Internet Engineering Task Force (IETF)
Request for Comments: 7151
Updates: 959
Category: Standards Track
ISSN: 2070-1721

P. Hethmon
Hethmon Brothers
R. McMurray
Microsoft Corporation

March 2014

File Transfer Protocol HOST Command for Virtual Hosts

Abstract

The File Transfer Protocol, as defined in RFC 959, does not provide a way for FTP clients and servers to differentiate between multiple DNS names that are registered for a single IP address. This document defines a new FTP command that provides a mechanism for FTP clients and servers to identify individual virtual hosts on an FTP server.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc7151.

Table of Contents

1. Introduction 2
2. Document Conventions 3
2.1. Basic Tokens 3
2.2. Server Replies 4
3. The HOST Command 4
3.1. Syntax of the HOST Command 5
3.2. HOST Command Semantics 7
3.2.1. REIN Command Semantics 8
3.2.2. User-PI Usage of HOST 9
3.2.3. State Diagrams 11
3.3. HOST Command Errors 16
3.4. FEAT Response for HOST Command 17
4. Security Considerations 17
5. IANA Considerations 19
6. References 19
6.1. Normative References 19
6.2. Informative References 20
Appendix A. Unworkable Alternatives 21
A.1. Overloading the CWD Command 21
A.2. Overloading the ACCT Command 21
A.3. Overloading the USER Command 22
A.4. Conclusion 23
Appendix B. Acknowledgements 23

1. Introduction

It is common on the Internet for many DNS names to resolve to a single IP address. This practice has introduced the concept of a "virtual host", where a host appears to exist as an independent entity but, in reality, shares its physical resources with one or more similar hosts.

Such an arrangement presents some problems for FTP servers, because an FTP server distinguishes incoming FTP connections by IP addresses rather than DNS names. Therefore, all DNS names that share a common IP address are handled by the same FTP server and share the same Network Virtual File System (NVFS).

This means that different virtual hosts cannot offer different virtual file systems to clients, nor can they offer different authentication systems. Any scheme to overcome this issue needs to indicate not only the destination IP address but also the virtual hostname that is associated with the desired virtual FTP server. Typical user-FTP processes currently use hostnames to perform hostname-to-IP-address resolution and then ignore hostnames for the

rest of the FTP session; therefore, any mechanism to overcome this issue would require modifications to the user protocol interpreter (user-PI) and server protocol interpreter (server-PI).

It should be noted that this same problem existed for HTTP/1.0 as defined in [RFC1945] and was resolved in HTTP/1.1 as defined in [RFC2616] through the addition of the Host request header field. The goal of this document is to bring a similar level of feature parity to FTP by introducing a new HOST command that allows user-FTP processes to specify which virtual host to connect to for a server-FTP process that is handling requests for multiple virtual hosts on a single IP address.

2. Document Conventions

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 [RFC2119].

In examples, "C>" and "S>" indicate lines sent by the client and server, respectively.

This document also uses notation defined in [RFC959] and [RFC1123]. In particular, the terms "reply", "user", "NVFS", "NVT", "file", "pathname", "FTP commands", "DTP", "user-FTP process", "user-PI", "user-DTP", "server-FTP process", "server-PI", "server-DTP", "mode", "type", "control connection", "data connection", and "ASCII", are all used here as defined there.

The required syntax is defined using the Augmented BNF defined in [RFC5234]. Some general ABNF definitions are required throughout the document; they will be defined in subsequent sections.

With the increased use of virtualization technologies, there may be several possible definitions for the term "virtual host". This document follows the definition from Section 4.1.14 of [RFC3875], where several virtual hosts share the same IP address, and hostnames are used by the server-FTP process to route user-PI sessions to the appropriate virtual host.

2.1. Basic Tokens

This document imports the core definitions given in Appendix B of [RFC5234]. There, definitions will be found for basic ABNF elements like ALPHA, DIGIT, SP, etc. To that, the following term is added for use in this document.

```
TCHAR = VCHAR / SP / HTAB    ; visible plus white space
```

The VCHAR (from [RFC5234]) and TCHAR rules give basic character types from varying subsets of the ASCII character set for use in various commands and responses.

Note that in ABNF, string literals are case insensitive. That convention is preserved in this document and implies that FTP commands and parameters that are added by this specification have values that can be represented in any case. That is, "HOST" is the same as "host", "Host", "HoSt", etc. Similarly, because domain names are defined to be case insensitive, "ftp.example.com" is the same as "Ftp.Example.Com", "fTp.eXample.cOm", etc.

2.2. Server Replies

Section 4.2 of [RFC959] defines the format and meaning of replies by the server-PI to FTP commands from the user-PI. Those reply conventions are used here without change.

```
error-response = error-code SP *TCHAR CRLF
error-code     = ("4" / "5") 2DIGIT
```

Implementers should note that the ABNF syntax used in this document and other FTP-related documents (but that was not used in [RFC959]) sometimes shows replies using the one-line format. Unless otherwise explicitly stated, multi-line responses are also permitted. Implementers should assume that, unless stated to the contrary, any reply to any FTP command (including QUIT) can be of the multi-line format described in [RFC959].

Throughout this document, replies will be identified by the three-digit code that is their first element. Thus, the term "500 reply" means a reply from the server-PI using the three-digit code "500".

3. The HOST Command

A new command, "HOST", is added to the FTP command set in order to allow a server-FTP process to determine to which of possibly many virtual hosts the client wishes to connect. If a HOST command is sent, it MUST be issued before the user is authenticated, as this will allow the authentication scheme and set of authorized users to be dependent upon the virtual host that is chosen.

Server-FTP processes MUST treat a situation in which the HOST command is issued more than once before the user has been authenticated as though only the last HOST command had been sent, and return the appropriate reply for the last HOST command. Server-FTP processes

MUST treat a situation in which the HOST command is issued after the user has been authenticated as an erroneous sequence of commands and return a 503 reply.

Servers should note that the response to the HOST command is a sensible time to send their "welcome" message. This allows the message to be personalized for any virtual hosts that are supported. It also allows the client to determine, via the FEAT response, the languages or representations supported by the server and select an appropriate one via the LANG command. See [RFC2640] for more information.

It should be noted that user-PI implementations that were created before the introduction of the HOST command will not support this new command. A similar problem existed with the introduction of the Host header for HTTP in [RFC2616], and HTTP server implementations had to determine how best to accommodate HTTP requests from down-level clients that did not support the Host header. With this in mind, server-FTP processes will need to determine how best to accommodate FTP requests from down-level FTP clients that do not support the HOST command, but those considerations are outside the scope of this document.

3.1. Syntax of the HOST Command

The HOST command is defined as follows. Note that [RFC3986] remains the normative specification for the syntactic form of IPv4 and IPv6 address literals, in order to ensure identical presentation in 'ftp' URI hostname parts and in the protocol element specified here.

```
host-command  = "HOST" SP hostname CRLF
hostname      = domain / IP-literal

domain        = sub-domain *("." sub-domain)
sub-domain    = let-dig [ldh-str]
let-dig       = ALPHA / DIGIT
ldh-str       = *( ALPHA / DIGIT / "-" ) let-dig

IP-literal    = ( "[" IPv6address "]" ) / IPv4address

IPv6address   = <see [RFC3986] Section 3.2.2>
IPv4address   = <see [RFC3986] Section 3.2.2>

host-response = host-ok / error-response
host-ok       = "220" [ SP *TCHAR ] CRLF
```

The "hostname" rule is a restricted form of the "host" rule specified in [RFC3986]. Details of the additional restrictions imposed by this document are given in the discussion of the syntax that occurs later in this section; they aim at simplifying implementations by only allowing what currently is specified precisely and in use on the Internet.

As with all FTP commands, the "HOST" command word is case independent and can be specified in any character case desired.

The "hostname" (given as a parameter) specifies the virtual host to which access is desired. This SHOULD be the same hostname that was used to obtain the IP address to which the FTP control connection was made, after any client conversions have been completed that convert an abbreviated or local alias to a complete (fully qualified) domain name, but before resolving a DNS alias (owner of a CNAME resource record) to its canonical name.

Internationalization of domain names is only supported through the use of Internationalized Domain Names for Applications (IDNA) "A-labels" for <sub-domain> as described in [RFC5890]. For example, the following HOST command specifies an internationalized domain name:

```
HOST xn--e1afmkfd.com
```

If the user was given an IPv4 or IPv6 literal address, and consequently was not required to derive the literal address from a hostname, the client MAY send the HOST command with the IPv4 or IPv6 literal address as specified to it. While it may seem counterintuitive to specify a literal address by using the HOST command after the client has already connected to the server using a literal address, this should be expected behavior because a user-FTP process should not be required to differentiate between a fully qualified domain name and an IPv4 or IPv6 network literal address. That being said, if the IPv4 or IPv6 literal address specified by the client does not match the literal address for the server, the server MUST respond with a 504 reply to indicate that the IPv4 or IPv6 literal address is not valid.

When the hostname parameter contains a literal address, square brackets are expected to disambiguate IPv6 address syntax from port numbers syntax. Therefore, if the literal address is an IPv6 address, the IPv6 address is required to be enclosed in square brackets (after eliminating any syntax that might also -- but is not required to -- be enclosed in brackets, and from which the server deduced that a literal address had been specified). For example, the

following examples MAY be sent if the client had been instructed to connect to "192.0.2.1", "2001:db8::c000:201", or "::192.0.2.1", respectively, and IPv6 syntax is preferred:

```
HOST 192.0.2.1
HOST [2001:db8::c000:201]
HOST [::192.0.2.1]
```

The client MUST NOT send the port number as part of the HOST command, even when the client has been instructed to connect to a non-standard port. The reason for this requirement is that the user-PI will have established a connection to the server-PI before the HOST command is sent; therefore, specifying a different port with the HOST command has no meaning. For example, the server-PI MUST respond with a 501 reply if the client sends a HOST command with syntax like either of the following examples:

```
HOST 192.0.2.1:2112
HOST [2001:db8::c000:201]:2112
```

The hostname parameter is otherwise to be treated as a fully qualified domain name or relative name as those terms are defined in Section 3.1 of [RFC1034]. This implies that the name is to be treated as a case-independent string, meaning that uppercase ASCII characters are to be treated as equivalent to their corresponding lowercase ASCII characters but otherwise preserved as given. It also implies some limits on the length of the parameter and of the components that create its internal structure. Those limits are not altered in any way here.

Neither [RFC1034] nor [RFC1035] imposes any other restrictions upon what kinds of names can be stored in the DNS. This specification, however, only allows the use of names that can be inferred from the ABNF grammar given for the "hostname". Similarly, this specification restricts address literals to the IPv4 and IPv6 address families well established on the Internet.

3.2. HOST Command Semantics

Upon receiving the HOST command, before authenticating the user-PI, a server-FTP process SHOULD validate that the hostname given represents a valid virtual host for that server and, if it is valid, establish the appropriate environment for that virtual host. The resultant actions needed to create that environment are not specified here and may range from doing nothing at all to performing a simple change of working directory, changing authentication schemes and/or username and password lists, or making much more elaborate state changes -- such as creating isolated environments for each FTP session.

The 220 reply code for the HOST command is the same as the code that is used in the initial "welcome" message that is sent after the connection is established.

If the hostname specified would normally be acceptable, but is temporarily unavailable, the server-FTP process SHOULD respond to the HOST command with a 421 reply and close the connection.

Example:

The server-FTP process is shutting down, so the server-FTP process responds to the HOST command with a 421 reply and closes the connection. In this scenario, the 421 reply informs the client it can retry at another time.

If the hostname specified is unknown at the server, or if the server is otherwise unwilling to treat the particular connection as a connection to the hostname specified, the server SHOULD respond with a 504 reply.

Examples:

The particular virtual host that was specified by the HOST command is disabled at the server. The server responds with a 504 reply and keeps the connection open in order to allow the user-PI an opportunity to specify another virtual host with a subsequent HOST command.

Alternatively, the server-FTP process might choose to route all connections with unknown hostnames to a different virtual host so that no connection attempts will result in failed connections. This design would be implementation specific and outside the scope of this specification.

3.2.1. REIN Command Semantics

As specified in [RFC959], the REIN command returns the state of the connection to what it was immediately after the transport connection was opened. This specification makes no changes to that behavior. The effect of a HOST command MUST be reset if a REIN command is performed, and a new HOST command MUST be issued afterwards in order to connect to a virtual host.

3.2.2. User-PI Usage of HOST

A user-PI MUST send the HOST command after opening the transport connection, or after any REIN command, before attempting to authenticate the user with the USER command. The following example illustrates what a typical login sequence might look like when the HOST command is used:

```
C> HOST ftp.example.com
S> 220 Host accepted
C> USER foo
S> 331 Password required
C> PASS bar
S> 230 User logged in
```

If a user-PI sends an additional HOST command before attempting to authenticate the user, a server-FTP process MUST treat the additional HOST command as though a previous HOST command was not sent and return the appropriate reply for the new HOST command. For example, if a user specifies the wrong virtual hostname by mistake, sending a subsequent HOST command will rectify the error. The following example illustrates what the login sequence might look like when the HOST command is sent twice before a user has been authenticated:

```
C> HOST foo.example.com
S> 220 Host accepted
C> HOST bar.example.com
S> 220 Host accepted
C> USER foo
S> 331 Password required
C> PASS bar
S> 230 User logged in
```

The HOST command can be used in combination with the ACCT command to differentiate between a user's various accounts on a specific virtual host. In this scenario, the user-PI sends a HOST command, which the server-PI uses to route activity to the correct virtual host; the user-PI sends credentials using the USER and PASS commands, which the server-PI validates; then, the user-PI sends an ACCT command to specify any additional account information for the server-PI implementation. The following example illustrates a sequential series of client commands that specify both a HOST and ACCT, with the server responses omitted for brevity:

```
C> HOST ftp.example.com
C> USER foo
C> PASS bar
C> ACCT project1
```

This is also true when the HOST command is used with the AUTH and ADAT commands that are discussed in [RFC2228] and [RFC4217]. In this scenario, the user-PI sends a HOST command, which the server-PI uses to route activity to the correct virtual host; then, the user-PI uses the AUTH and ADAT commands to negotiate the security mechanism and relevant authentication token(s) with the server-PI; then, the user-PI sends user credentials using the USER and PASS commands, which the server-PI validates, after which the user-PI MAY send an ACCT command to specify any additional account information for the server-PI implementation. The following example illustrates a sequential series of client commands that specify both HOST and ACCT commands when used in conjunction with the security commands that are discussed in [RFC2228] and [RFC4217], with the server responses omitted for brevity:

```
C> HOST ftp.example.com
C> AUTH <mechanism-name>
C> ADAT <base64data>
C> USER foo
C> PASS bar
C> ACCT project1
```

An exception to the above scenario would be when a user-PI is providing the hostname in the "server_name" extension of a Transport Layer Security (TLS) extended client hello as discussed in [RFC6066]. When the user-PI specifies the hostname in the "server_name" extension of a TLS extended client hello, the server-PI MUST verify that the hostname in the HOST command matches the value of the "server_name" extension. The following example illustrates a sequential series of client commands that specify the HOST command when used in conjunction with the TLS extensions that are discussed in [RFC6066], with the server responses omitted for brevity:

```
C> AUTH TLS
C> HOST ftp.example.com
C> USER foo
C> PASS bar
```

Additional security information about using the HOST command with the security extensions that are discussed in [RFC2228], [RFC4217], and [RFC6066] is provided in Section 4 of this document.

3.2.3. State Diagrams

The state diagrams in this section illustrate typical sequences for command and reply interchange between the user-PI and server-PI. These diagrams are modeled on the similar diagrams in Section 6 of [RFC959].

In each diagram, the (B) "begin" state is assumed to occur after the transport connection has opened or after a REIN command has succeeded. Other commands (such as FEAT [RFC2389]) that require no authentication may have intervened.

Additionally, a three-digit reply indicates a precise server reply code. A single digit on a reply path indicates any server reply that begins with that digit, except where a precise server reply code is defined on another path. For example, a single digit "5" will apply to "500", "501", "502", etc., when those reply codes are not expressly defined in the diagram. For each command, there are three possible outcomes: success (S), failure (F), or error (E). In the state diagrams below, we use the symbol "B" for "begin" and the symbol "W" for "wait for reply".

For each of these diagrams, without any state transitions being shown, a REIN command will return the diagram from any wait state to the (B) "begin" state.

The state diagram in Figure 1 shows a typical sequence of flow of control when HOST is used with USER and PASS to log in to a particular FTP virtual host.

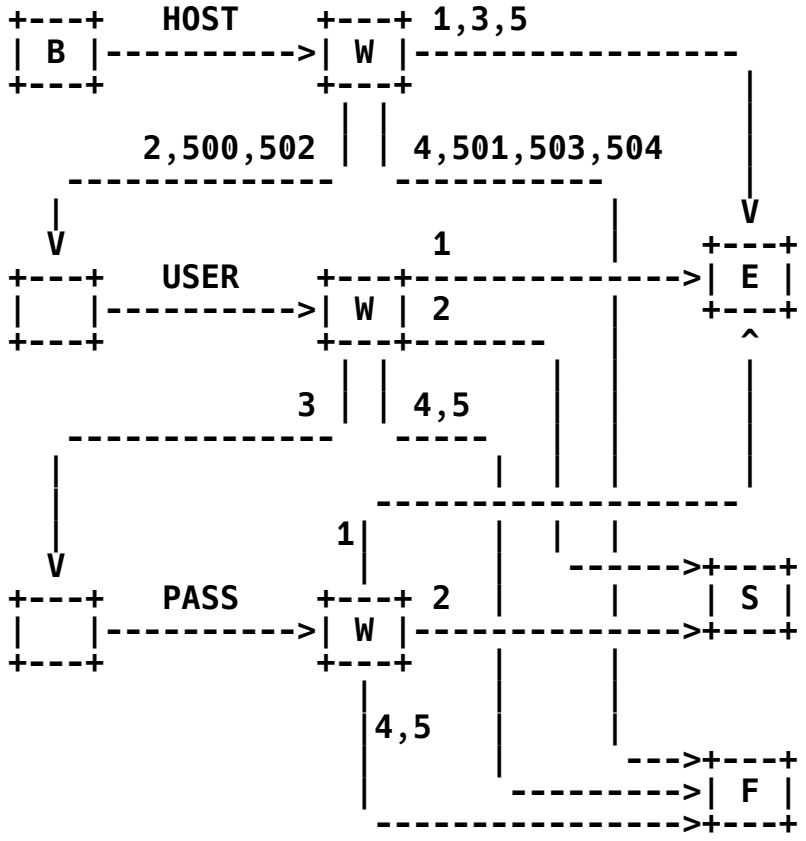


Figure 1: Typical Login Sequence with HOST Command

After a user has logged in, an additional account may be required by the server and specified by the client by using the ACCT command. With this in mind, the state diagram in Figure 2 shows a typical sequence of flow of control when HOST is used with USER and PASS to log in to an FTP virtual host and ACCT is used to specify an account.

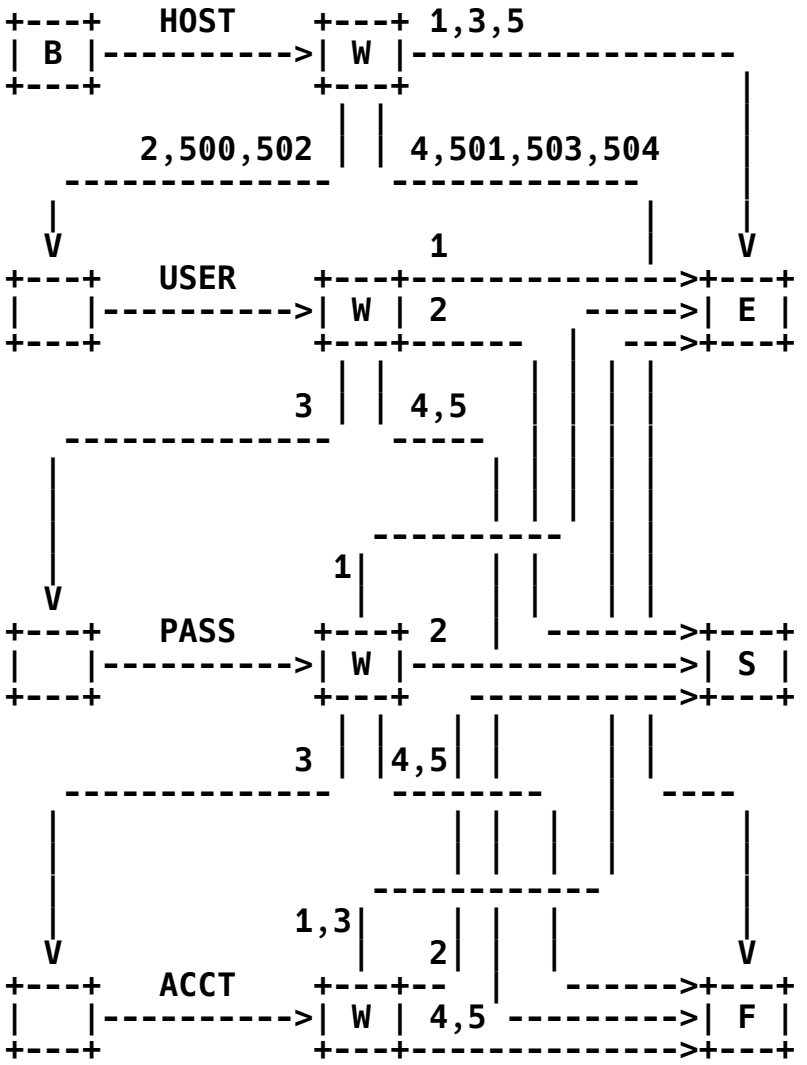


Figure 2: Login Sequence with HOST and ACCT Commands

The state diagram in Figure 3 shows a typical sequence of flow of control when HOST is used with the AUTH and ADAT commands that are discussed in [RFC2228]. (NOTE: Section 4 provides additional information about using the HOST command with TLS.)

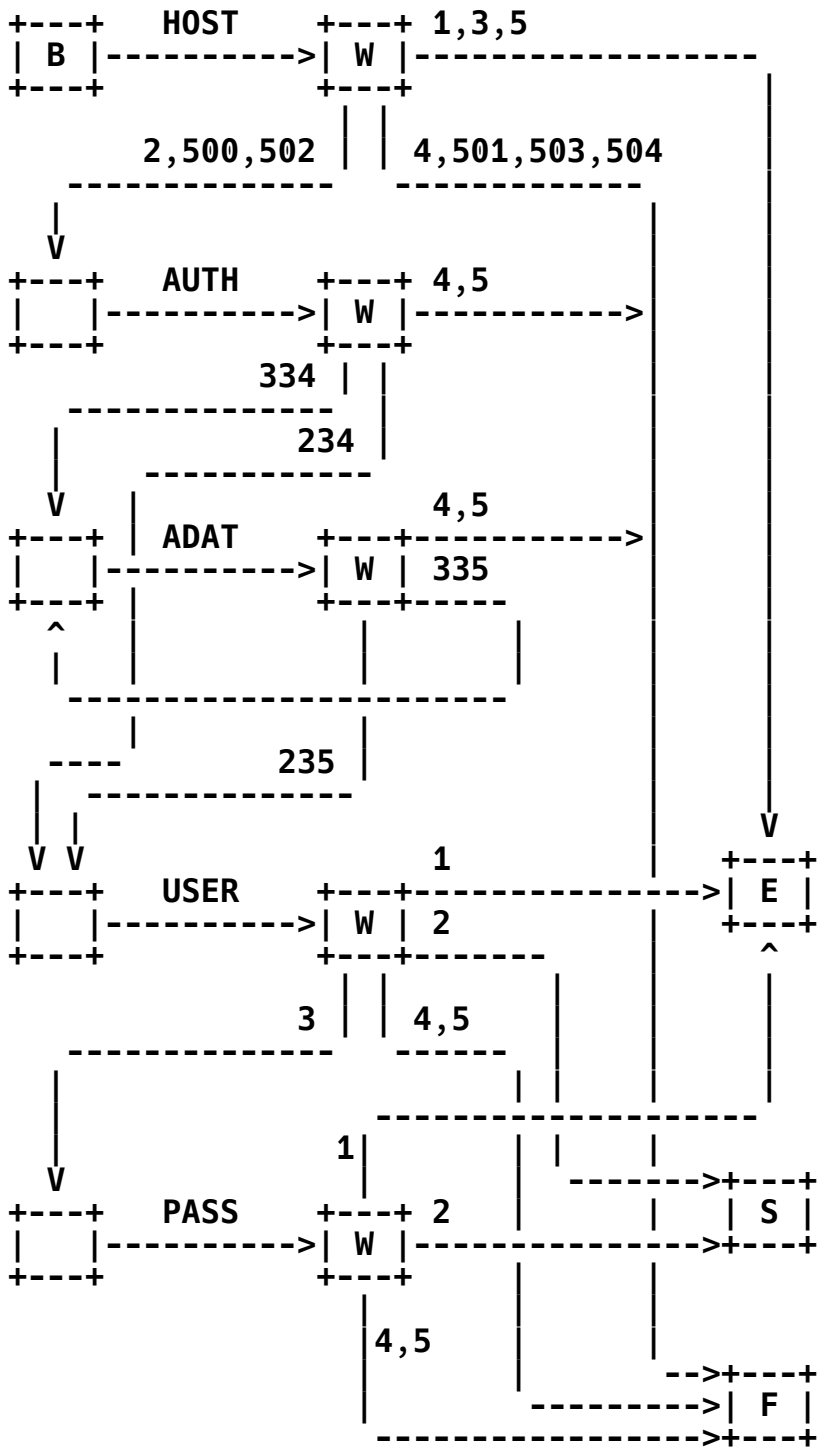
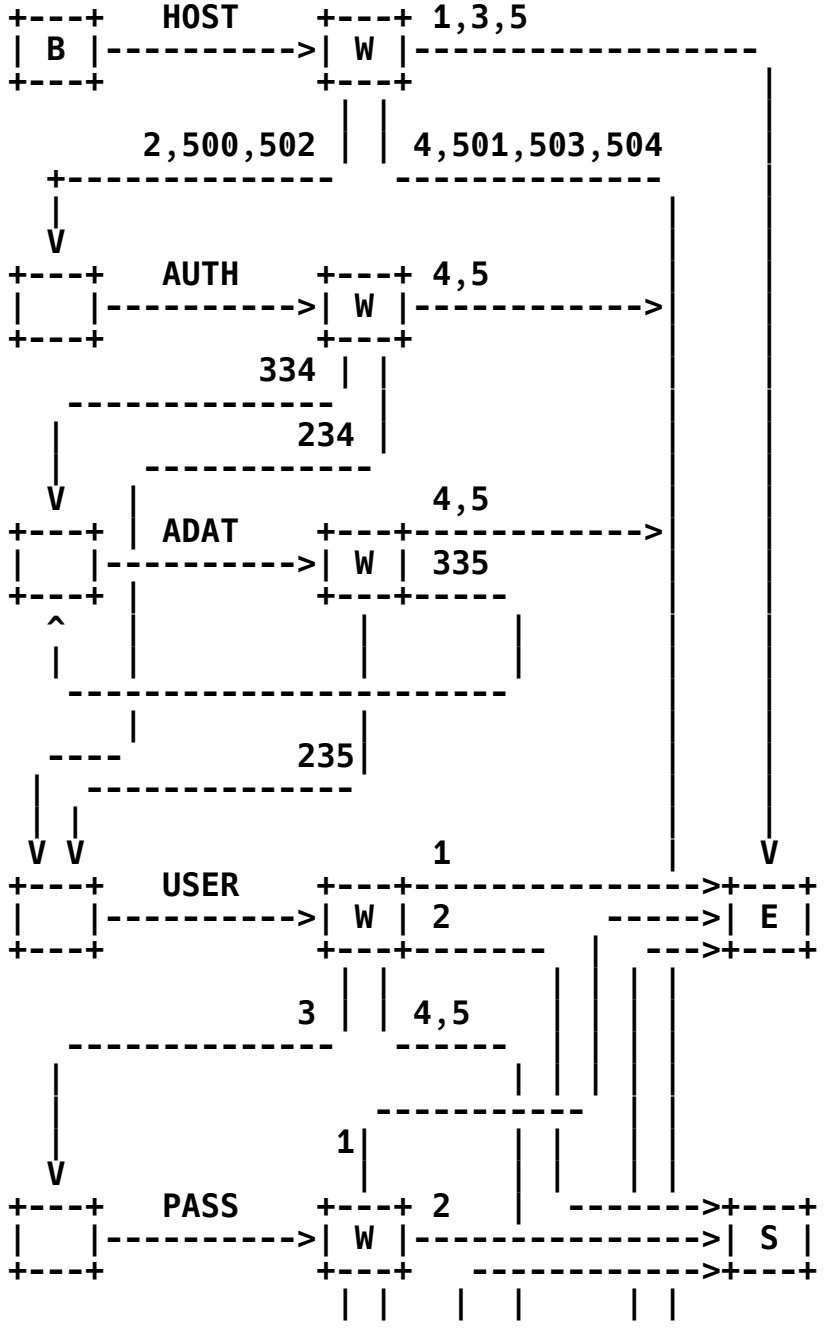


Figure 3: Login Sequence with HOST and AUTH/ADAT Commands

After a user has logged in with the security commands that are discussed in [RFC2228], an additional account may be required by the server and specified by the client by using the ACCT command. The state diagram in Figure 4 shows a typical sequence of flow of control when HOST is used with the AUTH and ADAT commands to log in to an FTP virtual host and ACCT is used to specify an account.



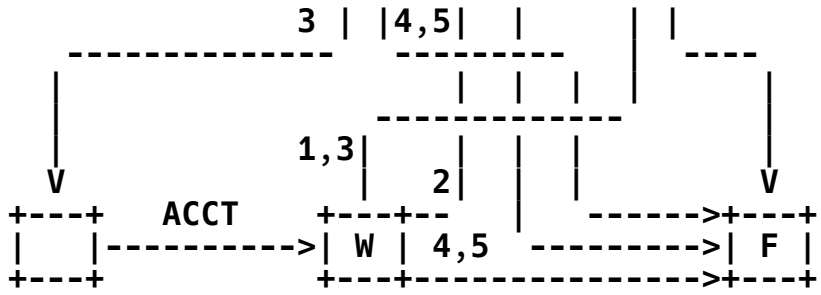


Figure 4: Login Sequence with HOST and AUTH/ADAT/ACCT Commands

3.3. HOST Command Errors

The server-PI SHOULD return a 500 or 502 reply if the HOST command is unrecognized or unimplemented, as specified in [RFC959]. For example, a server-PI that predates or otherwise does not conform to this specification would be expected to return a 500 or 502 reply.

As discussed in Section 3 of this document, if a HOST command is sent after a user has been authenticated, the server MUST treat the situation as an invalid sequence of commands and return a 503 reply.

A 501 reply SHOULD be sent if the hostname given is syntactically invalid, and a 504 reply SHOULD be sent if a syntactically valid hostname is not a valid virtual hostname for the server. In all such cases, the server-FTP process MUST do one of the following:

a. Ignore the HOST command and act as if a HOST command had not been sent. A user-FTP process MAY then send a subsequent HOST command with a different hostname.

b. Close the connection.

A user-PI receiving a 500 or 502 reply to a HOST command SHOULD assume that the server-PI does not implement virtual servers by using the HOST command. The user-PI MAY then proceed to log in as if the HOST command had not been sent.

A user-PI receiving an error reply that is different from the errors that have been described here SHOULD assume that the virtual HOST is unavailable and terminate communications.

A server-PI that receives a USER command to begin the authentication sequence without having received a HOST command SHOULD NOT reject the USER command. Clients that conform to earlier FTP specifications do not send HOST commands. In this case, the server MAY act as if some default virtual host had been explicitly selected, or the server MAY

enter an environment that is different from that of any supported virtual hosts, perhaps one in which a union of all available accounts exists and that presents an NVFS that appears to contain subdirectories that contain the NVFS for all supported virtual hosts.

3.4. FEAT Response for HOST Command

When replying to the FEAT command [RFC2389], a server-FTP process that supports the HOST command MUST include a line containing the single word "HOST". This word is case insensitive, but it SHOULD be sent in upper case so as to maximize interoperability with disparate implementations. That is, the response SHOULD be:

```
C> FEAT
S> 211- <any descriptive text>
S>  ...
S>  HOST
S>  ...
S> 211 End
```

The ellipses indicate placeholders where other features may be included but are not required. The one-space indentation of the feature lines is mandatory [RFC2389].

4. Security Considerations

As discussed in Section 3 of this document, a server implementation MUST treat an additional HOST command that was sent before a user has been authenticated as though a previous HOST command was not sent. In this situation, the server implementation MUST reset the authentication environment, as that would allow for segregation between the security environments for each virtual host on an FTP server. The implementation details for security environments may vary greatly based on the requirements of each server implementation and operating system, and those details are outside the scope of the protocol itself. For example, a virtual host "foo.example.com" on an FTP server might use a specific username and password list, while the virtual host "bar.example.com" on the same FTP server might use a different username and password list. In such a scenario, resetting the security environment is necessary for the virtual servers to appear to behave independently from a client perspective, while the actual server implementation details are irrelevant at the protocol level.

Section 15.1.1 of [RFC4217] discusses the use of X.509 certificates for server authentication. Taking the information from that document into account, when securing FTP sessions with the security mechanisms that are defined in [RFC4217], client implementations SHOULD verify

that the hostname that they specify in the parameter for the HOST command matches the identity that is specified in the server's X.509 certificate in order to prevent man-in-the-middle attacks.

When the HOST command is used in combination with the FTP security extensions that were introduced in [RFC2228] and [RFC4217], the HOST command SHOULD precede the security handshake when the user-PI is not providing the "server_name" in the extended client hello as defined in [RFC6066]. This allows both user-FTP and server-FTP processes to map an FTP HOST with the correct server name in the server's certificate. If the HOST command is sent after the security handshake, then mapping an FTP HOST to the correct security certificate will not take place before the secure session is established.

For example, if a server-FTP process has multiple virtual hosts defined and no hostname has been sent from a user-FTP process, the server-FTP process will be unable to route the connection to the correct virtual host when the connection is established. In this situation, the server-FTP process will be forced to choose a virtual host that will respond. When the user-PI attempts to negotiate a secure connection, the virtual host to which the connection was routed will respond with its server certificate during the security handshake. If the virtual host that was chosen by the server-FTP process does not match the virtual host to which the user-FTP process had intended to connect, the user-PI will be unable to verify the server's identity as presented in the server certificate message.

However, if the user-PI is providing the "server_name" in the extended client hello as defined in Section 3 of [RFC6066], the user-PI MAY provide the HOST command after the security handshake because the server will be able to route the connection to the correct virtual host based on the contents of the "server_name" extension and the client will be able to verify the server's identity as presented in the corresponding server certificate message. However, the server-PI MUST verify that the name in the HOST command matches the "server_name" that is provided in the extended client hello.

In general, client implementations SHOULD protect user credentials by using the FTP security extensions that were introduced in [RFC2228] and [RFC4217]; a detailed discussion for securing FTP sessions can be found in those documents, and a general discussion of security issues related to FTP can be found in [RFC2577].

5. IANA Considerations

IANA has registered the following FTP extension according to the procedure established by [RFC5797]:

cmd	FEAT Code	description	type	conf	RFC#s/References and Notes
HOST	HOST	Hostname	a	o	RFC 7151

6. References

6.1. Normative References

[RFC959] Postel, J. and J. Reynolds, "File Transfer Protocol (FTP)", STD 9, RFC 959, October 1985.

[RFC1034] Mockapetris, P., "Domain Names - Concepts and Facilities", STD 13, RFC 1034, November 1987.

[RFC1035] Mockapetris, P., "Domain Names - Implementation and Specification", STD 13, RFC 1035, November 1987.

[RFC1123] Braden, R., "Requirements for Internet Hosts - Application and Support", STD 3, RFC 1123, October 1989.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2228] Horowitz, M. and S. Lunt, "FTP Security Extensions", RFC 2228, October 1997.

[RFC2389] Hethmon, P. and R. Elz, "Feature negotiation mechanism for the File Transfer Protocol", RFC 2389, August 1998.

[RFC2640] Curtin, B., "Internationalization of the File Transfer Protocol", RFC 2640, July 1999.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC4217] Ford-Hutchinson, P., "Securing FTP with TLS", RFC 4217, October 2005.

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[RFC5890] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, August 2010.

[RFC6066] Eastlake, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, January 2011.

6.2. Informative References

[RFC1945] Berners-Lee, T., Fielding, R., and H. Nielsen, "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996.

[RFC2577] Allman, M. and S. Ostermann, "FTP Security Considerations", RFC 2577, May 1999.

[RFC2616] 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.

[RFC3875] Robinson, D. and K. Coar, "The Common Gateway Interface (CGI) Version 1.1", RFC 3875, October 2004.

[RFC5797] Klensin, J. and A. Hoenes, "FTP Command and Extension Registry", RFC 5797, March 2010.

Appendix A. Unworkable Alternatives

Due to the level of scope for adding a new command to FTP, a brief discussion of suggested alternatives to a HOST command and their respective limitations is warranted. The suggested alternatives that are discussed in this appendix have been proposed in the past, but each of these ideas was deemed insufficient for the reasons listed within each section of this appendix.

A.1. Overloading the CWD Command

One suggested method to emulate a form of virtual hosts would be for the client to simply send a CWD command after connecting, using the virtual hostname as the argument to the CWD command. This would allow the server-FTP process to implement the file stores of the virtual hosts as subdirectories in its NVFS. This suggestion is simple in concept, and most server-FTP implementations support this without requiring any code changes. While this method is simple to describe and implement, it suffers from several drawbacks:

a. The CWD command is available only after the user-PI has authenticated itself to the server-FTP process. Thus, all virtual hosts would be required to share a common authentication scheme if they used this method.

b. To make the virtual host truly transparent, either the server-FTP process needs to be modified to include information that shows the special nature of this first CWD command (negating most of the advantage of this scheme), or all users must see the same identical NVFS view upon connecting (they must connect in the same initial directory), or the NVFS must implement the full set of virtual host directories at each possible initial directory for any possible user.

c. Unless the server is specially modified, a user connecting this way to a virtual host would be able to easily move to any other virtual host supported at the same server-FTP process, exposing the nature of the virtual host.

A.2. Overloading the ACCT Command

Another suggested method would be to simply overload the ACCT command for FTP virtual hosts, but this proposal is unacceptable for several reasons with regard to when the ACCT command is sent during the request flow. Sections 5.4 and 6 of [RFC959] document the request flow for a login sequence as USER -> PASS -> ACCT. This flow of commands may be acceptable when you are considering a single user

having multiple accounts on an FTP server, but it fails to differentiate between virtual hosts when you consider the following two issues:

a. The first problem with overloading the ACCT command is certificate negotiation when using the FTP security extensions that are documented in [RFC2228] and [RFC4217]. In order to safeguard user credentials, negotiation of the security mechanism and certificate must occur before login credentials are sent by the client. The problem with using the ACCT command in this scenario is that there is no way of ensuring that the certificate matches the correct virtual host before the user credentials are sent.

b. The second problem with overloading the ACCT command is how user credentials are implemented for FTP virtual hosts. FTP server implementations may allow the use of custom user credentials on a per-virtual-host basis. For example, in one particular implementation the virtual host negotiation occurs, and then the user credentials are looked up using the account mechanism that is specific to that virtual host. So once again the virtual host negotiation must take place before the user credentials are sent.

A.3. Overloading the USER Command

An additional suggestion would be to overload well-known syntax through the existing USER command, as illustrated in the following example:

```
C> USER foo@example.com
S> 331 Password required
C> PASS bar
S> 230 User logged in
```

In this example, the user "foo" might be attempting to log on to the virtual host "example.com" on an FTP server. This suggestion may seem plausible at first, but it introduces several implementation problems. For example:

a. Some network environments already use the "username@hostname" syntax for network credentials, where the "hostname" portion refers to the location of the user's credentials within the network hierarchy. Using the "foo@example.com" syntax, it becomes difficult to differentiate between the user "foo" logging into a virtual host that is named "example.com" on an FTP server versus the user "foo@example.com" logging into an FTP server with no specified virtual host.

b. When using the FTP security extensions that are documented in [RFC2228] and [RFC4217], negotiation of the security mechanism and certificate must occur before login credentials are sent by the client. More specifically, the AUTH/ADAT commands must be sent before the USER command in order to safeguard user credentials. If you overload the USER command, there is no way of ensuring that the certificate matches the correct virtual host before the user credentials are sent by the client.

A.4. Conclusion

After examining the above alternatives, and in order to obtain an adequate emulation of "real" FTP servers, it was concluded that supporting virtual hosts will require both client and server modifications. Therefore, a new FTP command seems the most likely solution to provide the required level of support.

Appendix B. Acknowledgements

Robert Elz and Paul Hethmon provided a detailed discussion of the HOST command in their Internet-Draft titled "Extensions to FTP" as part of their work with the FTPEXT Working Group of the IETF. Their work formed the basis for much of this document, and their help has been greatly appreciated. They would also like to credit Bernhard Rosenkraenzer for having first suggested and described the HOST command.

Several people have provided a wealth of constructive feedback about earlier versions of this document that has helped to shape its development; many of their suggestions have been incorporated, and their contributions are gratefully acknowledged. There are far too many to mention here, but the authors of this document would like to specifically thank Alexey Melnikov, Alfred Hoenes, John Klensin, Joe Touch, Paul Ford-Hutchinson, Daniel Stenberg, Mykyta Yevstifeyev, Alec Rowell, Jaroslav Dunajsky, Wade Hilmo, Anthony Bryan, and Barry Leiba for their assistance.

Authors' Addresses

Paul Hethmon
Hethmon Brothers
2305 Chukar Road
Knoxville, TN 37923
USA

EMail: phethmon@hethmon.com

Robert McMurray
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
USA

EMail: robmcm@microsoft.com