
- It needs to be fast
  - So key management in the signaling is a problem
- Plus there are some weird edge cases (forking, retargeting, etc.)

Early Media



- Used to send ringback tone
- Also for IVR prompts

Unsuccessful Key Management Approaches

- MIKEY
  - Key exchange in the signalling protocol
  - Performance problems, race conditions, etc.
- SDES
  - A naked key in the signalling protocol
  - Insecure against attack by any proxy
  - Would have been fine if we had S/MIME encryption

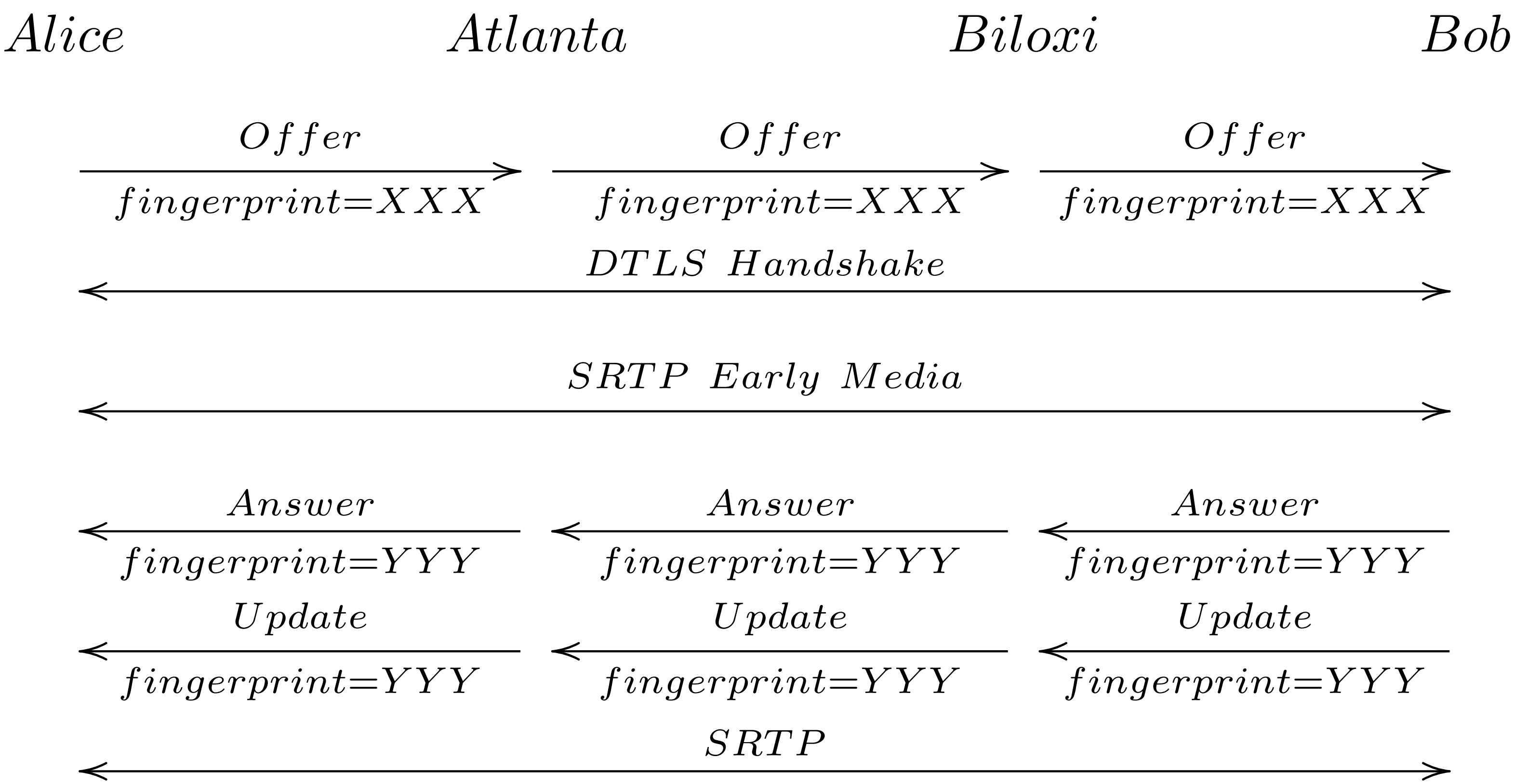
DTLS-SRTP [FTR08]

- Use DTLS for key management
  - What's DTLS? TLS with some modifications for UDP
  - But not for encrypting RTP
- DTLS outputs a key
  - Which we pass to SRTP
  - Encrypt the data with SRTP
- A bit of a hack
  - But cheaper than inventing a whole new key management protocol
- Finally pretty much done
  - Approved by IESG (in RFC publication queue)
  - Starting to appear in products

But how does the authentication part work?

- (D)TLS depends on certificates
  - And we just said there weren't any
- Leverage the signalling
  - Which can be authenticated via Identity
  - Each side generates a self-signed certificate
  - Certificate fingerprints (hashes) go in SIP
- Endpoints compare the fingerprints to the DTLS certificates

DTLS-SRTP Overview

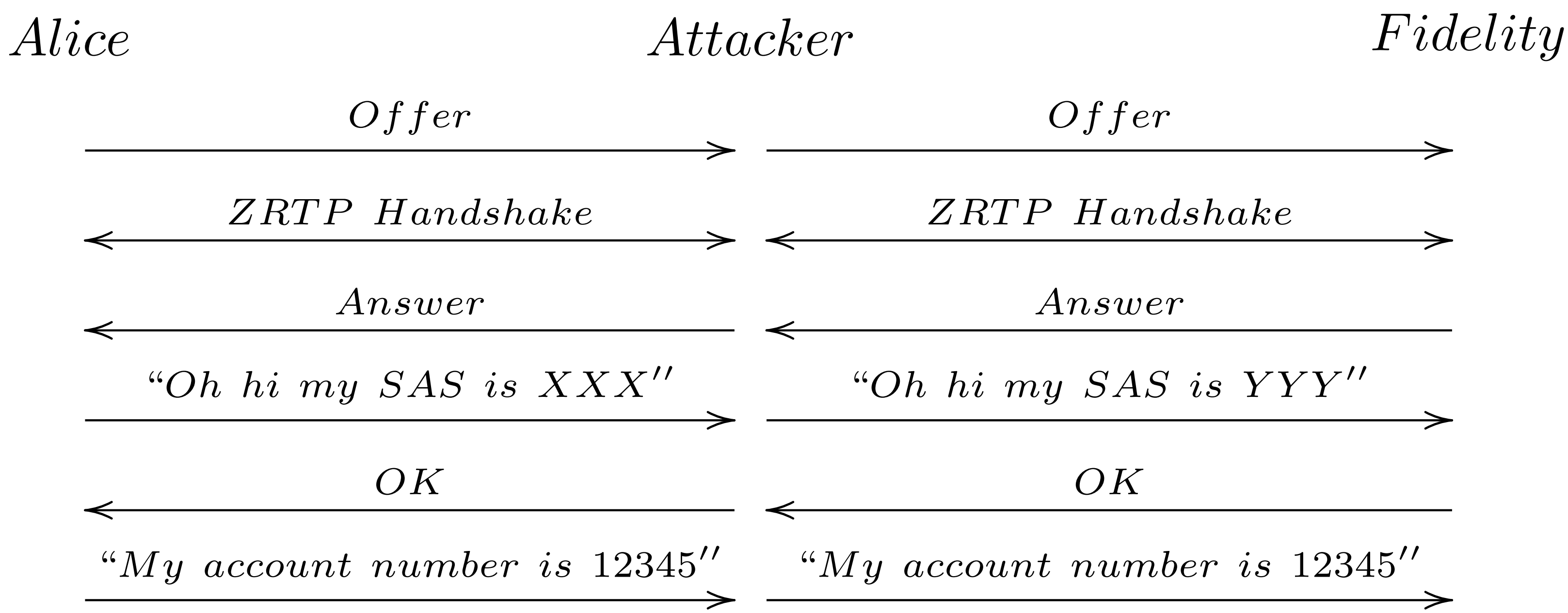


- Fingerprints are protected via Identity

ZRTP [ZJC08]

- Designed by Phil Zimmermann
- Perform a cryptographic handshake
- Authenticate the handshake over a voice channel
  - By means of a "short authentication string"
- Unfortunately this isn't secure in many settings
  - Very susceptible to MITM attacks when calling people you don't know
  - Cut-and-paste attacks on the authentication string
  - This assumes people will read the authenticator anyway [WT99]
- Doesn't work when gatewaying to the PSTN
- Probably useful in limited settings

Impersonation Attacks



- No way to distinguish an attacker from a legitimate answerer
  - How do you know what Fidelity’s CSR sounds like
  - The voice sounds the same throughout the call
  - Even easier to clone an IVR system
- This is a variant of the classic “mafia attack” [DGB87]

Cut-and-Paste Attacks

- SAS has a limited coding space (32 symbols)
- People will happily read their SAS to you
  - You get 4 symbols per call
  - 15 calls $\rightarrow$ 85% of symbols
  - 85% of symbols $\rightarrow$ 52% forgery probability
- Base-256 works better
  - But attack still possible
  - Especially on IVR

Privacy Motivations

- Domestic violence shelters
  - Call home without giving out location
- Whistleblowers
  - Enable anonymous tips
- National security
  - During 9/11 Cheney needed to call from a secure location

Types of privacy

- Identity privacy
  - PSTN: Caller-ID
  - VoIP: From field (and others) contain your AOR
  - We want the equivalent of Caller-ID blocking for VoIP
- Location privacy
  - PSTN: your phone number is your location (or at least your phone)
  - VoIP: your VoIP packets contain your IP address
    - Otherwise you couldn't talk to anyone
  - This is harder to hide

Anonymous From Field

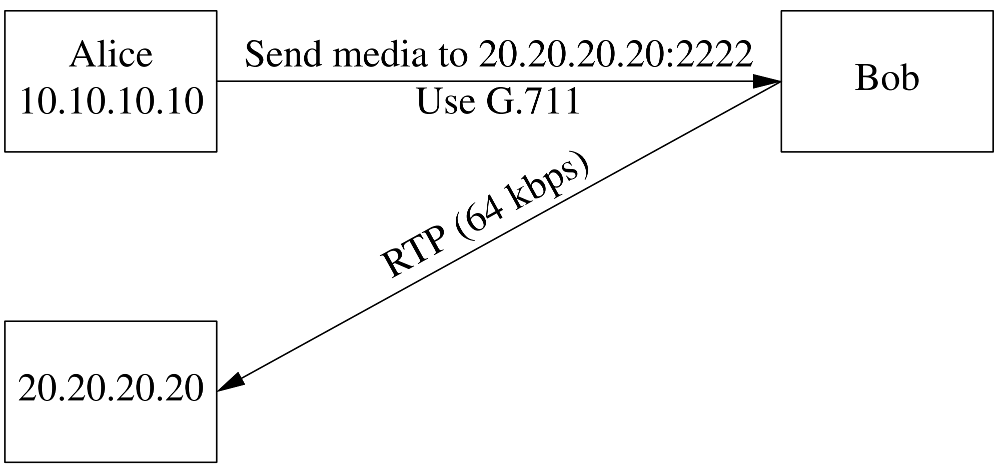
- The SIP “From” field contains your identity
  - e.g., `sip:ekr@rtfm.com`
  - But it isn’t used for anything
- You can use an anonymous From
  - `sip:anonymous@anonymous.invalid`
- This hids your name
  - But your proxy still appears in Via headers
  - More useful if you have an account from a big provider
- Anonymizing services could do a better job

Anonymizing Services

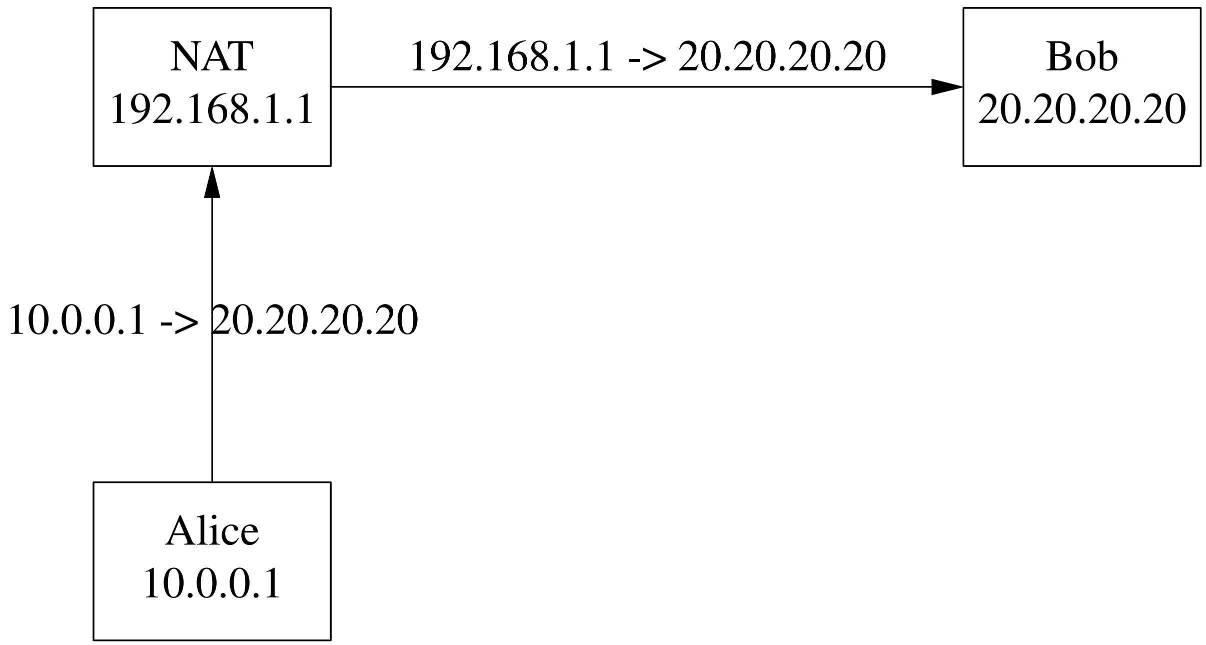
- Essentially a phone call forwarder
  - You call them
  - They call the callee
  - Forward signaling and media
- This provides good security against casual attackers
  - But records can be subpoenad
  - And an attacker who can see traffic coming and out of the service can trace you
- Multiple hops (onion routing) introduces serious latency issues
  - Especially if packets are intentionally retimed

Voice hammer

- During call setup, Alice tells Bob where to send media and vice versa
  - This is never checked
- Alice can make Bob flood anyone she chooses
  - Victim can send ICMP errors
  - But often ignored
- You need a positive acknowledgement

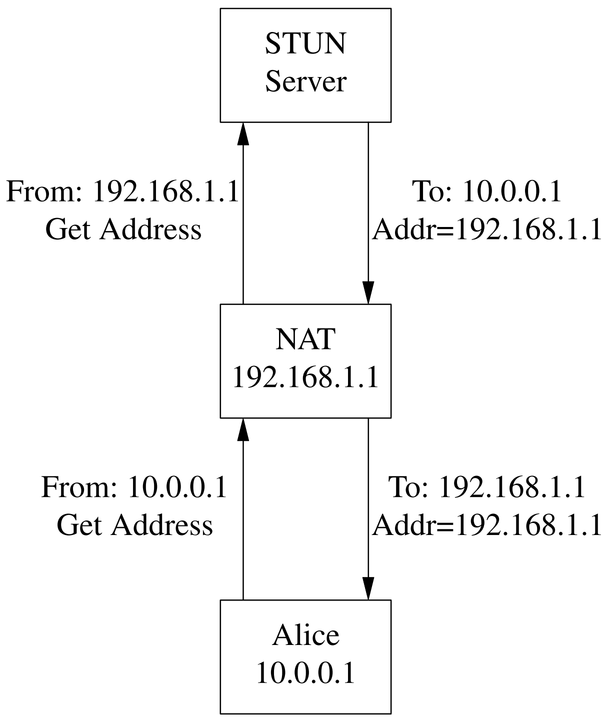


NATs and Media Setup



- Alice thinks her address is 10.0.0.1
- But it's being translated to 192.168.1.1
- Packets to 10.0.0.1 go nowhere

STUN [RMMW08]



- STUN servers allow NAT discovery
  - Simple server returns public address
- Problems
  - What if the server lies (voice hammer again)?
  - Multiple layers of NAT
  - NATs with aggressive filtering
  - Media relaying
- Bottom line: only sort of works

Interactive Connectivity Establishment (ICE) [Ros07]

- Collect all possible addresses
  - Local
  - From STUN servers
  - From media relays, VPNs, etc.
  - Send all addresses to the peer
- Each peer tries all possible combinations
  - Send a request/response on the address pair
  - Pick the best one that works (you get a response)
- This is fiendishly complicated
  - But it does work
  - Stops voice hammer

Skype

- Closed system
  - Single vendor
  - Proprietary protocols
  - Clients are hardened against reverse engineering
- Nevertheless some stuff is known
  - Commissioned analysis by Berson [Ber05]
  - Baset and Schulzrinne analyzed network traffic [BSer]
  - Biondi and Desclaux reverse engineered the client [BD06]

General Architecture

- Central enrollment server
- Acts as CA
  - Guarantees unique identity
  - Hands out certificates signed by Skype
- Advertised as peer-to-peer
  - Supernodes used for NAT traversal
  - Unclear what fraction of supernodes are independent

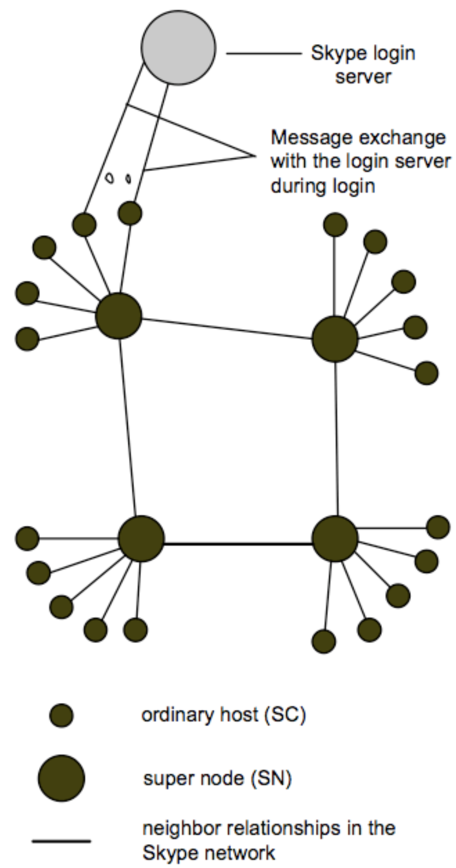


Diagram from Baset and Schulzrinne [BSer].

Per-Call Security

- Session establishment establishes session keys
  - Proprietary handshake
  - Authenticated with user certificates
- Traffic encrypted with counter mode (Berson)
  - Berson reports a weak integrity check (CRC)
  - This was in '05, maybe it's fixed
- Traffic encrypted with RC4 (Biondi and Desclaux)
  - Reuse of RC4 blocks?
- Details are fuzzy here

Insider attacks

- Skype is the CA
  - And they control the software
- This gives them several insider attack opportunities
  - Issue fake certificates and allow a MITM
    - Detectable by key caching?
    - Biondi and Desclaux imply new key per connection
  - Backdoor the client to leak keying material
    - Automatic checking for newer versions helps here
- Direct consequence of this being a closed system

Skype Lock-In

- Skype wants you to use their client
  - Branding, control, avoid free-riding
  - Enforced via protocol secrecy
- Extensive reverse engineering countermeasures
  - Code obfuscation
  - Binary encryption
  - Binary packing
  - Checksums to prevent code modification
- None of this is required in an open system
- This looks a lot like malware!

Should we expect VoIP spam?

- Yes
- There is already spam on the PSTN
  - We call it telemarketing
- Spam is a big problem in email systems
  - Because it's so cheap to send
- Why should VoIP be any different?

Why is VoIP spam hard?

- Decisions need to be made in real-time
  - Can't take two minutes to decide if something is spam
- No material for content analysis
  - Most email filters look at the body of the message
  - But with VoIP all the content is in the audio
- Unwanted phone calls are more annoying than e-mail

Candidate Approaches

- White listing
- Reputation systems
- Reverse Turing Tests & CAPTCHAs
- Payments at risk
- Traffic analysis
- Legal action

- Almost no VoIP spam to speak of
- Hard to know what will work

Summary

- VoIP is a very complicated system
  - SIP is by far the most complicated open system people have tried to secure
  - Skype much easier because it's closed
- Not one security system
  - A lot of interlocking pieces
- Everything is based on securing the end-proxies
  - This uses well-understood technologies (certs, TLS)
- The traditional COMSEC stuff is mostly well understood
  - After some false starts
- Spam, spit, etc. are a real challenge

References

[BD06] Philippe Biondi and Fabrice Desclaux. Silver Needle in the Skype. Black Hat Europe, March 2006.

[Ber05] Tom Berson. Skype Security Evaluation, October 2005.

[BMN+04] M. Baugher, D. McGrew, M. Naslund, E. Carrara, and K. Norrman. The Secure Real-time Transport Protocol (SRTP). RFC 3711, Internet Engineering Task Force, March 2004.

[BSer] Salman A. Baset and Henning Schulzrinne. An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol, 2004 September.

[DGB87] Y. Desmedt, C. Goutier, and S. Bengio. Special uses and abuses of the fiat-shamir passport protocol, 1987.

[DR08] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246, Internet Engineering Task Force, August 2008.

[FTR08] J. Fischl, H. Tschofenig, and E. Rescorla. Framework for Establishing an SRTP Security Context using DTLS. Internet-Draft draft-ietf-sip-dtls-srtp-framework-03, Internet Engineering Task Force, August 2008. Work in progress.

[Gut03] Peter Gutmann. Plug-and-Play PKI: A PKI your Mother can Use. In *Proceedings of the 12th USENIX Security Symposium*, 2003.

[RMMW08] J. Rosenberg, R. Mahy, P. Matthews, and D. Wing. Session Traversal Utilities for (NAT) (STUN). Internet-Draft draft-ietf-behave-rfc3489bis-18, Internet Engineering Task Force, July 2008. Work in progress.

[Ros07] J. Rosenberg. Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols. Internet-Draft draft-ietf-mmusic-ice-19, Internet Engineering Task Force, October 2007. Work in progress.

[RSC+02] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol. RFC 3261, Internet Engineering Task Force, June 2002.

[WT99] A. Whitten and J.D. Tygar. Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0. In *Proceedings of the 8th USENIX Security Symposium*, August 1999.

[ZJC08] P. Zimmermann, A. Johnston, and J. Callas. ZRTP: Media Path Key Agreement for Secure RTP. Internet-Draft draft-zimmermann-avt-zrtp-09, Internet Engineering Task Force, September 2008. Work in progress.