Network Working Group
Request for Comments: 1339

S. Dorner
P. Resnick
U. of Illinois at Urbana-Champaign
June 1992

Remote Mail Checking Protocol

Status of this Memo

This memo defines an Experimental Protocol for the Internet community. Discussion and suggestions for improvement are requested. Please refer to the current edition of the "IAB Official Protocol Standards" for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Abstract

This RFC defines a protocol to provide a mail checking service to be used between a client and server pair. Typically, a small program on a client workstation would use the protocol to query a server in order to find out whether new mail has arrived for a specified user.

Intent

This RFC defines a simple, low-overhead protocol for checking the status of a maildrop on a host. It is primarily intended for use in adjunct with "remote mail" servers such as those implementing the Post Office Protocol (RFC 1225). Remote mail clients must poll their servers to discover the arrival of mail. Using one of the remote mail protocols for periodic checking can be quite impractical and expensive for the server since either a constant connection between client and server must be maintained or repeated and expensive user validations must be done. Furthermore, users on less capable computers may not wish to devote the memory required to have a full implementation of the client polling for mail. Thus, we feel that an easy to implement and inexpensive to use polling scheme would be of benefit both to mail servers and their clients.

Protocol Overview

To avoid connection overhead, the Remote Mail Checking Protocol is based on the User Datagram Protocol (UDP), using UDP port 50 decimal (62 octal) for the server. The protocol provides for both non-authenticated and authenticated polling. Non-authenticated polling is simplest for both client and server. Authenticated polling provides a small increment of privacy, at the cost of more complexity in both client and server (but still far less than polling with one of the

remote mail protocols).

Non-Authenticated Protocol

In the non-authenticated version of the protocol, the server will listen on port 50 for maildrop check requests for users with maildrops on the machine. A client will send a single UDP datagram from a randomly chosen unreserved UDP port to UDP port 50 on the server. The datagram will contain a 32-bit (four-octet) number which is set to all zeros (0), followed by a case-sensitive ASCII string of a username on the server system. The server will find the maildrop on the system for that user and determine the amount of time that has passed since the last message in the maildrop was appended, as well as the amount of time that has passed since the maildrop was last accessed for reading. The server will then send back a single UDP datagram containing three 32-bit numbers in network byte order to the originating port on the client. Again, the first will be zero (0), the second will contain the number of seconds plus one since the last addition to the specified user's maildrop and the third will contain the number of seconds plus one since the last read on the user's maildrop. If the username provided does not exist, if the maildrop is not on the system or if the maildrop is empty, the server will send back zero (0) in the last two numbers for its reply. The client will consider the maildrop to contain new mail if the number of seconds since the last read access is greater than or equal to the number of seconds since the last addition access of the maildrop and either number is non-zero, old mail if the number of seconds since the last read access is less than or equal to the number of seconds since the last addition access of the maildrop and either number is non-zero, and empty if both numbers are zero.

Authenticated Protocol

The authenticated protocol operates identically to the non-authenticated protocol with the exception of the first interaction between the server and the client. After the client has sent its initial request containing the requested username, the server will send back a single UDP packet containing three 32-bit numbers. The first number will be a bit-mask instead of the normal 32-bits of zero. The bit-mask will indicate a request for authentication. Each bit in the mask represents a type of authentication that the server accepts. The bits (with the least significant bit numbered 0, and the most significant bit 31) are defined as follows:

0 Cleartext password The password for the maildrop, not NULL-terminated.

1-23 Reserved for future use

24-31 Implementation-dependent. Implementors wishing to experiment may use these.

For each type of authentication that the server accepts, the corresponding bit will be set to one. All other bits will be set to zero. The last two 32-bit numbers in the reply will be set to zero. If the client supports authentication, it will send back a 32-bit mask with the bit representing the kind of authentication it is using set to one, followed by the data used for authentication. The client is free to use any of the types of authentication indicated by the authentication request from the server. If the client does not support authentication and it receives an authentication request, it SHOULD stop sending requests (though this behavior is not required).

Once a valid authentication is received by the server for a particular maildrop, the server considers the IP address and UDP port of the client along with that maildrop to be an authenticated address/port/maildrop triple. From then on, normal non-authenticated transactions take place between the server and the client as described above. Should a datagram come from an authenticated address/port pair with a different username, or if some amount of time has elapsed since the last request (which is implementation dependent), the server should remove the address/port/maildrop triple from its list of authenticated triples and send another authentication request. Since the time required for an authenticated triple to become unauthenticated is implementation dependent, clients should be prepared to send an authentication reply to containing the server whenever it is requested.

Server Implementation Notes

Servers which implement either the authenticated or non-authenticated protocol may decide that they do not wish to reveal the actual amount of time that has passed since the last update or read from a maildrop. (See the "Security Considerations" section below for reasons some feel this is problematic.) In this case, a server may instead reply with the following:

	First 32 bits	Second 32 bits	Third 32 bits
New mail	0	0	1
Old mail	0	1	0
No mail	0	0	0

These values will appear to the client as correctly representing new, old or no mail respectively but will give no indication of the actual times that the changes took place.

Servers implementing the non-authenticated protocol MUST provide some mechanism by which users on the system can give permission for their maildrops to accessed by the protocol. See the "Security Considerations" section below for specifics.

Client Implementation Notes

Clients MUST not send more than one poll (and one authentication) per minute. In particular, lack of server response should not result in retransmission.

Since the last two numbers in an authentication request from a server are always 0 as are the last two numbers in a response for an empty or non-existent maildrop, clients that do not support authentication need not examine the first number in the server datagram at all (though they are encouraged to do so for the sake of proper reporting to the user).

Clients can turn the modification interval into absolute time, and track the changing of this absolute time in order to discern the arrival of new mail (as opposed to the mere existence of unread mail). However, such clients should bear three things in mind. First, network delays and clock vagaries may result in small inconsistencies in times. A "slop factor" of several seconds is encouraged. Second, the reading of mail often entails modification of the maildrop; the relationship of the access and modification intervals should always be consulted. Third, the special results of (1,0) and (0,1) are most properly handled as special cases.

Clients need not recall whether or not they are authenticated (though they must use a consistent port if they receive any authentication requests for a given maildrop). It is sufficient to issue requests when desired, and to respond to any authentication requests that appear.

Security Considerations

The are two security considerations for the protocol. The first is one mainly of privacy. Some sites and individual users consider it problematic to have information about mail arrival available freely. This can be a simple privacy issue for individuals or a security issue for highly secure sites. The authenticated version of the protocol allows sites to have a reasonable amount of security in that only people with passwords can access this information. The protocol

currently only uses cleartext passwords, but can be simply modified to use other authentication formats. The scheme mentioned in "Server Implementation Notes" of using only (0,1) and (1,0) in the responses also may limit access to some types of information. Implementations that do not use the authenticated scheme MUST have a mechanism by which a user can give consent to have this information made available; the default for the unauthenticated implementation should be that a user's maildrop cannot be accessed until consent of the user is given. (For example, UNIX server implementations may wish to make use of the "owner execute" bit to indicate whether a particular maildrop allows use of the unauthenticated protocol. If this is done, a single "stat" call can be used to gather all information required to respond to a poll.) Servers which do not implement authentication should simply return a zero-filled datagram for maildrops which don't have permission.

The other security consideration involves unknown maildrops and usernames. Some site administrators consider it a security risk give out any information which would reveal the existence or non-existence of a certain username or maildrop on the system. For this reason, we have chosen to have the server send back a zero-filled datagram as the response to either a request for an unknown username or a maildrop that does not exist or is empty. In this way, potential security violations are limited, since there is no way to tell the difference between an empty maildrop and non-existent maildrop, and also no way to tell if the user exists on the system or not. If greater security is desired, the protocol should probably not be run in the first place.

Authors' Addresses

Steve Dorner
Digital Computer Laboratory
University of Illinois at Urbana-Champaign
1304 West Springfield Avenue
Urbana, Illinois 61801

Phone: (217) 244-1765
EMail: s-dorner@uiuc.edu

Pete Resnick
The Beckman Institute
University of Illinois at Urbana-Champaign
405 North Mathews Avenue
Urbana, Illinois 61801

Phone: (217) 244-1265
EMail: resnick@cogsci.uiuc.edu