

**SIP Support for Application Initiation
draft-jennings-sip-app-info-01**

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of [Section 10 of RFC2026](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/1id-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on December 28, 2003.

Copyright Notice

Copyright (C) The Internet Society (2003). All Rights Reserved.

Abstract

This document describes SIP extensions to allow network elements to request a UA to initiate a scripted application that is associated with a dialog. It provides a mechanism for the network elements to find out a UA's ability to fetch and execute scripts.

This work is being discussed on the sip@ietf.org mailing list.

Table of Contents

1.	Conventions	3
2.	Introduction	3
3.	Overview	3
4.	User Agent Server Behavior	5
5.	Proxy Behavior	6
6.	Formal Syntax	6
6.1	The App-Info Header	6
7.	Security Considerations	7
8.	IANA Considerations	8
8.1	Registration of App-Info Header	8
8.2	Registration of Option Tag	8
9.	Acknowledgements	8
	Normative References	8
	Informative References	9
	Author's Address	10
	Intellectual Property and Copyright Statements	11

1. Conventions

A "script" refers to some markup, program, or script that the UA can fetch and execute.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC-2119](#) [2].

2. Introduction

The Session Initiation Protocol (SIP) [1] provides the ability for users to initiate, manage, and terminate communications sessions. Frequently, these sessions will involve a SIP application. A SIP application is defined as a program running on a SIP-based element (such as a proxy or user agent) that provides some value-added function to a user or system administrator. Examples of SIP applications include pre-paid calling card calls, conferencing, and presence-based [4] call routing.

In order for most applications to properly function, they need input from the user to guide their operation. For example, a pre-paid calling card application requires the user to input their calling card number, their PIN code, and the destination number they wish to reach. The process by which a user provides input to an application is referred to as "application interaction".

A set of high level requirements on a system for application interaction are described in [9]. To meet these requirements, a framework has been developed[11]. In this framework, applications can instantiate user interface classes on client devices, for the purposes of interacting with the user. Each class can represent different components of the the user interface in a single application. These user interface components are described using markup languages, such as VoiceXML and KPML [10]. The framework also defines a set of requirements for SIP extensions that allow for an application to discover the capabilities of the user device for supporting markup languages, for placing user interface components on the device, and for terminating the component. This document proposes a specific SIP extension that fulfills those requirements. This extension is the App-Info header.

3. Overview

The main mechanism of this draft is a new header field, called App-Info, that provides the UA with the URL of a script to execute. A network element can add this header field. The App-Info header field can occur in most SIP messages, including INVITE and MESSAGE

messages, as well as in reliable provisional responses.

This draft also defines an option tag for use in the supported header field to allow a UA to indicate support for this mechanism. Options tags are also used to indicate which types of markup are supported. The Caller Preferences mechanism is used to indicate the types of markup that the UA can process.

An example App-Info header field is:

```
App-Info: "Call Timer"  
         <http://mediasvr.provider.net/calltimer.vxml>;  
         id=app4323!sub4+svr56.provider.net
```

This indicates that the UA should fetch and execute the script found at <http://mediasvr.provider.net/calltimer.vxml>. A key part of the header field value is an application id that consists of an application instance and an application class separated by a "!". In the example above, the instance is "app4323" and the class is "sub4+src56.provider.net". The combination of these two MUST make a globally unique identifier. There may be multiple user interface components running on a UA that are part of the same application instance, and therefore, share the same instance identifier. The instance identifier can be used to correlate the applications. An application might want to authorize the execution of the markup based only on the class name so that this authorization can be cached and used the next time a different instance of this same application is used. If authorization is cached or shared, note the warnings in the Security Consideration section. The UA may use the display name for presentation purposes and for help in managing focus, but it has no other meaning. The formal syntax for the App-info header field is presented in [Section 6](#).

This approach also uses the Supported and Accept header fields as well as the schemes mechanism from the caller prefs draft [\[7\]](#). For example:

```
Supported: markup  
Accept: multipart/mixed, application/vxml, text/html  
Contact: sip:1.2.3.4;schemes="http,cid,file"
```

This indicates that the UA can accept markup as defined in this draft. In particular the UA can accept VoiceXML and HTML markup and is capable fetching scripts from using a http, cid, or file scheme. The cid scheme^[8] fetches the content from an inline body in the same message.

Jennings

Expires December 28, 2003

[Page 4]

4. User Agent Server Behavior

When a UAC sends a message it MUST include in the Supported header the markup option tag. It MUST also put the markups or scripts that it can process in the Accept header field and indicate the schemes it can support in the Contact header.

When a UA receives a message that contains an App-Info header field, it must process each header field value and decide what to do with it. There are three cases: creating a new application, updating an existing application, and stopping a script that has been previously started.

In the case when the application identifier does not match any of the scripts that are currently running, a new application instance is created. The UA associates the application identifier with the dialog it was received on. The script is fetched. After fetching the script, execution starts in a context associated with the dialog.

If the application identifier matches the identifier for a previously fetched script, and the App-Info header field URL value does not match the previous header field URL value for this script, then the script is fetched and then used to replace the existing script. If the application identifier matches an existing script and the URL in the App-Info header field value is empty, then the existing script is terminated. If the URL has not changed, this header field value is ignored.

The UA fetches the script by using the URI found in the URI portion of the App-Info header field value. A UA which supports the App-Info header field MUST support fetching scripts from multipart MIME bodies using the cid scheme and SHOULD support the http[12] scheme.

When a dialog ends, all the applications associated with it SHOULD be immediately terminated.

A UA may add the App-Info header field to initiate an application on the other UA in the dialog. The UA SHOULD NOT request services that the other UA has not indicated it supports.

The App-Info header follows the general http scripting model for reporting errors - that is to say that the UA fetching the markup and executing it does not report errors to the application server. Part of the problem is that there is no easy way to report errors when fetching some markup fails, or when the markup can not be interpreted, or when the markup is terminated. There is no guarantee that the system will get any stimulus from the markup even if there are no errors. This means the application must be written in a way

that detects when the application is not getting input and deals with this situation appropriately. For many web applications this has turned out to be fairly easy to do.

5. Proxy Behavior

Proxies MAY add header field values to the App-Info header field but they SHOULD NOT delete or modify any existing header field values that they did not originally add. App-Info header fields MAY be added to reliable provisional response. The proxy SHOULD NOT request services that the UA has not indicated it supports.

6. Formal Syntax

The following syntax specification uses the augmented Backus-Naur Form (BNF) as described in [RFC-2234](#) [3].

6.1 The App-Info Header

```

App-Info      = "App-Info" HCOLON app *(COMMA app)
app           = [ display-name ] LAQUOT [absolute-uri] RAQUOT
                  *(SEMI app-param)
app-param     = app-id-param / app-name-param / generic-param
app-id-param  = "id" EQUAL app-id-value
app-id-value  = app-instance-id "!" app-class-id
app-instance-id = app-token
app-class-id  = app-token
app-token     = 1*(alphanum / "-" / "." / "%" / "*" / "_" / "+"
                  / "'" / "`" / "~" ) ; this is a token with no "!"
app-name-param = "app-name" EQUAL gen-value

```

This document adds the following entry to Table 2 of [RFC-3261](#) [1]. Note that App-Info is only valid in something that is in a dialog or forms a dialog.

Header field	where	proxy	ACK	BYE	CAN	INV	OPT	REG
-----	-----	-----	---	---	---	---	---	---
App-Info		adr	-	-	-	0	0	-
			SUB	NOT	REF	INF	UPD	PRA
			---	---	---	---	---	---
			0	0	0	0	0	-

In addition it would be listed as an optional header for the MESSAGE message so this document adds the following line to Table 1 in [RFC-3428](#) [6].

Jennings

Expires December 28, 2003

[Page 6]

Header Field	where proxy MESSAGE
--------------	---------------------

App-Info	o
----------	---

7. Security Considerations

This document describes a mechanism that allows non trusted parties to request that a UA execute an arbitrary script. This mechanism should only be used to initiate scripts that are in scripting languages intended for situations in which scripts from non trusted parties are expected. HTML is a good example of a markup language that is considered safe to render content that is not trusted. A scripting language that allowed scripts that automatically caused the UA to hang up and then dial a toll service phone number would certainly not be appropriate for this mechanism. The scripting language should not be able to access information on the UA that is not associated with the dialog, such as the user's address book.

The mechanism in this document does not address the secure transport, authorization, and integrity of the markup, but there are multiple mechanisms to ensure that the markup came from the correct party and that the markup has not been tampered with. If the script is fetched using https, both the identity of the party providing the markup and the integrity of the markup can be secured. Alternatively, if the script is transferred using the cid scheme, S/MIME can be used to sign the markup.

Proxies are allowed to insert App-Info headers so the App-Info headers can not be secured using an end to end mechanism. Using the sips URL in the SIP messages does provide some assurances that, as long as the user trusts all the proxies that the call traversed, the user can be sure that no rogue markup has been requested.

When some markup has been requested, the UA should require authorization to run it. It is possible to authenticate the party that sent the markup using https or S/MIME signing of cid markup. Authorization SHOULD be obtained for each different class of markup but the UA MAY reuse the authorization for two different instances of the same application class. When authorization is cached or used for different instances of the same class, the system MUST ensure that all the markup sharing the authorization came from the same entity. For example, if the system authorizes a class named doSomething that was received from a source which could authenticate (using something like https or smime) as good.example.com, the system should not trust a class called doSomething received from evil.example.com.

8. IANA Considerations

{NOTE to IANA: Please replace XXXX with the rfc number of this specification}

8.1 Registration of App-Info Header

This document defines a new header field, "App-Info". As recommended by [RFC-3261](#) [1] these headers fields should be registered by the IANA in the SIP header registry, using the RFC number of this document as its reference.

Name of Header: App-Info

Short form: none

Registrant: Cullen Jennings
fluffy@cisco.com

Normative description: [Section 6.1](#) of RFC XXXX.

8.2 Registration of Option Tag

This specification registers a new option tag called markup. The required information for this registration, as specified in [RFC-3261](#) [1], is:

Name: markup

Description: This option tag is for fetching scripts into a UA. When present in a Supported header field, it indicates that the UA can supports the mechanism in RFC XXXX.

Registrant: Cullen Jennings
fluffy@cisco.com

9. Acknowledgements

Eric Burger, Robert Fairlie-Cuninghame, Jonathan Rosenberg, and I were the members of the Application Stimulus Signaling Design Team. All members of the team contributed significantly to this work. In addition, thanks to Bert Culpepper.

Normative 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] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.
- [3] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", [RFC 2234](#), November 1997.
- [4] Day, M., Rosenberg, J. and H. Sugano, "A Model for Presence and Instant Messaging", [RFC 2778](#), February 2000.
- [5] Resnick, P., "Internet Message Format", [RFC 2822](#), April 2001.
- [6] Campbell, B., Rosenberg, J., Schulzrinne, H., Huitema, C. and D. Gurle, "Session Initiation Protocol (SIP) Extension for Instant Messaging", [RFC 3428](#), December 2002.
- [7] Rosenberg, J., Schulzrinne, H. and P. Kyzivat, "Caller Preferences and Callee Capabilities for the Session Initiation Protocol (SIP)", [draft-ietf-sip-callerprefs-08](#) (work in progress), March 2003.
- [8] Levinson, E., "Content-ID and Message-ID Uniform Resource Locators", [RFC 2111](#), March 1997.

Informative References

- [9] Culpepper, B. and R. Fairlie-Cunninghame, "Session Initiation Protocol Based Application Interaction Requirements", [draft-culpepper-sipping-app-interact-reqs-03](#) (work in progress), March 2003.
- [10] Burger, E., "Keypad Markup Language (KPML)", [draft-burger-sipping-kpml-02](#) (work in progress), June 2003.
- [11] Rosenberg, J., "A Framework and Requirements for Application Interaction in SIP", [draft-rosenberg-sipping-app-interaction-framework-01](#) (work in progress), June 2003.
- [12] Fielding, R., Gettys, J., Mogul, J., Nielsen, H., Masinter, L., Leach, P. and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", [RFC 2616](#), June 1999.

Author's Address

Cullen Jennings
Cisco Systems
170 West Tasman Drive
MS: SJC-21/3
San Jose, CA 95134
USA

Phone: +1 408 527-9132
EMail: fluffy@cisco.com

Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in [BCP-11](#). Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

Full Copyright Statement

Copyright (C) The Internet Society (2003). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assignees.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION

HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF
MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgement

Funding for the RFC Editor function is currently provided by the
Internet Society.