P2PSIP, ICE, WEBRTC, AND SIP EXTENSIONS

T-110.5150
APPLICATIONS AND SERVICES IN INTERNET
OCTOBER 29TH, 2013

Jouni Mäenpää, Dr.Sc.(Tech)
NomadicLab
Ericsson Research Multimedia Technologies
Communication Services
AGENDA

› Extending SIP
› IP Multimedia Subsystem (IMS)
› Peer-to-Peer SIP (P2PSIP)
› Interactive Connectivity Establishment (ICE)
› Web Real-Time Communication (WebRTC)
› (Examples of SIP extensions)
REAL-TIME COMMUNICATION

WEBRTC

SIP & SDP

SIP PROXY-REGISTRAR

SIP & SDP

RTP & RTCP & ICE

NAT

SIP: Session Initiation Protocol
SDP: Session Description Protocol
ICE: Interactive Connectivity Establishment
RTP: Real-Time Protocol
RTCP: RTP Control Protocol
P2PSIP: Peer-to-Peer SIP
WebRTC: Web-Real-Time Communication
Global interoperability possible since the core functionality of SIP as specified in RFC 3261 is present in every implementation
- A given SIP application can always assume that another SIP application is able to understand the core protocol

However, many implementations require functionality beyond the core protocol
- Thus, extensions are required
- SIP is flexible and easy to extend

Use of extensions can be negotiated during session establishment
- Two things are negotiated: the extensions the remote party supports and the extensions that will actually be employed in the session
SIP Extension Negotiation Mechanism

- Three header fields: **Require**, **Supported**, and **Unsupported**
- When a dialog is being established, the UAC lists
  - The names of the extensions it wants to use in a **Require** header field
  - The names of the extensions it supports in a **Supported** header field
  - The **Unsupported** header field is used in error responses
- The UAS can also request extra extensions
- **Proxy-Require** header field can be used to require support of extensions from proxies
- The extensions that a proxy or another UA supports can be queried by using an **OPTIONS** method
- The names of extensions are referred to as **option tags**
SIP EXTENSION NEGOTIATION MECHANISM

1. INVITE
   Require: ext1, ext2
   Supported: ext3, ext4

2. 200 OK
   Require: ext3

3. ACK

Alice wants to use extensions ext1 and ext2. In addition to these, Alice supports ext3 and ext4.

Bob wants to use an extra extension, ext3.
In a SIP dialog, UAs need to know which methods the other end understands
- An **Allow** header field lists all the methods a UA supports

```
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE
```

However, the **Allow** header field can’t be used to express that a method is **required**
- Instead, an option tag associated with the method can be used

- Processing of unknown methods and header fields:
  - **Proxies** forward unknown methods and header fields
  - **Redirect servers** ignore unknown header fields, methods and option tags
  - **UASs** ignore unknown header fields and reject unknown methods
Examples of SIP Extensions

- Reliability of provisional responses (RFC 3262)
- SIP-specific Event Notification (RFC 3265)
- User agent capabilities (RFC 3840)
- Caller preferences (RFC 3841)
- Preconditions (RFC 3312, 4032)
- Signaling Compression (RFC 3320, 3486)
- Content Indirection
  - SIP REFER method
    - Refer peers to third parties (RFC 3515)
    - Can be used to implement e.g. call transfer
  - Instant messaging (RFC 3428)
    - The MESSAGE method allows the transfer of instant messages
  - SIP UPDATE method (RFC 3311)
    - Update the parameters of a session
  - Event state publication (RFC 3903)
    - The PUBLISH method to publish e.g. presence information
  - Session timers in SIP (RFC 4028)
    - Periodic refresh of SIP sessions
  - SIP INFO method (RFC 2976)
    - To carry session related control information generated during a session
    - E.g. carrying DTMF digits generated during a SIP session
  - And many others…
IP MULTIMEDIA SUBSYSTEM (IMS)

› Evolution of 3G mobile networks to deliver multimedia to mobile users
  – Core network evolution from circuit switching to packet switching
› Defined by 3rd Generation Partnership Project (3GPP)
› Uses IETF protocols (e.g., SIP, SDP, RTP, and RTCP)
  – IMS uses SIP as the main signaling protocol
› A collection of different functions linked by standardized interfaces
IMS ARCHITECTURE (SIMPLIFIED)

- **User Equipment (UE)**
- **Home Subscriber Server (HSS)**
- **Call Session Control Function (CSCF)**
  - P-CSCF
  - I-CSCF
  - S-CSCF
- **Media Resource Function (MRF)**
  - MRFC
  - MRFP
- **Application Server (AS)**
- **Interconnection Border Control Function (IBCF)**
- **Breakout Gateway Control Function (BGCF)**
- **IP Multimedia Gateway (IM-MGW)**
- **PSTN**
- **Another IMS Network**
- **Translation Gateway (TRGW)**
PEER-TO-PEER SESSION INITIATION PROTOCOL (P2PSIP)
PEER-TO-PEER SIP OVERVIEW

- Conventional **client/server SIP** relies on centralized proxy-registrar servers
- In Peer-to-Peer SIP (P2PSIP), SIP is used in an environment where the centralized functions are replaced by a **P2P overlay network**
- In the overlay network, address-of-record to contact URI mappings are distributed amongst the peers in the overlay
- P2PSIP is being standardized in the P2PSIP working group of the **IETF**
- "Standardized Skype"
Standardized in the **P2PSIP Working Group** (WG) of the IETF

The WG is responsible for:
- Defining concepts, terminology, rationale, and use cases for P2PSIP
- Standardizing a P2PSIP Peer and Client Protocols
- Producing a usage document for P2PSIP

Topics that are out of the scope of P2PSIP:
- Issues specific to applications other than locating users and resources for SIP-based communications and presence
- Research type of questions
- Locating resources based on something other than URIs
- Multicast and dynamic DNS based approaches as the core lookup mechanism
P2PSIP - CLARIFICATIONS

› **P2PSIP**
  - Not a protocol (despite of the name)
  - Rather, a decentralized VoIP service
  - Uses RELOAD as the P2P signaling protocol

› **RELOAD (REsource LOcation And Discovery)**
  - A generic P2P signaling protocol
  - A binary protocol
  - Provides an overlay network service
  - Used together with SIP to implement a P2PSIP VoIP service

› **SIP**
  - One of the application protocols using RELOAD
  - Uses RELOAD as a rendezvous service
    › Maps SIP AoRs (Address of Record) to Node-IDs
  - P2PSIP uses SIP for setting up communication sessions

› **Distributed Hash Table (DHT)**
  - An overlay algorithm
  - Used to organize peers in a P2P overlay network topology
  - RELOAD uses a DHT called Chord
**P2PSIP OVERLAY**

**P2PSIP Bootstrap Peer**

First point of contact for a peer joining the overlay. Can be located:
- By remembering peers from the last time the peer was in the overlay
- Through multicast discovery
- Through manual configuration
- By contacting a bootstrap server

**P2PSIP Peer**

- **<Node-ID>**
- **<Resource-ID>**
  - P2PSIP Resource Record

Interacts with the P2PSIP overlay through its associated peer using the Client Protocol. Does not run the distributed database algorithm.

Participates in the P2PSIP overlay and provides storage and transport services to other nodes.

**NAT**

**P2PSIP Peer**

**Peer Protocol**

The protocol spoken between P2PSIP Peers to share information and organize the P2PSIP Overlay network. A protocol called Resource Location and Discovery (RELOAD) is used as the peer protocol.

**P2PSIP Overlay**

**Joining peer** is a node attempting to become a P2PSIP Peer. Admitting peer helps the joining peer join the network.

**P2PSIP Client**

The protocol spoken between clients and peers. RELOAD is also used as the client protocol.

Interacts with the P2PSIP overlay through its associated peer using the Client Protocol. Does not run the distributed database algorithm.

**Enrollment Server**

Assigns Node-IDs and certificates.
P2PSIP peers are capable of performing operations such as:
- Joining and leaving
- Store and fetch
- Storing information on behalf of the overlay
- Transporting messages

**Joining**: to join a P2PSIP overlay, a joining peer needs to:
- Contact an **enrollment server**
  - To obtain an overlay configuration document, **certificate** and **Node-ID**
  - Central enrollment process vs. self-generated certificates
- Contact a **bootstrap peer**
  - The bootstrap peer will refer the joining peer to an **admitting peer**
- Contact an admitting peer
  - The admitting peer will help the joining peer learn about other peers in the overlay and establish connections to them as appropriate
P2PSIP OPERATIONS (2/2)

› **Storing data**: to perform a user registration (i.e. to insert the user’s contact information into the overlay), a user needs to:
  - Calculate a hash of her user name (e.g., `alice@example.com`) to produce a **Resource-ID**: `hash(alice@example.com) = 32B4A7F02C`
  - Locate the peer that is responsible for that Resource-ID
  - Store a `<Resource-ID, Node-ID>` mapping in the responsible peer

› **Fetching data**: to initiate a call:
  - Calculate a hash of the callee’s user name to produce a Resource-ID
    - `hash(alice@example.com) = 32B4A7F02C`
  - Locate the peer that is responsible for that Resource-ID in the P2PSIP overlay
    - A P2PSIP Resource Record with contact information is obtained: `alice@example.com → Alice’s Node-ID`
  - Establish a direct connection with the callee across NATs
  - Send a SIP INVITE request to the callee
EXAMPLE: ALICE CALLING BOB (1/3)

1. LOOKUP

(1) Calculate hash(bob@p2psip.net) = 4

Bob
sip:bob@p2psip.net
Node-ID: 15
Resource-ID: hash(bob@p2psip.net)=4

(2) Fetch Resource Record with Resource-ID 4

(3) Forward Fetch request

(4) Return Resource Record: Bob’s Node-ID = 15

(5) Forward Fetch response

(6) Alice learns that Bob’s Node-ID = 15

Alice
sip:alice@p2psip.net
Resource-ID: hash(alice@p2psip.net)=2
Node-ID: 11

Carol
sip:carol@p2psip.net
Node-ID: 4

Resource-ID: 4
Content: Bob’s Node-ID=15

Bob’s Resource Record

Bob’s Resource Record
EXAMPLE: ALICE CALLING BOB (2/3)

2. ATTACH

(1) Establish a connection with Node-ID 15 (Bob’s terminal)

(2) Send an Attach request to Bob

(3) Return an Attach response to Alice

(4) A direct connection for SIP between Alice and Bob

Alice
sip:alice@p2psip.net
Resource-ID: 2
Node-ID: 11

Bob
sip:bob@p2psip.net
Node-ID: 15
Resource-ID: 4

Carol
sip:carol@p2psip.net
Node-ID: 4

Bob’s Resource Record
Resource-ID: 4
Content:
Bob’s Node-ID=15
EXAMPLE: ALICE CALLING BOB (3/3)

3. INVITE

Alice
sip:alice@p2psip.net
Resource-ID: 2
Node-ID: 11

Bob
sip:bob@p2psip.net
Node-ID: 15
Resource-ID: 4

(1) SIP INVITE

(2) SIP 200 OK

(3) SIP ACK
SOME CHALLENGES FOR P2PSIP

› Security and identity assertion
  – No fully distributed system for security exist which would be as robust as a centralized solution
  – Solution: RELOAD uses a centralized entity contacted at enrollment time

› Network Address Translators (NATs)
  – Most peers can be located behind NATs
  – Solution: use of standardized NAT traversal protocols
    › Session Traversal Utilities for NAT (STUN)
    › Traversal Using Relays around NAT (TURN)
    › Interactive Connectivity Establishment (ICE)

› Regulatory issues
  – Lawful intercept, emergency calls
RESOURCE LOCATION AND DISCOVERY (RELOAD)

› A **P2P signaling protocol** specified by the P2PSIP WG
› Used to provide a self-organizing overlay network service, including
  - Distributed storage
  - Message forwarding
› Allows access from **client nodes** which don’t route traffic or store data
› Provides the following features:
  - Security framework
  - Usage model
  - NAT traversal
  - Routing
  - Pluggable overlay algorithms

![Diagram of P2PSIP Overlay](image-url)
Each application defines a RELOAD usage

End-to-end reliability, request state management, dispatches messages and operations

Provides packet forwarding services. Handles setting up connections across NATs using ICE.

Usages use RELOAD through Messaging API

Processes messages related to storage and retrieval of data.

Implements an overlay algorithm.
RELOAD FEATURES (1/2)

▶ Security framework
  – Node-IDs and certificates are assigned by a central enrollment server
  – Also self-signed certificates can be used
  – Security at three levels: connections, messages, stored objects

▶ Usage model
  – Allows the definition of new application usages
  – RELOAD can be used also by other applications than P2PSIP

▶ NAT traversal
  – Allows RELOAD to function in environments with NATs and firewalls
  – Interactive Connectivity Establishment (ICE) is used to establish new RELOAD and application protocol connections
RELOAD FEATURES (2/2)

› **Routing**
  - A lightweight forwarding header to minimize the load of intermediate peers
    › Via list and destination list
  - Basic routing mechanism is symmetric recursive

› **Pluggable overlay algorithms**
  - RELOAD has an abstract interface to the overlay layer
  - Each overlay can select an appropriate overlay algorithm
    › All algorithms rely on the common RELOAD core protocol
  - RELOAD defines three methods for overlay maintenance: Join, Leave and Update
  - Chord DHT is mandatory to implement

![Routing Diagram]

1. Request
   Dest: D
2. Request
   Dest: D
   Via: A
3. Request
   Dest: D
   Via: A, B
4. Response
   Dest: C, B, A
5. Response
   Dest: B, A
6. Response
   Dest: A
INTERACTIVE CONNECTIVITY ESTABLISHMENT (ICE)
Network Address Translation (NAT)

- Mapping of IP addresses from one realm to another
- E.g., connect an isolated address realm with private addresses to an external realm with globally unique addresses
- Thanks to NAT, a host in a private network can transparently communicate with destinations on an external network
  - And vice versa

Types of address and port mapping
- Endpoint independent mapping
- Address dependent mapping
- Address and port dependent mapping

Types of filtering
- Endpoint-independent filtering
- Address-dependent filtering
- Address and port dependent filtering
SIP, RELOAD, and WebRTC use Interactive Connectivity Establishment (ICE) to set up connections across NATs. ICE makes use of STUN and TURN protocols.

**STUN** – Session Traversal Utilities for NAT
- Determine IP address and port allocated by NAT (server reflexive address)
- Check connectivity
- Keep-alives

**TURN** - Traversal Using Relays Around NAT
- Obtain a relayed address
- Control the operation of a relay

ICE is used to discover a working path through NATs:
- **(1)** Gather candidate addresses
- **(2)** Exchange candidates
- **(3)** Perform connectivity checks
NAT TRAVERSAL FOR MEDIA IN SIP

1) TURN Allocate
2) Allocate response
3) SIP INVITE <ICE candidates>
4) SIP 100 Trying
5) SIP INVITE <ICE candidates>
6) TURN Allocate
7) Allocate response
8) SIP 200 OK <ICE candidates>
9) SIP 200 OK <ICE candidates>
10) SIP ACK
11) SIP ACK
12) ICE connectivity checks for media
13) RTP media
WEB REAL-TIME COMMUNICATION (WEBRTC)
Voice and video telephony in **HTML5**
- HTML5: the 5th revision of the HTML standard
- Interoperable, no plugins required

Some aspects of real-time communication in HTML5
- Getting multimedia streams from local devices
- Recording streams locally
- Connecting to remote peers using NAT traversal
- Sending streams to remote peers and receiving streams
- Displaying the streams using HTML5 `<video>` or `<audio>` elements
- Sending arbitrary data to remote peers

**RTCWeb** WG in the **IETF**
- Scope: the protocols that browsers talk to each other
- For WG charter, see [1]

**WebRTC** in **W3C** (World Wide Web Consortium)
- Scope: APIs that are offered to Javascript applications to take advantage of the browser’s functionality
- For current API draft, see [2]
RTCWEB

› IETF RTCWeb WG focuses on the protocols
› Functionality groups
  – **Data transport** – sending and receiving data, NAT traversal
  – **Data framing** – RTP and SRTP (Secure Real-Time Protocol)
  – **Data formats** – codecs, format specifications
  – **Connection management** – setting up, negotiating, and tearing down connections
  – **Presentation and control** – W3C API effort, user control over browser’s interaction with input/output devices
  – **Local system support functions** – e.g., echo cancellation, volume control
WEBRTC TRAPEZOID

WEB SERVER

WEB SERVER

SIGNALLING PATH
(SIP)

WEB

MAEDIA PATH

JS/HTML/CSS

WEB

WEB

JS/HTML/CSS

ALICE

BROWSER

BORWER

PEERCONNECTION API

(SRTP)

PROPRIETARY OVER
HTTP/WEBSOCKETS

PROPRIETARY OVER
HTTP/WEBSOCKETS

Page 33
CALL ESTABLISHMENT IN WEBRTC

1. **Download** a video communication web application (JavaScript)
2. **Obtain access to media and create a PeerConnection**
   - PeerConnection allows two users to communicate directly, browser-to-browser
   - `new PeerConnection(configuration, signalingCallback)`
     - `configuration`: address of a STUN/TURN server
3. **Use a signaling protocol** over bidirectional HTTP or WebSocket to talk to server
   - Bidirectional HTTP: e.g., long polling, HTTP streaming
   - WebSocket: bi-directional, full-duplex communication channel over a single TCP socket
     - Implemented in web browsers and web servers
   - Signaling protocol not specified by WebRTC
   - Support for SDP and offer/answer model is mandatory
     - ICE candidates in SDP
4. **Servers** may talk SIP to each other
5. **Media path** directly between browsers
   - Enabled by PeerConnection
   - ICE negotiation
   - SRTP (Secure Real-Time Protocol) for media transport
JAVASCRIPT SESSION ESTABLISHMENT PROTOCOL (JSEP)

› A signaling API for establishing P2P connections
› Signaling state machine is implemented in JavaScript
  – In contrast to ROAP (RTCWeb Offer/Answer Protocol)
  – JS app can fully control the signaling plane of a multimedia session
› Format of session descriptions is SDP
› Examples of JSEP API calls
  – `New PeerConnection()`
  – `addStream()`
  – `createOffer()`
  – `setLocalDescription()`
  – `setRemoteDescription()`
  – `createAnswer()`
  – `startIce()`
WEBRTC IMPLEMENTATIONS

- Google Chrome (stable version)
- Mozilla Firefox (stable version)
- Chrome for Android (since version 29)
- Firefox for Android (since version 24, out Sept 17th 2013)
- Opera
- Opera Mobile
REFERENCES

› [1] RTCWeb charter
  - http://tools.ietf.org/wg/rtcweb/charters

› [2] WebRTC API
  - http://dev.w3.org/2011/webrtc/editor/webrtc.html