A Survey of Security Issues and Solutions in Presence[*]

Vishal K. Singh and Henning Schulzrinne
Department of Computer Science, Columbia University
{*vs2140, hgs*}@cs.columbia.edu

Date: 10th Feb 2006

Abstract: With the growth of presence based services, it is important to securely manage and distribute sensitive presence information such as user location. We survey techniques that are used for security and privacy of presence information. In particular, we describe the SIMPLE based presence specific authentication, integrity and confidentiality. We also discuss the IETF's common policy for geo-privacy, presence authorization for presence information privacy and distribution of different levels of presence information to different watchers. Additionally, we describe an open problem of getting the aggregated presence from the trusted server without the server knowing the presence information, and propose a solution. Finally, we discuss denial of service attacks on the presence system and ways to mitigate them.

1 Introduction

Instant messaging and presence forms an integral part of Internet communications, both in corporate and consumer world [26]. A user's presence information [13] includes his availability and willingness for communication. The main components of a presence service include presentity, watcher and presence server. A *presentity* is an entity such as a user, resource, service or application whose presence information is processed [13] and distributed. A *watcher* is an entity which requests presence information of a presentity. The presentity can generate presence information from various sources such as cell phone or PC client. The presence server (presence agent) combines the information to give a consistent view of the presentity status to the watchers.

The privacy and security of presence information is important, especially if it contains sensitive information such as user location. The presentity specifies a set of rules known as presence authorization rules [15] to the server, so that the server can give different presence information to different watchers. A malicious user may snoop on the network to learn the sensitive presence information. To prevent this, the presence information needs to be stored and distributed securely. Secondly, only authorized watchers should receive presence information from the server. Lastly, the server must authenticate the presence sources of the presentity.

This paper surveys presence security and privacy issues. It describes mechanisms defined by the Session Initiation Protocol (SIP) [1] and SIMPLE [5] standards to deal with these issues. It does not discuss threat models and attacks except the DoS attack; however, we refer to specific attacks while describing different security mechanisms.

The remainder of this document is organized as follows: Section 2 presents the background of a SIP/SIMPLE based presence system. Section 3 presents the security requirements for presence. Section 3.1 explains the presence identity model, Section 3.2

[*] This work is supported by Verizon Labs.

discusses authentication as a way to determine digital identity of a presentity and ways to achieve authentication in a presence system in different scenarios. Section 3.3 discusses presence source authorization, followed by presence data security in section 3.4. Section 4 discusses presence privacy. Section 5 discusses trust in a presence system, followed by provider trust and anonymity from provider. Section 6 discusses denial of service attacks on the presence system, illustrating ways to mitigate it. Section 7 lists future work identified in this survey, followed by conclusion in Section 8 and references in Section 9.

2 Background

SIP [1], initially defined for managing multimedia sessions, and was extended by the SIMPLE working group of the IETF to compose and distribute presence information, which was later generalized to distribute generic events. The details about SIP request routing and the roles of different SIP header fields like method type, "From" and "To" headers, "Contact" header is explained in detail in [1].

Watchers subscribe to presence information of presentities using SIP SUBSCRIBE [5] message and are notified about the state changes of presentities by SIP NOTIFY [5] messages. Presence data for presentity is published from multiple presence sources using SIP PUBLISH [14]. Diverse sources of presence information like wireline and wireless phones, applications like calendars and meeting maker applications and location sensors update presence information on the server using SIP PUBLISH. Fig. 1 shows the basic block diagram of presence system.

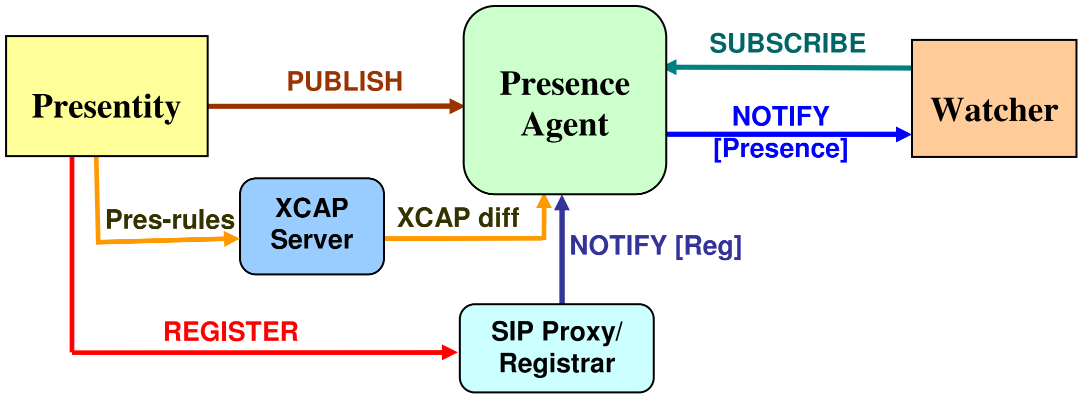

Figure 1 Basic block diagram of SIMPLE based presence system

3 Presence System Security

This presence information is critical and if compromised can be misused and potentially used for harmful purposes. The following are the security requirements for presence.

- All presence sources must be authenticated before they can update presence information for a presentity.
- Only authorized sources can update presence information for a presentity.
- Only authenticated watchers can subscribe to presence information.

- Only authorized watchers can obtain the presence information based on the privacy filter of the presentity.
- Watcher authorization should be based on privacy filter rules.
- Only a presentity can create and modify its own privacy filter rules.
- Presentities can specify privacy filters only after they are authenticated.
- Confidentiality and integrity of presence information and privacy filters must be ensured.

In the next section, we describe what an identity is in SIP based presence systems.

3.1. Presence Identity Model

In presence system, a unique identity is represented by a presentity. It is a principal which owns or manages a program, application or service. Identity of a user agent which updates its presence information and gets updated about others presence information must be determined. This is to identify who is the sender of a message and map the sending application or device to its principal. For example, when a message is received by a user *candy@columbia.edu* from *bob@cisco.com*, Candy knows that the identity bob@cisco.com maps to Bob who is her friend in Cisco.

In SIP, the identity is expressed using SIP address-of-record (AOR), which is a SIP or SIPS URI present in "From" header of the request. RFC 3674 defines "*An address-of-record represents an identity of the user, generally a long-term identity, and it does not have a dependency on any device; users can move between devices or even be associated with multiple devices at one time while retaining the same address-of-record. A simple URI, generally of the form 'sip:egdar@example.com', is used for an address-of-record.*". When a SIP request is created by a user agent, it populates the AOR of target user in its "To" header field and Request-URI. The AOR of the user that is sending the request populates the "From" header field of the message; the contact address of the device from which the request is sent is listed in the "Contact" header field. A SIP device (i.e., a user agent) is associating its own contact address with the user's address-of-record. In doing so, the device is assuming the identity of the principal of the AOR and becomes eligible to receive requests that are sent to the AOR.

A unique identity in a presence system is called presentity. The presentity is identified by an identifier which is a URI and generally of the form user@domain. It has a protocol specific prefix called scheme, e.g., pres:, imp:, sip: etc. The presence server owns the domain part and verifies the user identifier in its domain. The presence data model [17] discusses the presentity's identity in detail. The draft specifies that "*For each unique presentity in the network, there is one or more presentity URIs. Presentity may have multiple URI because they are identified by both a URI from the Presence (pres) scheme and a protocol specific URI, such as a SIP URI or an XMPP URI [20]. Or, it can be because a user has several aliases in a domain, all of which are equivalent identifiers for the presentity*". "*Presentity URI is independent of any of the services or devices that the presentity possesses*". Thus, a URI represents a resource which can be subscribed to obtain its presence information.

We consider the SIP address of record as the presentity's identity. When a presence document is constructed, the presentity URI is set to the identifier used to request the document i.e., request URI in the SUBSCRIBE request. An identity in a presence system is established by the process of authentication. In the following section, we describe different mechanism to achieve authentication. Also, we describe authentication mechanisms in inter-domain and intra-domain scenarios.

3.2. Authentication

Authentication is the primary step in presence security and is used to verify the identity of a presentity. There can be varying degrees of authentication. Authentication authorizes a source to use a given identity, in this case to send PUBLISH and SUBSCRIBE requests, receive NOTIFY requests and update privacy filter rules using XCAP [21].

In the next section, we describe various mechanisms for authentication.

3.2.1 Ways to Perform Authentication

Presentities and watchers are authenticated by an authentication proxy server or a presence server using one of the following ways.

Digest Authentication

Presence uses SIP's digest authentication. This mechanism is explained in RFC 2617 [4]. Details about using digest authentication for SIP messages is explained in Section 22.2-22.4 of RFC 3261 [1]. The digest scheme is based on a simple challenge-response paradigm. It uses a nonce value to give the challenge. A valid response contains a checksum (by default, the MD5 checksum) of the username, the password, the given nonce value, the method type, and the requested URI. In a SIP network, the authentication takes place between the user agent client and the proxy or between a User Agent Client (UAC) and User Agent Server (UAS), where the server requires the user agent client to authenticate itself.

For a UAC to UAS authentication, the **authorization header** is used. It consists of credentials containing the authentication information of the UAC. Specifically, It contains a hash generated using the nonce, the realm, the request method (the type of request message), the request-method version, and the authorization type. For, UAC to proxy authentication, **proxy-authorization header** is used. This header contains the type of authentication, credentials of the user agent, or the realm of the resource being requested.

Example of Authorization header field;

```
Authorization: Digest username="bob",
realm="biloxi.com",
nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
uri="sip:bob@biloxi.com",
qop=auth,
nc=00000001,
cnonce="0a4f113b",
response="6629fae49393a05397450978507c4ef1",
opaque="5ccc069c403ebaf9f0171e9517f40e41"
```

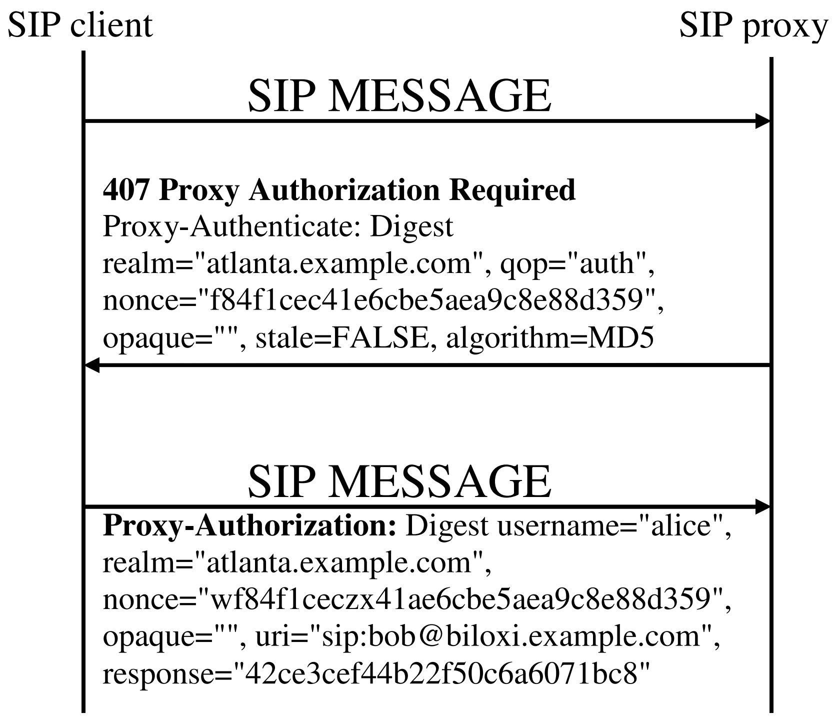
SIP client
SIP proxy
SIP MESSAGE
407 Proxy Authorization Required
Proxy-Authenticate: Digest realm="atlanta.example.com", qop="auth", nonce="f84f1cec41e6cbe5aea9c8e88d359", opaque="", stale=FALSE, algorithm=MD5
SIP MESSAGE
Proxy-Authorization: Digest username="alice", realm="atlanta.example.com", nonce="wf84f1ceczx41ae6cbe5aea9c8e88d359", opaque="", uri="sip:bob@biloxi.example.com", response="42ce3cef44b22f50c6a6071bc8"

Figure 2 SIP authentication example

Asserted Identity

SIP servers assert the identity of authenticated users by inserting P-Asserted-ID header. The user in this case can be authenticated using the digest mechanism explained above. If the received request contains a P-Asserted-ID header field [2] and is received from a trusted proxy server, the request is considered authenticated. This mechanism is detailed in RFC 3325 [2] and is used within a trusted network.

An example of asserted identity is given below.

```
INVITE sip:vs2140@columbia.edu SIP/2.0
Via: SIP/2.0/TCP sipd.columbia.edu;branch=z8hy6
Via: SIP/2.0/TCP sipd.cs.columbia.edu:branch=u78Ty
To: <sip:vs2140@columbia.edu>
From: "Anonymous" <sip:anonymous@anonymous.invalid>;tag=9802748
Call-ID: 24345456567787
CSeq: 2 INVITE
Max-Forwards: 69
P-Asserted-Identity: "Abc" <sip:abc@xyz.com>
Privacy: id
```

Cryptographically Verified Identity

For cryptographically verified identity, the request is authenticated if it contains an **Identity** header field as defined in SIP Identity draft [8]. The Identity header field validates the "From" header field of the request. The signature in the identity header is used to verify that the request legitimately came from this identity to be considered authenticated.

The signature is created using the AOR of the UA sending the message, the AOR to which the request is being sent, the Call-ID header filed value, the CSeq header's digit (1*DIGIT) and method portions, the Date header field value , the Contact header field value and the content body of the message. The signature creation scheme is detailed in Section 9 of SIP Identity draft [8].

An example of cryptographically verified identity message is given below.

```
INVITE sip:bob@biloxi.exmple.org SIP/2.0
Via: SIP/2.0/TLS pc33.atlanta.example.com;branch=z9hG4bKnashds8
To: Bob <sip:bob@biloxi.example.org>
From: Alice <sip:alice@atlanta.example.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Max-Forwards: 70
Date: Thu, 21 Feb 2002 13:02:03 GMT
Contact: <sip:alice@pc33.atlanta.example.com>
Identity:"kjOP4YVZXmF0X3/4RUfAG6ffwbVQepNGRBz58b3dJq3prEV4h5Gn
           4F6udDRCI4/rSK9cl+TFv45nu0Qu2d/0WPPOvvc3JWwuUmHrCwG
          C+tW7fOWnC07QKgQn40uwg57WaXixQev5N0JfoLXnO3UDoum89
           hXPAIp2vffJbD4="
Identity-Info: <https://atlanta.example.com/atlanta.cer>;alg=rsa-sha1
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 pc33.atlanta.example.com
s=Session SDP
c=IN IP4 pc33.atlanta.example.com
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

The original authenticator, which can be client's local proxy server, forms the identity signature and adds an "Identity" header to the request containing this signature. It also adds an "Identity-Info" header. The Identity-Info header contains a URI by which its certificate can be acquired. Thus, the local domain proxy or the UA itself communicates to other entities that the request is authenticated. This mechanism can be used for end to end security and inter domain scenarios or if the users don't have their own certificates.

Certificate-based Authentication (S/MIME)

The use of S/MIME is only possible if the end user agent has its own certificate. These certificates assert that the holder is identified by an AOR (user@domain portion of SIP or SIPS). The UserAgent's (UA's) certificates are also used to sign and encrypt SIP request message bodies. The bodies are signed using private keys of the sender and encrypted using the public key of receiver. Using S/MIME, sender can sign and encrypt the header of SIP message within MIME bodies of type "message/sip". The main problem with this approach is creation and distribution of end user keys. This problem is addressed in the draft "Certificate Management Service for SIP" [25]. The certificate management service stores the certificates and allows SIP UA's to discover certificates of other users using SIP SUBSCRIBE with event "certificate".

Next, we describe ways to achieve authentication in intra-domain and inter-domain (multiple federations) scenarios.

3.2.2 Authentication Scenarios

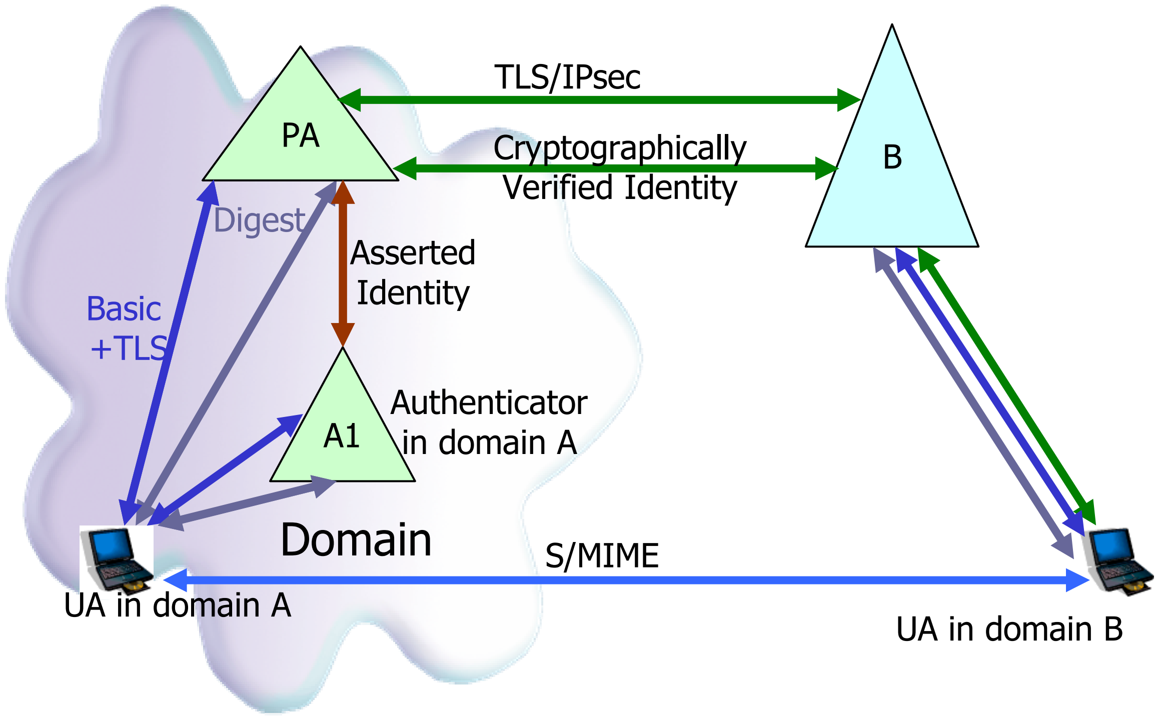

Figure 3 SIP authentication scenarios

Figure 3 shows different SIP based authentication techniques when used in inter-domain and intra-domain scenario. A1 can be a proxy to which UA in domain A authenticate itself, requests from UA's in domain A can be asserted by A1 before being sent to PA in domain A. Alternatively, the PA can authenticate the UA directly using the techniques specified above like digest, basic auth + TLS. PA in domain A can authenticate proxy or PA in domain B using TLS or Cryptographically Verified Identity.

Intra-Domain Authentication

In a single domain, the entities that want to communicate with the presence server are watchers, publishers, proxy servers and the XCAP server. XCAP clients need to communicate with the XCAP server. The watchers, publishers and the XCAP client have shared secrets which they can use to authenticate themselves. The servers use certificates to authenticate themselves to the clients.

Inter-Domain Authentication

In inter-domain or multiple federations, authentication is based on transitive trust. The UA (watcher or publisher) needs to authenticate itself to local presence server or authentication proxy server using the authentication mechanism. This can be done over

TLS. The foreign domain (remote federation) presence server receives the SUBSCRIBE request from the local domain presence server or a proxy server. These messages identify the watcher in the local domain and are treated based on trust relationships between the domains or providers. The request must be sent to foreign domain presence server over a TLS connection, so that the foreign domain can authenticate the local domain's proxy or presence server.

3.3. Authorization

Once authenticated, the entity is authorized to send and receive messages. However, presentity must authorize the watchers so that the server can distribute the presence information to them. This will be explained in more detail in Section 4. However, sources which update the presence information for a presentity must also be authorized. Unauthorized sources can publish incorrect presence information and make the presence state inconsistent. Currently, a source is authorized to update presence information if it has presentity's credentials, in other words, it is based on authentication. The source must authenticate itself to use the identity of presentity, i.e., SIP AOR of presentity in the FROM header of SIP PUBLISH requests. In future, third party sources may be able to update presence information for presentity. In that case, the presentity will authorize each presence source using **presence source authorization rules**. If there are use cases which need third party PUBLISH, IETF may need to provide a specification to do this. Additionally for creating presence authorization rules using XCAP, it is required that the application must be authorized to create, modify or delete the rule. This authorization is based on authentication of XCAP client. There may be a requirement to give other entities authorization to create or change the authorization policy, e.g., manager doing for his employees, a parent for their children.

3.4. Presence Data Security

Presence data security implies ensuring confidentiality and integrity of presence data. We divide presence data security into two sections, namely, transport security and storage security.

3.4.1 Transport and Network Layer Security

Transport security must provide protection of presence data and messages from man-in-middle attacks, replay attacks and to ensure confidentiality and integrity of presence data, presence requests and privacy filter documents (created using XCAP over HTTP). It requires the usage of Transport Layer Security (TLS) [11] or Secure Multipurpose Internet Mail Extension (S/MIME) [10] or IP Security (IPSEC) [19].

Confidentiality

This implies confidentiality of presence information contained in SUBSCRIBE, PUBLISH and NOTIFY requests and bodies, watchers and privacy filter rules, all of which are sensitive and must be stored and transported confidentially. Confidentiality of SIP messages and message bodies is discussed in 23.4.1.2 in the SIP specification [1]. XCAP clients should use HTTP over TLS for updating policy documents.

Integrity

The publishing of presence information, subscribe and notification messages as well as privacy filter data must be protected against content modification on the network. The main goal is to protect against altered and injected messages. Integrity protection of SIP message headers and bodies is discussed in [1] in Section 23.4.1.1 and 23.4.2.

The above goals can be achieved by using TLS, S/MIME or IPsec. SUBSCRIBE, PUBLISH or NOTIFY request cannot be encrypted as proxy servers may need to look at certain headers, but body of each of them should be encrypted. This can be achieved using S/MIME. The privacy filter rule must be updated over secure channel. This can be done using a TLS connection. TLS or S/MIME requires an administrative domain to have a certificate which includes a certification authority (CA) for issuing certificates to servers. In an intra domain scenario, a certificate issued by the local domain CA is accepted by all internal clients and servers. In inter-domain scenario, the CA must be trusted by both the parties or the certificates need to be signed by a trusted third party, a certificate signing authority.

3.4.2 Storage Security (Securing Stored Presence Data)

Data encryption prevents unauthorized access to presence data when stored at the server. This assumes the medium where the data is stored can potentially be compromised either physically or by a worm, a virus attack. Encrypting the data using a secret key can secure it even on insecure physical media. The encryption scheme, key deduction and other cryptographic details of encrypting the presence data in storage is beyond the scope of this survey.

4 Presence Privacy

Presence information like location and reachability is sensitive. This information must be distributed only to authorized watchers. The presentity must authorize the watchers using the authorization policy document. An authorization policy document contains the authorization rules and permissions specifying what parts of presence information can be sent to the watcher. The common policy draft [22] describes a framework for representing authorization policy, e.g., for geo-location and presence information. The authorization policy is created and modified by the presentity using XCAP or other offline mechanisms. In the next sections, we explain about privacy policy, presence authorization policy, location privacy and privacy filtering.

Privacy Policy Requirements

- Authorization of watchers before giving them presence information.
- Selective notification: Presentity must be able to specify what parts of presence information are given to watchers. This can be based on time of day, etc.
- Differential presence information: Presentity must be able to distribute different presence information to different watchers.
- Provide capability for presentity to specify authorization policy for anonymous subscriptions. Watcher can hide his identity using anonymous. Mechanisms to achieve privacy in SIP are specified in RFC 3323 [6].

- Local or national rules, e.g., E911 may override presentity's authorization rules.
- Location privacy, requirements for this are detailed in RFC 3693 [3].

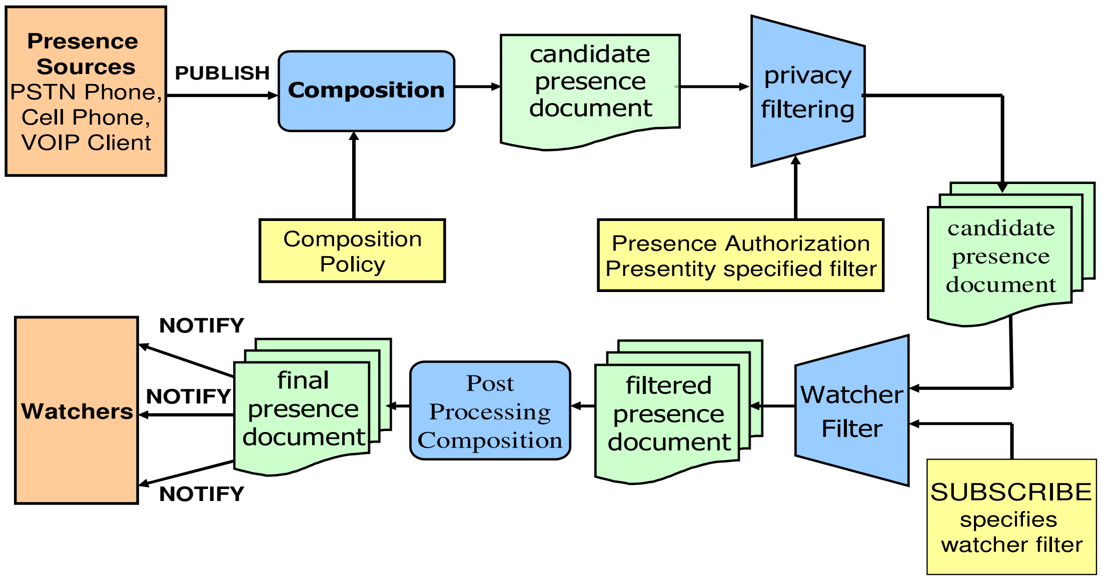
Presence Sources PSTN Phone, Cell Phone, VOIP Client
PUBLISH
Composition
candidate presence document
privacy filtering
Composition Policy
Presence Authorization Presentity specified filter
candidate presence document
NOTIFY
Watchers
NOTIFY
NOTIFY
final presence document
Post Processing Composition
filtered presence document
Watcher Filter
SUBSCRIBE specifies watcher filter

Figure 4 Presence processing data flow

Presence Authorization Policy

An authorization policy document contains a set of rules. Each rule contains conditions, actions, and transformations. Actions and transformations are together called as permissions. The conditions specify request attributes e.g., watcher identity and presentity state that should match, for the rule is to be applied. The actions element tells the server what actions to take. Actions include rules like block, polite-block or allow for the watcher identities. If the action allows the request, the transformations determine how the presence data is modified before being presented to that watcher, and as such, defines a privacy filtering operation.

The authorization policy itself is sensitive and can be used to tell if watcher is liked or disliked by the presentity. Thus, the policy document itself must be protected. The authorization policy document creation, modification and storage must be secure using authentication, encryption and integrity checks. We discussed in Section 2.4.1 ways to protect confidentiality and integrity of policy documents.

Privacy Filters

Privacy filtering step is invoked in the presence data processing after the composition step as shown in Figure 4. A candidate presence document is generated as a result of composition operation. Privacy filters are applied to provide selective access of presence data. The presence authorization rules [15] and geo-privacy [18] specify these filters for selective access to the presence and location data. Authorization rules can be based on time of day, location, identity. They specify permissions, i.e., actions and transformations for different parts of presence data. Some examples of permissions include *<provide-devices>, <provide-services>, <provide-persons>, <mood>, <place>, <place-type>, <relationship>, <status-icon>, <activities>*. The effect of these permissions is availability of mood, place, place-type, etc., in the final presence document which is sent to the watchers. "Sub-handling" can be used to deny or allow subscription request based on matched conditions. "Block" tells the server to place the subscription in the rejected state. "Confirm" tells the server to place the subscription in the "pending" state, and wait for input from the presentity to determine how to proceed. "Polite-block" tells the server to place the subscription into the "accepted" state, and to produce a presence document that indicates that the presentity is unavailable. "Allow" tells the server to place the subscription into the "accepted" state and send presence notification based on other permissions.

A sample Presence authorization rule from [15] is below.

```
<provide-devices>
  <device-id>urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6</device-id>
  <class>biz</class>
</provide-devices>.
```

The watcher receives presence information containing device-id or device family specified in the rule.

Location Privacy

Location information needs to be protected against unauthorized access. The requirements for location privacy are detailed in RFC 3693 [3].

Distribution of location information must also use authentication, confidentiality and integrity checks. The mechanisms to do so can be in transport protocol or in presence location object itself. The issues concerning location information are mainly related to (i) location information collection (ii) location information retention and (iii) location information distribution and use. These issues are addressed by geo-privacy [18] to an extent. The geo- privacy [18] rules specify the conditions and permissions that allow a presence server (Location Server (LS)) to distribute location information to watchers with different amount of location precision. Example of a location filter is "give my location to my family to a resolution of one km and to my friends to a resolution of five km". It also specifies rules specifying if this location information can be retained and for how long as well as if it can be further distributed.

Like presence policy, geo-privacy's conditions and transformations restrict access to location and are specified in [18]. One example of geo-privacy condition match is *<civic-loc-condition>* element matches. This happens only when the current location of the target matches all the values specified in the child elements of this element as shown in example below. The civic location transformation can be specified by means of the *<civic-loc-transformation>* element to restrict the level of civic location information which the LS is allowed to provide. From lowest to highest level, the names of these levels are: 'null', 'country', 'region', 'city', 'building', 'full', depending on which the presence location information distributed can be complete (full) or none (null). Each level has a set of civic location data items such as <country> and <A1>... <A6>, as defined in

[23]. Similarly, geospatial location transformation can be specified by means of the <*geospatial-loc-transformation*> element to restrict the resolution of the geospatial location information to the value provided in *the* <*latitude-resolution*>*,* <*longitude-resolution*> *and* <*altitude-resolution*>*,* child elements of the <*geospatial-loc-transformation*> element.

Sample Geo-Privacy Rule from [18]

```
<cp:rule id="AA56i09">
        <cp:conditions>
                <cp:validity>
                    <cp:from>2004-11-01T00:00:00+01:00</cp:from>
                    <cp:until>2005-11-01T00:00:00+01:00</cp:until>
                </cp:validity>
                <gp:civic-loc-condition>
                    <gp:country>DE</gp:country>
                    <gp:A1>Bavaria</gp:A1>
                    <gp:A3>Munich</gp:A3>
                    <gp:A4>Perlach</gp:A4>
                    <gp:A6>Otto-Hahn-Ring</gp:A6>
                    <gp:HNO>6</gp:HNO>
                 </gp:civic-loc-condition>
        </cp:conditions>
        <cp:actions/>
       <cp:transformations>
                <gp:distribution-transformation>true
                         </gp:distribution-transformation>
                <gp:keep-rules-transformation>true
                         </gp:keep-rules-transformation>
                <gp:civic-loc-transformation>full
                          </gp:civic-loc-transformation>
                 <gp:geospatial-loc-transformation>
                    <gp:lat-resolution>0.00001
                         </gp:lat-resolution>
                    <gp:lon-resolution>0.00001
                         </gp:lon-resolution>
                 </gp:geospatial-loc-transformation>
        </cp:transformations>
</cp:rule>
```

Notifier Privacy Mechanism

The presence server must respond to subscription requests such that it does not reveal any presence information, nor reveals level of filtering used for that watcher. The SUBSCRIBE response message codes, e.g., 200, 202, 400 or 600; reveal different information to the watcher about presentities authorization rules. Information about whether a user is authorized to subscribe to the requested state should never conveyed back to the original user.

5 Trust and Anonymity

Trust extends from Identity (Authentication)

A user is trusted by the presence server if he possesses the shared secret credentials and can authenticate him using them. Trust association is on a per message basis. Trust in inter-domain scenario is achieved using inter-domain authentication mechanism as described in section 2.2.1. Even though the client and server authenticate each other, it is the client who trusts the server and gives it presence data. The real trust is client's trust on the server's identity and not the server proving its identity.

Provider Trust

The presentity's principal has no way to ensure if the provider fully protects its presence information or this information is compromised by the provider either knowingly or unknowingly. Importantly, presentity might not want the service provider to know its presence information.

One potential solution which can be explored to deal with this issue is to use Peer-to-Peer based presence distribution where each p2p enabled UA acts as presence agent. In this case, there is no single server which has the entire presence document, but a random peer may have the presence document or a part of it. The watcher and notifier UA perform distributed processing of presence information. P2P based management and distribution of presence information is an area for further research.

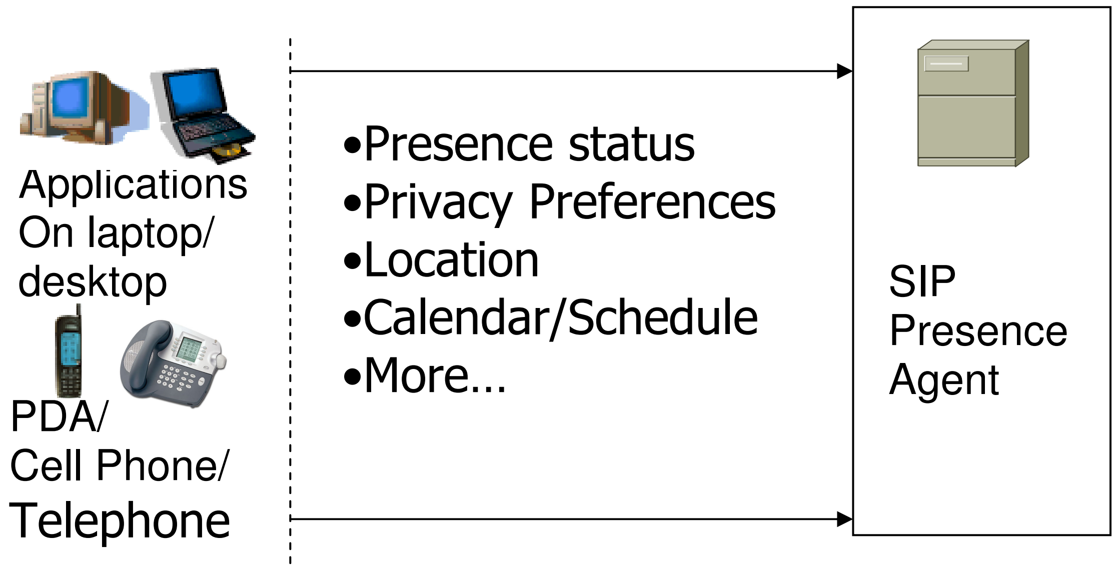

Figure 5 Presence service provider trust

Anonymity from Provider

"My presence information must not be compromised. Is my availability information really private? Can the Provider target advertisement on me? How much does my provider know about me?" These issues require that presence information remain anonymous from the presence server. This creates a requirement that the presence agent should be able to do data aggregation, filtering and distribution without being able to know the actual information. This problem and our proposed solution are explained below.

Problem Statement

A Presence server must be able to perform basic composition and filtering without actually getting the presence information. To do this, the server should be able to process the PIDF just by looking at XML tags without being able to see the data within the XML tags. The value inside the tags needs to be encrypted and clients who possess the right keys will be able to know the information within the XML tags. In other words, we want to get the server based services like aggregation, filtering and distribution without actually revealing any presence information to the provider. The server is assumed to be trusted to distribute the presence data. The objective is to achieve anonymity or hiding the presentity's presence information from the server.

There can be various scenarios in which this can be useful. Before describing the solution we list our assumptions for the solution.

Assumptions

All watchers are authenticated not by the presence server itself but by third party authenticator (authentication service). This ensures presence server does not indulge in anomalous behavior by enacting as another watcher who is authenticated by server itself, to get the presence information and its unauthorized distribution. Watchers can also be authenticated by presentity itself using certificates signed by a trusted CA. A trusted server will not pose as authorized watcher and try to get presence information. To prevent this we propose use of authentication service.

Participating entities have public key of other entities like watchers, presentity and presence server which is originally distributed using some public key distribution mechanism.

Proposed Solution

The idea is to have a shared key between the presentity and the watcher which is not with the server. Parts of XML (PIDF based presence data) will be encrypted using the shared key. This can only be decrypted by the watchers as only they have the shared key.

There are two variants of the solutions. In the first variant, both the watcher and presentity have a shared session key. The shared session key is given to authorized watchers to decrypt the encrypted data within the XML tags. In the second variant, we give each presentity another set of (public, private) key pair, public key of which is given to each authorized watcher. Thus, presentities have two sets of keys. First set is the public-private key pair used by the presentity and is installed using some PKI distribution mechanism. Let's call this as (Pu1, Pr1). The second set is self generated (public, private) key pair to be used only for presence purposes, specifically to solve this problem. Let's call this as (Pu2, Pr2). The second key pair is used for encrypting and decrypting presence information and can be used to generate the session key. Since, a shared session key is cheaper and can be used to achieve the same purpose, we need not use a public-private key pair based variant of solution, instead use a session key based solution.

When a SUBSCRIBE message is received from an authenticated watcher, the presentity sends its shared session key (or the presence public key (Pu2)) encrypted using public key of the watcher. Only watchers can get this key and not the presence server or any other entity as it encrypted using watcher's public key. The data within the XML tags is encrypted using this shared session key (or presence private key (Pr2) of presentity). Higher layer XML tag names and attributes remain unencrypted and are assumed to be sufficient for composition and basic filtering operations by the presence server. Once the presence server performs data processing operations it encrypts the presence data for each watcher using the public key of the watcher and sends notification messages. Watchers decrypt the presence data, first using their private keys and then using the shared session key (or the presence public key (Pu2)) given by the presentity initially.

The presentity creates an internal subscription or the watcher SUBSCRIBE's for changes in key as shown in figure 2 below, so that presentity sends a NOTIFY message with the shared key whenever the shared key changes.

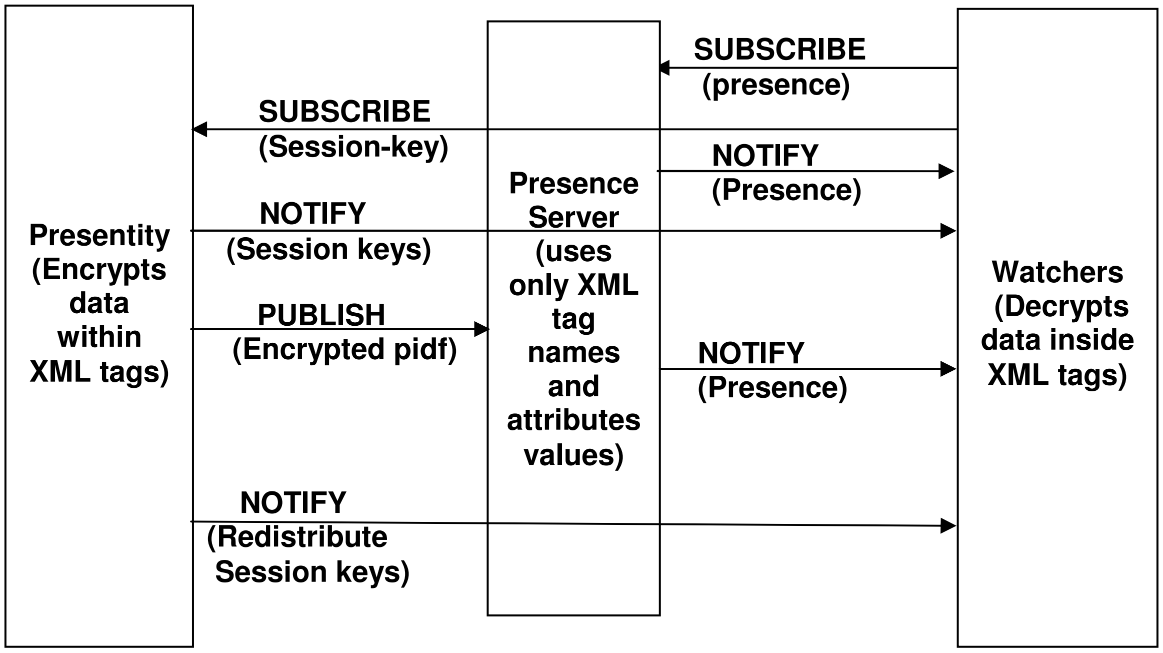
Presentity (Encrypts data within XML tags)
Presence Server (uses only XML tag names and attributes values)
Watchers (Decrypts data inside XML tags)
SUBSCRIBE (presence)
SUBSCRIBE (Session-key)
NOTIFY (Presence)
NOTIFY (Session keys)
PUBLISH (Encrypted pidf)
NOTIFY (Presence)
NOTIFY (Redistribute Session keys)

Figure 6 SIMPLE messages and Key distribution

Fig. 2 above shows the distribution of keys and message exchanges. The figure shows that the keys are directly sent to watchers. Even if NOTIFY is not directly delivered, the keys are sent encrypted using the public key of watcher and can be decrypted only by the watcher who has the correct private key.

6 Denial of Service

We explained the mechanism to achieve confidentiality and integrity, anonymity and trust in presence systems. Denial of service attacks (DoS) affect the availability of the presence service. DoS attack can be centralized or distributed (DDoS) launched either on the network or the presence server or any other user agent.

A presence service generates multiple NOTIFY messages for each PUBLISH message. Each message causes twice the number of filtering operations (one for privacy filter and second for watcher filtering). Therefore, PUBLISH messages arriving at a very high rate can cause a large number of NOTIFY generation which can cause network congestion and server overloading. To prevent bogus PUBLISH messages causing network or server overload, PUBLISH requests should be authenticated before being processed.

Processing of a SUBSCRIBE message involves authentication, authorization and filtering operations to generate the NOTIFY message. A large number of SUBSCRIBE requests with bogus authentication details can also lead to DoS attack on the server.

Attackers can modify the "From" SIP header to have the address of another user agent; so that the response will be sent to that user agent. This could cause DoS attack on that user agent by generating unsolicited notifications. This needs to be prevented by authentication and return routability verification.

Man-in-the-middle attacks using SUBSCRIBE could create arbitrary subscriptions, hijack existing subscriptions, terminate outstanding subscriptions, or modify the resource to which a subscription is being made. To prevent such attacks integrity protection across "Contact", "Route", "Expires", "Event", and "To" headers of SUBSCRIBE messages must be provided.

A change in authorization policy requires re-evaluating authorization status of existing subscriptions. Hence, another form of DoS attack can be launched if an adversary can modify authorization policy document. Hence, the presence user agent must be protected from getting compromised and authorization of XCAP requests to change policy document must be done.

7 Future Research Work

Presence source authorization rules need to be defined. It depends on use cases or requirements for presence source authorization. Similarly, requirement for XCAP source authorization can be evaluated.

A peer to peer based presence data processing and distribution system can be developed. This p2p system may also help to achieve presence data anonymity from provider. The solution can be as simple as p2p storage of presence data to as complex as distributed presence data processing, e.g., distributed presence composition.

Further work is possible on detection and prevention of denial of service attacks, detailed analysis of privacy implications of presence data, analysis of threat models and attacks on presence system.

8 Conclusion

In this survey paper, we described a complete architecture to ensure security and privacy of presence information. We explain mechanisms to ensure authenticity, confidentiality and integrity of presence requests (PUBLISH, SUBSCRIBE and NOTIFY) and privacy filter update requests (XCAP). We also described technologies and protocols that can be used to perform authentication of different entities in inter-domain and intra-domain scenarios. Secure transmission of presence data can be achieved using IPsec and TLS protocols. Privacy mechanisms for access control and filtering using the common policy, presence authorization rules and geo-privacy is described. We discuss provider trust and anonymity and propose a solution for it which can be further worked on. Finally, we discuss denial of service attacks that might affect the availability of presence service and ways to minimize the impact of such an attack.

9 References

1. Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.
2. Jennings, C., Peterson, J., and M. Watson, "Private Extensions to the Session Initiation Protocol (SIP) for Asserted Identity within Trusted Networks", RFC 3325, November 2002.
3. Cuellar, J., Morris, J., Mulligan, D., Peterson, J. and J. Polk, "Geopriv Requirements", RFC 3693, January 2004.
4. Franks, J., Hallam-Baker, P., Hostetler, J., Lawrence, S., Leach, P., Luotonen, A., and L. Stewart, "HTTP Authentication: Basic and Digest Access Authentication", RFC 2617, June 1999.
5. Roach, A., "Session Initiation Protocol (SIP)-Specific Event Notification", RFC 3265, June 2002.
6. Peterson, J., "A Privacy Mechanism for the Session Initiation Protocol (SIP)", RFC 3323, November 2002.
7. Rosenberg, J., "A Presence Event Package for the Session Initiation Protocol (SIP)", RFC 3856, August 2004.
8. Peterson, J. and C. Jennings, "Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP)", draft-ietf-sip-identity-06 (work in progress), Oct 2005.
9. Schulzrinne, H., Gurbani V., Kyzivat P., Rosenberg J., "RPID: Rich Presence Extensions to the Presence Information Data Format (PIDF)", Draft-ietf-simple-rpid-10 (work in progress), December 2005.
10. Ramsdell B., "S/MIME Version 3 Message Specification", RFC 2633, June 1999.
11. Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999.
12. Peterson J., "A Presence-based GEOPRIV Location Object Format", RFC 4119, December 2005.
13. Day, M., Rosenberg, J. and H. Sugano, "A Model for Presence and Instant Messaging", RFC 2778, February 2000.
14. Niemi, A., "Session Initiation Protocol (SIP) Extension for Event State Publication ", RFC 3903, October 2004.
15. Rosenberg, J., "Presence Authorization Rules" draft-ietf-simple-presence-rules-04, October, 2005.
16. Sugano, H., Fujimoto, S., Klyne, G., Bateman, A., Carr, W., and J. Peterson, "Presence Information Data Format (PIDF)", RFC 3863, August 2004.
17. Rosenberg, J., "A Data Model for Presence"draft-ietf-simple-presence-data-model-04, August 23, 2005.

18. Schulzrinne, H., Tschofenig H., Morris J., Cuellar J., Polk J., " A Document Format for Expressing Privacy Preferences for Location Information" draft-ietf-geopriv-policy-07.txt, Oct,2005.
19. Kent, S. and R. Atkinson, "Security Architecture for the Internet Protocol", RFC 2401, November 1998.
20. Saint-Andre, P., Ed., "Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence", RFC 3921, October 2004.
21. Rosenberg, J., "The Extensible Markup Language (XML) Configuration Access Protocol (XCAP)",draft-ietf-simple-xcap-08, Oct 2005.
22. Schulzrinne, H., "A Document Format for Expressing Privacy Preferences", draft-ietf-geopriv-common-policy-06, Oct 2005.
23. Peterson, J., "A Presence-based GEOPRIV Location Object Format", RFC 4119, December 2005.
24. Danley, M., Mulligan, D., Morris, J., Peterson, J., "Threat Analysis of the Geopriv Protocol", RFC 3694, February 2004.
25. Jennings, C., Peterson, J., "Certificate Management Service for SIP.", draft-ietf-sipping-certs-02.tx, July, 2005
26. Yahoo, AOL, MSN messenger.