
Sendmail and Beyond: Kewl Tips and Tricks By Ankit Fadia ankit@bol.net.in

---

Welcome to yet another Hacking Truths manual. Although this manual comes after a long break, it is really nice to get back to writing for HT. Anyway, in the past, we have had a number of explanations on how to send forged emails, how to play with the Sendmail daemon, email headers and everything else to do with SMTP (Simple Mail Transfer Protocol) and emails. Although this manual too throws light on related matter, it is however more focused on advanced tips and tricks and other uncommon but extremely useful pieces of information.

Now, we have already learnt how one can, telnet to Port 25 of a mail server and send an email (even a forged email) by simply typing out some SMTP commands. However, for the benefit of beginners and to refresh the memory of experienced but forgetful people, we would quickly be going through the process. I promise to make it as short as possible, at the same time easy to understand.

Port 25 is the Sendmail Port where the SMTP daemon runs. This daemon is infact the daemon handling all the outgoing mails. All email clients send mail by connecting to Port 25 of the mail server and issuing SMTP commands. This process is automated and occurs in the background. However, one could also manually connect (telnet) to Port 25 of a mail server and manually type out the SMTP commands in order to send emails. So the basic outline of the entire process as below. For details regarding the usage of individual commands, simply type the word ‘help’ followed by the command at the Sendmail prompt.

Note: The below sends a mail from ankit@bol.net.in to namitas@bol.net.in by issuing SMTP commands to the mail server: mail.isp.com Responses from the mail sever have a number preceding them while the commands typed by the user do not have any number preceding.

C:\windows>telnet mail.isp.com

220 mail.isp.com ESMTP Sendmail 8.9.1 (1.1.20.3/07Jul00-0916AM) Thu, 7 Dec 2000 17:18:50 +0530 (IST)

helo ankit.com

250-mail.isp.com Hello [203.xx.yyy.91], pleased to meet you

mail from: ankit@bol.net.in

250 ankit@bol.net.in... Sender ok

rcpt to: namitas@bol.net.in

250 namitas@bol.net.in... Recipient ok

data

354 Enter mail, end with "." on a line by itself

This is the part where the body of the message is typed in.

.

250 RAA0000001693 Message accepted for delivery

The headers of the above email as seen by the recipient is as follows:

Return-Path: <ankit@bol.net.in>

Received: from ankit.com by mail.isp.com (8.9.1/1.1.20.3/07Jul00-0916AM)

id RAA0000001693; Thu, 7 Dec 2000 17:19:49 +0530 (IST)

Date: Thu, 7 Dec 2000 17:19:49 +0530 (IST)

From: Ankit Fadia <ankit@bol.net.in>

Message-Id: <200012071149.RAA0000001693@mail.isp.com>

X-UIDL: 920156a3b926c5193036933e6d04efd5

This is the part where the body of the message is typed in.

Anyway, now that we have recalled the basic outline of the process of manually sending an email, let us move on with the main subject of this manual.

The Subject Field Un-subjected

Now, ever since I released the manual on sending forged emails (Sending emails using SMTP commands) I have received a number of emails asking me questions like: “How to Specify the Subject of an email sent manually by connecting to Port 25 of a system”. Or “How to Specify CC and BCC recipients when doing the same?” Well, in this section we discuss just that.

Firstly, let us learn how to specify the subject of an email engineered manually by SMTP commands. Well, the process of specifying the Subject remains pretty much similar to the normal process of sending emails manually. Actually all the commands remain the same until we reach the ‘data’ command. After we issue the ‘data’ command, the remote mail server will reply with the below message:

354 Enter mail, end with "." on a line by itself

This server response means that we can start typing the body of our message now. However, it also indirectly specifies that this is the time that we type in the Subject of the email. We can specify the subject of the email as follows:

Subject: Hi

Where ‘subject:’ is the keyword, which tells the mail server that you are ready to type in your subject and ‘Hi’, is the subject of your choice. You can continue with the body of the email by pressing the ‘Enter’ key and typing in the characters. The end with the: ‘ . ‘(Period) and everything else remains the same.

Let us go though the entire process, step by step. Please note that I have inserted comments wherever necessary within brackets. Both the brackets and the characters within the brackets are not a part of the actual commands.

For this example, we need to keep the following pieces of information in mind:

Mail Server: mail.isp.com

Recipient’s Email Address: namitas@bol.net.in

Sender’s Email Address: ankit@bol.net.in

Subject: Hi!!!

Body: This is a test message

C:\windows>telnet mail.isp.com

220 mail.isp.com ESMTP Sendmail 8.9.1 (1.1.20.3/07Jul00-0916AM) Thu, 7 Dec 2000 17:18:50 +0530 (IST)

helo ankit.com

250-mail.isp.com Hello [203.xx.yyy.91], pleased to meet you

mail from: ankit@bol.net.in

250 ankit@bol.net.in... Sender ok

rcpt to: namitas@bol.net.in

250 namitas@bol.net.in... Recipient ok

data

354 Enter mail, end with "." on a line by itself

subject: Hi!!!

This is a test message

.

250 RAA0000001693 Message accepted for delivery

Now if you examine the headers of this email, you will find that they unlike the headers that we viewed earlier in the manual will have a separate Subject line.

Return-Path: <ankit@bol.net.in>

Received: from ankit.com by mail.isp.com (8.9.1/1.1.20.3/07Jul00-0916AM)

id RAA0000001693; Thu, 7 Dec 2000 17:19:49 +0530 (IST)

Date: Thu, 7 Dec 2000 17:19:49 +0530 (IST)

From: Ankit Fadia <ankit@bol.net.in>

Message-Id: <200012071149.RAA0000001693@mail.isp.com>

Subject: Hi!!!!

X-UIDL: 920156a3b926c5193036933e6d04efd5

This is a test message

CC's and BCC's

What are the SMTP commands equivalent to the BCC and CC fields of your email client? Well, this question has only one simply answer: none. The following few lines will tell us why.

To understand the answer to the above question, let us first understand how exactly does an email client handle a CC or a BCC. How does it do what we are supposed to do with the CC and BCC features?

Now, when you hit the Send button, then your email client connects to Port 25 of the mail server that you specified during the configuration time. Then it will issue SMTP commands to the remote mail server and send it the required information. And in this process your email is sent. The order in which the various SMTP commands are given is same as described earlier.

Normally, when you have only a single recipient, then your email client issues only a single 'RCPT TO:' command, to the mail server. However, when there is more than a single recipient, then the email client issues multiple instances of 'RCPT TO:' Or in other words, when the CC field of your email client is not empty then multiple RCPT commands are issued.

You see the Simple Mail Transfer Protocol does not provide any special command for CC'ing an email to someone. The entire concept of CC relies on the issue of multiple RCPT commands to the mail server. The same is the case when you have multiple recipients in the 'To:' field of the email client. So basically this means that it really doesn't matter whether you add a recipient's email address to the CC field or to the 'To:' field. The SMTP command issued and the headers created will remain the same.

Let us take a practical example to make it clearer. The recipients' list for this example is as follows:

To: ankit@bol.net.in; ankitfadia@hotmail.com

CC: ankit_Fadia@hotmail.com ; namitas@bol.net.in

In this case, the following are the commands, which will send a blank email with the subject ‘test’ from the email address: test@bol.net.in to the above list of recipients.

C:\windows>telnet mail.isp.com

220 mail.isp.com ESMTP Sendmail 8.9.1 (1.1.20.3/07Jul00-0916AM) Thu, 7 Dec 2000 17:18:50 +0530 (IST)

helo ankit.com

250-mail.isp.com Hello [203.xx.yyy.91], pleased to meet you

mail from: test@bol.net.in

250 test@bol.net.in... Sender ok

rcpt to: ankit@bol.net.in

250 ankit@bol.net.in... Recipient ok

rcpt to: ankitfadia@hotmail.com

250 ankitfadia@hotmail.com... Recipient ok

rcpt to: ankit_fadia@hotmail.com

250 ankit_Fadia@hotmail.com... Recipient ok

rcpt to: namitas@bol.net.in

250 namitas@bol.net.in... Recipient ok

data

354 Enter mail, end with "." on a line by itself

subject: Test

.

250 RAA0000001693 Message accepted for delivery

Get it? Now, let us move on to as to how BCC works.

Now, in the above case i.e. in the case of CC, the email client used multiple RCPT's in the same SMTP session to send the same email to multiple recipients. However, in such a case the email any recipient can view the email addresses of all the recipients. The reason behind this privacy invasion is the fact that a single email sent to either a single or multiple recipients has to have the same exact email headers. This means that all recipients in the 'CC' and 'To' fields of the same email have to have the same email headers. This is due to the fact that the email addresses of all the recipients were given to the mail server during the same SMTP session. All this may sound quite vague and weird. If that is the case, then read the following paragraphs to understand better.

Now, when you CC a single email to multiple recipients (Say 3) then the following procedure takes place:

Email Client Starts Session at remote mail server.

It introduces itself and the sender.

It uses multiple RCPT commands to send the same email to multiple recipients.

The email client disconnects.

As the email addresses of all the recipients are mentioned in the same session at the remote mail server, they constitute the same email headers. Thus all the recipients are able to view the email addresses to which this email was sent.

Now, in a situation, when we BCC the same email to multiple recipients (Say 2) then the following procedure takes place:

Email Client Starts Session at remote mail server.

It introduces itself and the sender.

It uses a single RCPT commands to send the same email to the first email address in the BCC list.

The email client disconnects.

It again starts a new session at the remote server.

It again introduces itself and the sender.

It uses a single RCPT commands to send the same email to the second email address in the BCC list.

The email client disconnects, once again.

In this case, each recipient was sent an email through a unique session at the remote mail server, thus each recipient received unique email headers and the identity of none of the other recipients in the BCC list was not given away.

The above description of the usage of CC and BCC is based on how Outlook Express works. However, actually Sendmail does provide a manner in which the CC recipients can be specified. After giving the DATA command, one can give the CC list by giving the following command:

CC:Recipient List

However, giving the BCC command instead of CC does not produce the desired result.

Sending Attachments through Sendmail

Today, MIME attachments are used to transfer files attached to an email. MIME attachments use Base64 encoding to encode the binary data. Earlier another encoding standard was used, which was called the Uuencode encoding standard. You can send attachments through Sendmail using any of the above methods.

UU-encoding or Unix-to-Unix encoding is an encoding standard, which converts all kinds of files into ASCII for safe transmission over Networks. Files, which are to be sent over networks, are encoded at the sender's end and decoded at the receiver's end. This ensures that files (attachments) can be transferred over different kinds of networks, systems routers etc without any loss. However, this method turned out be corruption prone and is thus not the most preferred one.

According to a University, the basic mechanism of UU-encoding is as follows:

The basic scheme is to break groups of 3 eight-bit characters (24 bits) into 4 six-bit characters and then add 32 (a space) to each six-bit character, which maps it into the readily transmittable character. Another way of phrasing this is to say that the encoded 6 bit characters are mapped into the set: `!"#$%&'()*+,-
./012356789:;<=>?@ABC...XYZ[\]^_ for transmission over communications lines.

Such encoding increases the file size by about 42%. So, the mechanism of UU-encoding can be concluded as follows:

File is Uuencoded at sender's end ----------à File is Uudecoded at the receiver's end.

All attachments too can be sent over networks in uuencoded form.

You see if you enter the uuencoded code of any file after you have issued the DATA command at the Sendmail prompt, then the recipient will be able to receive the attachment and view it too. Almost all email clients allow Uudecoding. (Even if the email client used by the recipient does not allow Uudecoding then are several utilities, which do

it for you.) All files including images, audio files, video files, text files etc can be encoded by the Uuencoding standard to obtain the uuencoded code.

The method by which attachments in the form of their uuencoded form can be sent as attachments is a 2-step process-:

Converting the file to be sent as an attachment into uuencoded form.

Given the uuencoded form to the mail server after the DATA command.

Let, us first tackle the first step:

If you are using a Windows platform, then all you need to perform Uuencoding is WinZip. If you do not already have WinZip, then you could get it from: http://www.winzip.com

WinZip can easily be used for obtaining the Uuencode of any file. Simply create a new archive containing the file you want to Uuencode and select Action > Uuencode. You could also simply press Shift + U.

WinZip will save the Uuencode form of the .zip file in the form: filename.uue

A typical .uue file (In this case of an image file) would be as follows:

_=_

_=_ Part 001 of 001 of file new.zip

_=_

begin 666 new.zip

M4$L#!!0``@`(`#5S_RCDJL7+;P```'4````'````;F5W+F=I9G/W=+.P3)1G

MX&%8R``"_T$`Q%#\R<+(P,#(H`/B@.0=F-QZ\INZ%.\\$DX(:]"N_76TM7"V

M:6]\T+)755;)-P(C;UB]*)FR+OSYCGV';_HI7<P)::DQ$Y_Y[%*(UX1`H4U;

M3Z55KVB;<EV#@<$:`%!+`0(4`!0``@`(`#5S_RCDJL7+;P```'4````'````

K`````````(`"V@0````!N97<N9VEF4$L%!@`````!``$`-0```)0`````````

`

end

The first few lines are only comments added by WinZip and are not actually a part of the Uuencoded code. So, simply eliminate everything above the following line:

begin 666 new.zip

This gives you the Uuencode code of the file you want to transmit as an attachment using Sendmail.

****

HACKING TRUTH: If you are on a Unix platform then getting the Uuencode of a file becomes extremely easy. Simply go to the Unix shell so you can use uuencode on the file you're trying to send. For purposes of this example, let's presume the file you're trying to send is called "myfile.doc".

At the Unix shell prompt, type the command:

uuencode myfile.doc myfile.doc > tempfile.uu

This tells the uuencode command to encode the file "myfile.doc" and store the name "myfile.doc" in the resulting encoded file. The results are then redirected (by the > sign) into another file that you'll place into your mail message later.

DOS versions of this utility are also easily available at various download sites.

****

Now, once you have encoded the file and obtained the Uuencoded form, then all you need to do is Copy it and Paste it after the DATA command has been issued at the Sendmail prompt. This will send the file as an attachment.

This was the method in which one can send attachment using the Uuencoding standard. I will describe how to send attachments using the new MIME standard in the later version of this manual.

More Sendmail Tips and Tricks

Normally when you connect to the Sendmail Port of a system, then you only have standard SMTP commands available to you. Although they are more than what you will ever need, however, for those of you who like to play with various options, there are also some other commands, which are by default not available to you.

What I am talking about here is ESMTP commands or Extended Mail Transfer Protocol commands. A mail server with ESMTP enabled decides whether these ESMTP commands are available to the client on the basis of how the client introduces itself to it. Now, normally you introduce yourself by giving the below command:

HELO domain

Now, when you introduce yourself using the HELO command, then most mail servers by default make only the SMTP commands available to the client. Now, in order to make sure that even the ESMTP commands are available to you, you need to introduce yourself to the server by the EHLO command. For Example:

ehlo ankit.com

Now, if the mail server you are connected to, has ESMTP enabled, then it will respond by giving a list of ESMTP commands. Something like the below:

220 mail.isp.com ESMTP Sendmail 8.9.1 (1.1.20.3/07Jul00-0916AM) Thu, 7 Dec2000 17:18:50 +0530 (IST)

ehlo ankit.com

250-mail.isp.com Hello [203.xx.yy.91], pleased to meet you

250-EXPN

250-VERB

250-8BITMIME

250-SIZE

250-DSN

250-ONEX

250-ETRN

250-XUSR
250 HELP

****

HACKING TRUTH: One way of finding out whether your ISP has ESMTP commands enabled, is to see the daemon banner that comes up, when you telnet to Port 25 of its mail server. The word 'ESMTP' tells you that such commands are available. For Example,

220 mail.isp.com ESMTP Sendmail 8.9.1 (1.1.20.3/07Jul00-0916AM) Thu, 7 Dec2000 17:18:50 +0530 (IST)

****

Coming Soon (Probably on Monday): How to send more authenticate mails. More Tricks to play with email headers.

Ankit Fadia

ankit@bol.net.in

http://www.ankitfadia.com

To receive manuals on EVERYTHING YOU DREAMT OF written by Ankit Fadia, in your Inbox join his mailing list, by sending a blank email to: programmingforhackers-subscribe@egroups.com