Network Working Group
Internet-Draft
Expires: September 3, 2001

D. Gupta
HP
T. Buchheim
B. Feinstein
G. Matthews
R. Pollock
Harvey Mudd College
March 5, 2001

IAP: Intrusion Alert Protocol
draft-ietf-idwg-iap-05

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 September 3, 2001.

Copyright Notice

Abstract

Intrusion Alert Protocol (IAP) is an application--level protocol for exchanging intrusion alert data between intrusion detection elements, notably sensor/analyzers and managers, across IP networks. The specification of alerts carried using this protocol is described in the Intrusion Detection Message Exchange Format (IDMEF) [5], a companion document of the Intrusion Detection Working Group of the IETF.

Table of Contents

1. Introduction

1.1 Purpose

The Intrusion Alert Protocol (IAP) is an application--level protocol for exchanging intrusion alert data. The protocol is designed to provide the necessary transport and security properties to allow sensitive alert data to be sent across IP networks. In addition, the protocol is designed so that future extensions may use the application layer for configuring intrusion detection sensor/analyzers or sending responses (the current charter of the working group does not include configuration response messages).

1.2 How to Read This Document

The key words "MUST", "MUST NOT", "SHOULD", "SHOULD NOT", and "MAY" in this document are to be interpreted as defined in "Key words for use in RFCs to Indicate Requirement Levels" [3].

1.3 Transport Layer Protocol

IAP uses the Transmission Control Protocol (TCP) [6] as its underlying transport layer mechanism. TCP is used for a wide variety of IP services. It has a number of features such as congestion avoidance, slow start, etc. that enable it to work over large networks with a wide variety of latency, bandwidth and congestion characteristics. TCP provides reliable and sequenced delivery of data between IP peers.

1.4 Terminology

Terminology is borrowed from [13].

1.5 Overall Operation

IAP is primarily oriented towards supporting the transmission of alert data from an intrusion detection sensor/analyzer, which detects a potential intrusion, to a manager, that displays the alert to a human, logs it to a database or takes appropriate action.

In the simplest case, a sensor/analyzer (SA) sends alerts to a manager(M).

```
                  SA  ------------------->  M
```

In a more complex situation, there are more than one intermediary in the communication path. Two common forms of intermediary are:

Proxy: A proxy is a forwarding agent which MAY do some rewriting of the content, and forward the message. A proxy MAY be used to relay two connections when the communication needs to pass through an intermediary such as a firewall.

Gateway: A gateway is used to translate messages from/to some native format (e.g. from SNMPv3 UDP wire protocol to IDMEF [5]).

A proxy may be used to relay two connections when the communication needs to pass through an intermediary such as a firewall.

An example of a connection with intermediaries is shown below:

```
SA ----> P  -----> G ----> M
```

Here, IDMEF [5] data is generated by SA and passed to a proxy P. P connects to a gateway G, which translates alerts into a native format to be consumed by the manager M. In this exchange, the connections between SA and P, and P and G use IAP. The connection between G and M uses the native protocol supported by M.

IAP communication takes place over an IP network using TCP. To connect with other networks, gateways should be used to transform protocol data into the native representation. It is expected that the IANA will allocate a designated TCP port for IAP communications.

1.6 Augmented BNF

We use the augmented BNF [4] definitions of HTTP/1.1 [8] to specify the grammar for the protocol.

1.7 Protocol Parameters

IAP uses a <major>.<minor> numbering scheme to indicate versions of the protocol. The minor number is changed when updates to the protocol add features that do not change the parsing algorithm. The major number is changed when the parsing algorithm is modified. The exception to this is the 0.x preliminary versions, where each new minor version number change may indicate a parsing change.

This document describes version 0.5 of IAP. The version string for this is IAP/0.5, and is denoted by iap-version.

1.8 Media Types

IAP uses Multipurpose Internet Mail Extensions (MIME) (see [2], [9], and [10]) to denote the type of alert data. MIME Content-Types are used to define the encapsulation of data, in a manner that can be

extended without requiring changes to the application protocol. It is expected that the alert data will be sent using the XML data type definitions from a companion document of the working group.

The only Content-Type used by IAP/0.5 is application/xml. IAP entities MUST accept data sent in this format.

2. IAP Communication Model

IAP communications occur on top of TCP. The TCP connection carries request-response pairs much like the payload of HTTP [8]. The TCP connection MAY be initiated by the analyzer SA or the manager M. This is to accommodate security perimeter configurations that proscribe certain kinds of connections. For instance, if the manager resides in an outsourcer's environment whereas the analyzer is inside a protected network, perimeter security needs of the protected network may be better served if the analyzer initiates the connection.

In order to accommodate this variation, the roles of IAP entities are not determined by who initiates the TCP connection. This is in contrast to HTTP where the client (user agent) always initiates the TCP connection. Response codes are borrowed from the specification of version 1.1 of HTTP.

2.1 Why Not HTTP?

HTTP/1.1 satisfies a number of needs of the application, but has a few assumptions that make it difficult to use in-toto as the application layer. The first is the assumption that the TCP connection initiator is the HTTP client or requester in the request-response scheme - this precludes running the protocol "in reverse". A second issue is the inability to give the proxy enough hints about the connection if the protocol is being run over a TLS session, although there is ongoing work in that direction. This draft uses the recommendations of that work, in particular the Upgrade: keyword and the CONNECT method as outlined in [11].

The protocol specifies the use of the CONNECT method to establish a transparent tunnel between IAP peers. Intermediate proxies enable connectivity across security perimeters and handle issues with DNS resolution limitations inside security perimeters. The use of HTTP protocol elements allows an IAP entity to provide virtual hosting thus enabling public IPv4 addresses to be conserved while permitting security properties to be established about the peers.

2.2 Request-Response

A configured IAP sensor/analyzer - manager pair MAY have two simultaneous active connections, one for analyzer-initiated flow and another for manager-initiated flow. These active flows MUST be carried over separate TCP connections.

An IAP entity MAY close a connection if it finds unexpected data from its peer.

An analyzer MUST implement connections in which it sends the requests once the channel setup is complete. It MAY choose to implement connections in which it responds to requests from a manager.

2.3 Phases

The protocol is divided into two major phases. The first (setup) phase is a request-response protocol where requests are sent by the peer that initiated the TCP connection. This establishes roles for the peers. The two parties agree whether the connection will be used for request-response pairs where the analyzer acts as the sender, or vice versa.

The second (data) phase is also a request-response session in which the sender of requests may be reversed, based on the negotiations in the first phase. If the connection is used for carrying payload initiated by the analyzer, such as alerts, the requests will be from the analyzer. Otherwise, they will be from the manager.

2.3.1 Proxies

An IDMEF entity in a proxy role does not have an IAP identity. It acts as a relay of messages without understanding their content. It MAY do some rewriting of the content, but in a manner that does not impact the security properties of alerts.

3. IAP Setup Phase

This is the first phase after a TCP connection is in the established state. In this phase, the two parties set up the transport parameters of the protocol. An IAP entity in a proxy role MAY rewrite content to set up the protocol in this phase. This phase uses a request--response form, with the TCP connection initiator as requestor. The phase is divided into three subphases, where the TCP connection, security, and channel parameters are agreed upon by the peers.

3.1 TCP Setup

The TCP connection initiator sends an iap-connect-request message.

A corresponding peer MAY choose to accept this connection, and respond using an iap-response message, with the status code set to 200 to denote success. The pair can then proceed to the security setup section.

Alternatively, the entity MAY reject the connection request by setting the status code to 4xx to denote failure. In particular, the 403 Forbidden status MAY be used by the responder.

An intermediate entity in a proxy role MAY, upon receiving the request, rewrite the iap-connect-request message. A proxy MAY append a Via: header to the connection request before passing it on to the receiving entity. The proxy MUST NOT make arbitrary connections based on the host specified in the connection request since this could expose the internal network. Instead, it SHOULD send the message to a preset next-hop host, possibly selecting one based on fields such as the destination and source given in the connection request.

A proxy SHOULD relay the server's response back to the client after appending a Via: header. A proxy MUST verify that existing Via: headers in the request don't match its own identity in order to limit the damage from misconfigured proxies. Otherwise, it MUST send a bad gateway (502) response to the peer that requested the connection.

Any entities in a proxy role MUST forward data transparently once the upgrade phase is reached. End entities detect any attempts to manipulate or inject messages into the communications channel.

3.2 Security Setup

A single request--response pair is used to upgrade the security of a connected transport. The initiator of the TCP connection, upon

receiving an iap-response message without any errors, issues an iap-upgrade-request.

A server should expect the iap-upgrade-request message after sending an iap-response reply indicating acceptance of the connection. The server SHOULD terminate if the request is not received, or some other request is received. Upon receiving the request, it SHOULD send an iap-response message with the status code set to 101 to indicate acceptance of the upgrade.

It MAY also send 500 to denote server configuration error, if for instance, it is not yet configured and does not have a certificate.

Once the TCP connection initiator receives an iap-response message indicating success of its request to upgrade the security of the connection, the parties initiate the handshake for the TLS 1.0 [7] protocol. This step negotiates the protocol version and cryptographic algorithms, performs mutual authentication, and generates shared secrets for the next phase. The TLS handshake is initiated by the originator of the TCP connection (client) which assumes the TLS client role. The connection initiator sends a TLS client-hello message to initiate the handshake.

If the parties complete the TLS handshake protocol successfully, the TCP initiator sends the channel setup request using the TLS record protocol. This is used by the parties to verify connection parameters.

3.3 Channel Setup

Data in the channel setup phase is sent using the TLS record layer, and is thus impervious to passive and active attacks. The purpose of this phase is to verify the IAP version information and decide on the roles each entity will take.

The TCP connection initiator sends the iap-channel-setup-request message. The recipient then:

> Verifies the version information against what it received during the protocol setup phase;
>
> Checks that it is able to either accept requests from, or send requests to, the peer.

It then issues a iap-response reply to indicate its agreement with the parameters specified by the TCP connection initiator.

The protocol, security and channel setup phases are now complete, and

the channel is ready to carry data. The directionality of this phase is dependent on the parameters agreed upon during channel setup. Note that the channel, once set up, can be used for sending multiple alerts. The parties can also implement some polling method that tears down connections after some period of no traffic, or some other polling strategy.

3.4 Secured Data Transport

This is the main phase of the protocol. In this phase, encoded IDMEF alerts are sent by the sender (sensor/analyzer) to the receiver (manager) over the TLS record layer. Note that since the IDMEF allows multiple alerts in a single message, more than one alert can be sent as part of the same Content-Body.

In addition to data in the form application/xml, compatible peers MAY send other data using this transport. A peer, upon receipt of data that it is unable to decode (unknown IAP content type), SHOULD reject the data. It MAY choose not to terminate the connection. This allows peers supporting richer content types to communicate with legacy applications.

3.5 Termination

Termination can be initiated by either peer by sending a TLS close-notify alert (section 7.2.1 of [7]). The recipient, on receipt of this alert, should in turn respond with a close-notify alert and the parties can close the connection gracefully.

3.6 Example

In this example, an analyzer A is configured to connect to proxy P. P connects to a manager M to deliver the alerts. The following diagram depicts the abstract message flow for the case where there are no errors, and a connection is set up to deliver alerts from A to M.

```
          A                        P                   M
   [TCP Setup]

     --------------->
     iap-connect-request


                            --------------->
                            iap-connect-request
                                              <---------------
```



```
                                              iap-response

                        <---------------
                        iap-response

   [proxy is now a transparent forwarding agent]
   [Security Setup]
     --------------->
     iap-upgrade-request

                                              <---------------
                                              iap-response

              (TLS handshake negotiation initiated by A)

   [TLS handshake completed: data sent using the TLS record layer]
   [Channel Setup]
     --------------->
     iap-channel-setup-request

                                              <---------------
                                              iap-response

   [Secured Data Transport]

     --------------->
     iap-content-request

                                              <---------------
                                              iap-response
```

4. IAP Wire Protocol

4.1 Syntax

The syntax of IAP is reminiscent of HTTP/1.1. Several definitions are identical to those in HTTP/1.1 and are not repeated here. The following definitions from [8] are used.

```
message-header

message-body

SP

CRLF
```

Like HTTP/1.1, IAP is a request--response protocol. IAP entities exchange messages, each of which is either a request or a response.

```
iap-message      = ( iap-request | iap-response )
```

An IAP message contains the appropriate start line, zero or more headers, a blank line, and possibly a message body. Note that some types of requests define required headers which MUST be included in those requests.

```
iap-request      = iap-request-start-line
                   *(message-header CRLF)
                   CRLF
                   [ message-body ]

iap-response
                 = iap-response-start-line
                   *(message-header CRLF)
                   CRLF
                   [ message-body ]
```

Message headers and bodies are defined in HTTP/1.1 [8].

If a message-body item is present in an IAP message, then a Content-Length header MUST be included in that message. In this version of IAP a message-body is only included in the iap-content-request. Also, none of the currently defined requests expect a message-body to be returned.

The version of the protocol is denoted by iap-version.

```
iap-version     = "IAP/0.5"
```


4.2 Protocol Messages

4.2.1 Requests

Several types of requests are defined in this version of IAP.

```
iap-request-start-line =

   ( iap-connect-request |
     iap-upgrade-request |
     iap-channel-setup-request |
     iap-content-request )
```

4.2.1.1 Connection Request

A iap-connect-request denotes a request message sent by an IAP client to establish an IAP protocol connection.

```
iap-connect-request
                  = "CONNECT" SP host ":" port SP iap-version CRLF
```

An iap-connect-request must have both Host: and Source: headers. These should contain the sender's own name and the name of its intended destination, respectively.

No message-body is defined for this request type and no message-body is defined for the response.

As noted in Section 3.1 a proxy may modify an iap-connect-request.

4.2.1.2 Upgrade Request

An iap-upgrade-request contains a notification from the client that it wishes to upgrade the security of the connection.

```
iap-config-url    = "/iap/config/"

iap-upgrade-request
                  = "PUT" SP iap-config-url SP iap-version CRLF
```

An Upgrade: header MUST be sent, indicating the version of TLS to be used.

```
                  "Upgrade: TLS/1.0" CRLF
```


No message-body is defined for this request type and no message-body is defined for the response.

4.2.1.3 Channel Setup

An iap-channel-setup-request contains a notification requesting that the peer verify the version of IAP being used. Furthermore, the request specifies the role played by each peer.

```
iap-channel-setup-request
                  = "PUT" SP iap-config-url SP iap-version CRLF
```

An IAP-Version: header MUST be included to verify the version of IAP in use.

```
                    "IAP-Version: 0.5" CRLF
```

An IAP-Role: header MUST be included to indicate the role of the TCP connection initiator.

No message-body is defined for this request type and no message-body is defined for the response.

```
                    "IAP-Role:" SP ("Sender" | "Receiver") CRLF
```

By choosing the appropriate string, it signals whether it wants to send requests to the peer or receive them from the peer.

4.2.1.4 Alert Content

The iap-content-request message requests that the Receiver accept content, for instance an encoded alert, sent using IAP.

```
iap-content-request
                  = "PUT" SP iap-content-url SP iap-version CRLF

iap-content-url   = "/iap/alert/"
```

A message-body is always sent with this type of request. Therefore a Content-Length: header MUST be sent. A Content-Type: header SHOULD be sent. The response to this request has no message-body defined.

```
iap-content-length
                  = "Content-Length:" SP +DIGIT CRLF

iap-content-type  = "Content-Type:" SP "application/xml" CRLF
```


4.2.2 Responses

All requests must be answered with an iap-response message indicating whether the recipient was able to successfully interpret (and act on) the request. Response codes are borrowed from HTTP/1.1 and have the same interpretation.

Although a response may contain a message-body, none of the requests defined in this version of IAP require a message-body in the response.

```
iap-response-start-line
                  = iap-version SP 3*DIGIT CRLF
```

4.3 Example

Here is a transcript of a scenario in which an IDMEF sensor/analyzer A wishes to send alerts to manager M. The proxy P mediates this connection. After the connection has been set up, A sends an IDMEF alert 3632 octets in length in an XML dialect.

```
          A                        P                    M
iap-connect-request
---------------->
CONNECT M.DOM.ORG:1443 IAP/0.5 CRLF
Host: M.DOM.ORG CRLF
Source: A.DOM.ORG CRLF
CRLF
                     iap-connect-request
                     ---------------->
                     CONNECT M.DOM.ORG:1443 IAP/0.5 CRLF
                     Host: M.DOM.ORG CRLF
                     Source: A.DOM.ORG CRLF
                     Via: P.DOM.ORG CRLF
                     CRLF
                                                iap-response
                                                <----------------
                                                IAP/0.5 200 OK CRLF
                                                CRLF
                     iap-response
                     <----------------
                     IAP/0.5 200 OK CRLF
                     Via: P.DOM.ORG CRLF
                     CRLF

[proxy is now a transparent forwarding agent]
iap-upgrade-request
```

```
   --------------->
   PUT /iap/config/ IAP/0.5 CRLF
   Upgrade: TLS/1.0 CRLF
   CRLF
                                                    iap-response
                                                    <---------------
                                                    IAP/0.5 101 CRLF
                                                    CRLF

                        (TLS handshake negotiation)

   [TLS handshake completed: data sent using the TLS record layer]
   iap-version-verify
   --------------->
   PUT /iap/config/ IAP/0.5 CRLF
   IAP-Version: 0.5 CRLF
   IAP-Role: Sender CRLF
   CRLF
                                                    iap-response
                                                    <---------------
                                                    IAP/0.5 200 CRLF
                                                    CRLF


   iap-content
   --------------->
   PUT /iap/alert IAP/0.5 CRLF
   Content-Type: application/xml CRLF
   Content-Length: 3632 CRLF
   CRLF
   <?xml version="1.0" encoding="UTF-8"?>

      <!DOCTYPE IDMEF-Message PUBLIC "-//IETF//DTD RFCxxxx IDMEF v0.3//EN"
        "idmef-message.dtd">

      <IDMEF-Message version="0.3">
        <Alert alertid="12345" impact="attempted-user">
          ...
          [removed for brevity]
          ...
        </Alert>
      </IDMEF-Message>

                                                    iap-response
                                                    <---------------
                                                    IAP/0.5 200 CRLF
                                                    CRLF
```


5. Scenarios

5.1 Simple Analyzer

A simple analyzer can only initiate connections, and only be able to connect with a single manager at a time. In this case, the analyzer's configuration is expected to include the manager's IP address and a specification of the manager's certificate (such as the signer's public key and patterns in the common name or organizational unit).

The analyzer would initiate the TCP session. As a result, any firewalls in the path MUST be configured to let through traffic initiated by the analyzer(s) with the manager's IP address and IAP destination port as the other endpoint of the session.

5.2 Simple Analyzer with Proxy

If perimeter security demands that a proxy be deployed, the analyzer should be configured with the proxy's IP address. The proxy would then establish a connection to the manager, and forward traffic between the two connections. The proxy's configuration would include the manager's address. Note that the proxy does not participate in the TLS handshake and does not need TLS related configuration parameters.

5.3 Manager Design Considerations

The protocol is oriented to allow lightweight implementation of sensor/analyzers. A manager MUST accept TCP connections from multiple sensor/analyzers by listening on the IAP port.

6. Implementation Considerations

6.1 TCP Connection Initiation

The entity that initiates a TCP connection will be variable, and dependent on the exact deployment model. One scenario is that of sensor/analyzers outside a security perimeter, with the manager inside. In such configurations, the manager MAY initiate the connection in line with perimeter security requirements.

Another scenario is that of remote sensor/analyzers being managed by a service provider. In this case, the sensor/analyzer MAY contact a proxy on the perimeter, which in turn connects to the remote manager in a service provider's network.

The communications protocol should allow chained proxies to carry IAP data across multiple security perimeters.

6.2 Urgent Data

Urgent data at the TCP layer MUST NOT be used by an entity using IAP. Endpoints MUST ignore urgent data and MAY choose to terminate a connection upon receipt of urgent data, because this data is not protected by TLS and could be forged.

6.3 TLS/1.0 Protocol

The TLS 1.0 protocol MUST be used. An IDMEF entity MUST not allow older protocol HELLO messages, and reject attempts to negotiate an older version of the protocol. The following TLS ciphersuite MUST be supported in line with recommendations from the TLS working group:

    TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA

The recommendations in sections 2.1 and 2.2 of [12] must be followed by IAP client and server implementations.

6.4 Mandatory Certificates

In line with the requirement for strong mutual authentication, certificates for sensor/analyzers acting in IAP client or server roles are mandatory. Each entity should verify the peer's certificate during the TLS handshake in accordance with the administrative domain's certificate verification policies.

6.5 Certificate Conventions

One or more of the following extended key usage extensions MUST be

specified in X.509v3 certificates issued to an IAP client or server entity:

    IAP_ALERT_GENERATOR

    IAP_ALERT_CONSUMER

    IAP_IDMEF_CONFIGURATOR

The object identifiers (OIDs) for these extensions will be specified in a companion document. The presence of the extension means that the signer believes that the holder of the certificate is allowed to function in the corresponding IAP role.

A manager MUST have the IAP_ALERT_CONSUMER extension in its certificate. Similarly, an analyzer MUST have the IAP_ALERT_GENERATOR extension in its certificate.

6.6 Verification of Peer Identity

As the peer identity is initially sent in the clear, it is essential that IAP entities verify the identity of their peer using criteria based on their public key certificates. Implementors SHOULD adopt prudent security practice and reject certificates that are not in accordance with the installation's configuration. For instance, they may wish to verify subfields of the peer's identity, such as the Organizational Unit, in addition to verifying the correctness of the signature and the signer's identity.

The version of the protocol MUST be verified during the channel setup phase to stop protocol downgrade attacks. The mechanism specified in section 2.4 of [12] for verifying peer certificates MUST be followed.

6.7 TLS Session Resumption

Any entity that listens for iap-connect-requests MUST be configured to support TLS session resumption. Because of the possibility of external entities maliciously terminating IAP sessions, clients and servers MAY attempt to resume a session if the TLS close-notify alert was not received before the transport closed.

7. Security Considerations

As IAP is intended to be used for carrying security--sensitive data, we will list a number of security considerations.

7.1 Reliable and Sequenced Delivery

Although reliable and sequenced delivery can be built on top of UDP, such implementations usually reimplement some of the protocol features of TCP. Certain deployment scenarios may prefer fast unreliable delivery with the manager doing a "best effort" attempt to keep up with the flow of alerts. This proposal does not address such a scenario. One potential alternative for this scenario is to use the SNMP trap as a means to represent the alert. We remark that the above scenario is at odds with a number of items in section 6 of the requirements document of the working group. In addition, the use of UDP-based protocols is likely to result in unresponsive or aggressive flows which further exacerbate the congestion problem in the Internet.

7.2 TCP Handshake

TCP uses a three--message handshake mechanism to set up connections. This opens the protocol up to certain well-known denial of service attacks. IAP does not address this vulnerability. The effect of such attacks can be mitigated by a number of techniques, including:

restricting the set of peer IP addresses allowed to connect to the node.

restricting the number of open connections permitted to each known peer.

restricting the number of open connections permitted to subsets of known peers.

7.3 Key Management

For a public--key based communications model to be useful, good key management principles must be used for the lifecycle of public key certificates. The pkix working group of the IETF is defining mechanisms that can be used for this purpose [1].

7.4 Message Length

Traffic analysis may allow an observer to induce the type of alert from the size of the message. If this is a threat, IDMEF entities

SHOULD pad their data so that it observes some known distribution over time.

7.5 Proxy Considerations

As the proxy transparently forwards encrypted traffic after connections are established, it is prudent to deploy the proxy in a manner that it will not be used to violate the perimeter security policy. For instance, a proxy may only accept requests from connections on its inside interface to known locations outside the security perimeter.

8. Acknowledgements

This document makes heavy use of prior work in the IETF on HTTP, MIME and TLS. Their effort is gratefully acknowledged. Members of the IETF's intrusion detection working group (idwg) have made extensive comments that are reflected in the draft.

References

[1] Adams, C. and S. Farrell, "Internet X.509 Public Key Infrastructure Certificate Management Protocols", RFC 2510, March 1999.

[2] Borenstein, N. and N. Freed, "MIME (Multipurpose Internet Mail Extensions): Mechanisms for Specifying and Describing the Format of Internet Message Bodies", RFC 1341, June 1992.

[3] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[4] Crocker, D., "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

[5] Curry, D. and H. Debar, "Intrusion Detection Message Exchange Format Data Model and Extensible Markup Language (XML) Document Type Definition", February 2001, <draft-ietf-idwg-idmef-xml-03 (work in progress)>.

[6] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, Sep 1981.

[7] Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999.

[8] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P. and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[9] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996.

[10] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, November 1996.

[11] Khare, R. and S. Lawrence, "Upgrading to TLS Within HTTP/1.1", RFC 2817, May 2000.

[12] Newman, C., "Using TLS with IMAP, POP3 and ACAP", RFC 2595, June 1999.

[13] Wood, M. and M. Erlinger, "Intrusion Detection Message Exchange Requirements", February 2001, <draft-ietf-idwg-requirements-05 (work in progress)>.

Authors' Addresses

Dipankar Gupta
Hewlett-Packard
3404 E Harmony Road
MS A2
Fort Collins, CO 80525
US

Fax: +1 970 898 3077
EMail: dg@fc.hp.com

Timothy C. Buchheim
Harvey Mudd College

EMail: tcb@cs.hmc.edu
URI: http://www.cs.hmc.edu/

Benjamin S. Feinstein
Harvey Mudd College

EMail: bfeinste@cs.hmc.edu
URI: http://www.cs.hmc.edu/

Gregory A. Matthews
Harvey Mudd College

EMail: gmatthew@cs.hmc.edu
URI: http://www.cs.hmc.edu/

Roy A. Pollock
Harvey Mudd College

EMail: rpollock@cs.hmc.edu
URI: http://www.cs.hmc.edu/

Full Copyright Statement

Acknowledgement

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