Vol. 15 No. 2 October 2002
$2.50

MUUGlines

The Manitoba UNIX User Group Newsletter

October 8, 2002: Introduction to DHCP

Kevin McGregor, Network Analyst with the City of Winnipeg, will describe the theory, benefits and general operation of the Dynamic Host Configuration Protocol. DHCP is a well-defined protocol which is widely implemented but not well understood. This presentation will leave you with a good idea of how and why to implement DHCP in your network environment, whether you manage a home network or a WAN in a large company.

The meeting will also include our usual roundtable discussion, plus the votes on both the by-law amendments and the MUUG board of directors.

Meetings are held at the IBM offices at 400 Ellice Ave. (between Edmonton and Kennedy). When you arrive, you will have to sign in at the reception desk, and then wait for someone to take you (in groups) to the meeting room. Please try to arrive by about 7:15 PM, so the meeting can start promptly at 7:30 PM. Don't be late, or you may not get in.

Limited parking is available for free on the street, or in a lot across Ellice from IBM, for $1.00 for the evening. Indoor parking is also available nearby, at Portage Place, for $2.00 for the evening.

Vote on Proposed Amendments to MUUG By-Laws

At the September meeting, a notice of motion to amend the MUUG by-laws was tabled for discussion. This same notice of motion was published in the September issue of the newsletter, in order to provide all members with sufficient advance notice, in compliance with the by-laws.

As a result of the discussion, two additional names were gathered in support of the motion, to meet the requirement for a total of five supporters, according to by-law 12.(b). We now reprint the notice of motion, updated to include the names of all five supporters. All members in good standing are hereby invited to attend the October 8 meeting to vote on this motion.

Notice of Motion to Amend MUUG By-Laws

To be tabled as a Special Resolution at the Annual Meeting on October 8, 2002.

That the MUUG By-Laws (Draft 6, dated September 28, 1992) be amended as indicated below:

1. (f) shall now read as follows: "UNIX is a registered trademark of The Open Group in the United States and other countries"

3. (c) shall no longer include: "(e.g.: UniForum and its affiliates)"

3. last sentence of last paragraph shall no longer include: ", and supporting the activities and initiatives of UniForum Canada"

5. (a) "not less than five Directors" shall be revised as follows: "not less than three Directors"

11. (f) shall be deleted.

11. (f) shall now read as follows: "The Treasurer will report directly to Members at a general meeting at least once per year."

12. (c) shall be deleted.

Submitted by Gilbert Detillieux. Supported by Kevin McGregor, Doug Shewfelt, Adam Thompson, and Shawn Wallbridge.

The complete text for the current draft of the by-laws can be found online, in various formats: http://www.muug.mb.ca/pub/bylaws/

General Election 2002

As required by the MUUG by-laws, following are the nominees for MUUG Board Directors, their supporters, and biographical information.

Nominee: Gilbert Detillieux, Systems Analyst, University of Manitoba

Supported by: Michael Doob, Professor, University of Manitoba

Gilbert Detillieux has been working with UNIX as a programmer, system administrator, and trainer since 1980. He worked as a computer consultant specializing in UNIX, from 1983 to 1989. He is currently working as a Systems Analyst for the University of Manitoba's Department of Computer Science, where's he's worked since 1989, installing, supporting and upgrading the department's network and UNIX server infrastructure. He was co-founder and past president of the Technical UNIX User Group (now MUUG), and has been an active member of the MUUG board ever since. He also contributed to the implementation of the initial MUUG Online system, in 1992, which he's since maintained through two complete upgrades and numerous revisions over the years.

Nominee: Kevin McGregor, Network Specialist, City of Winnipeg

Supported by: Shawn Wallbridge, Frantic Films

Kevin McGregor provides network administration and system support services for the City of Winnipeg. After having briefly used UTS on an Amdahl mainframe in university in the mid-80s, he dabbled in Coherent and then converted to Linux. He has been a member of MUUG since the early 90s and has served on the board for the majority of that time. Kevin is the editor of the MUUG newsletter.

Nominee: Doug Shewfelt, Systems Specialist, City of Winnipeg

Supported by: Kelly Kitson, Canadian Wheat Board

Doug Shewfelt first encountered UNIX in 1980 at the University of Manitoba. He has been worked in various IT positions for the City of Winnipeg since 1985, and currently supports the City's corporate financial systems. He has been a member of the MUUG board for the past several years.

Nominee: Adam Thompson, Consultant, athompso.net

Supported by: Gilbert Detillieux, Systems Analyst, University of Manitoba

Adam has over 15 years' experience in the IT field, ranging from Programmer to System Administrator to Chief Technology Officer. He currently teaches a variety of IT courses and provides consulting services to clients.

Nominee: Sean Walberg, Network Analyst, Ceridian Canada

Supported by: Doug Shewfelt, Systems Specialist, City of Winnipeg

Sean started using Unix in 1994 at the U of M, and moved into Systems Administration and Linux in 1995 with a small ISP. Sean joined MUUG about two years ago, and has given two presentations. He's also been writing a weekly newsletter about Linux for the past two years. He's looking forward to bringing his experience to the MUUG Board.

Nominee: Shawn Wallbridge, Frantic Films

Supported by: Kevin McGregor, Network Specialist, City of Winnipeg

Shawn Wallbridge discovered Linux in 1996 while at Red River College. Over the years he has used Unix wherever possible for various functions. He is currently the Systems Administrator for Frantic Films and runs a Web hosting company in his spare time. He runs seven different versions of Unix at home. He has been a MUUG member for over two years and has done numerous presentations for MUUG.

Using vi (part I)

Copyright 2002 by Daniel Christle

Regardless of the operating system you use, when you have problems you end up at the command line.

Invariably there is some configuration file that needs to be edited or recreated. Under DOS and Windows (all versions) we have DOS Edit. In UNIX there is vi. Despite the fact that there are literally dozens of text editors available for UNIX and Linux, only vi is consistently available on all *nix platforms. vi is available in a wide range of variants, the most popular of which is vim.

Before we get started on how to use vi, we should look at the history of this wonderful and powerful application.

vi was written by Bill Joy, currently the chief scientist for Sun Microsystems, in the early Seventies. He was looking for a replacement to the popular line editors of the time: ex and ed. The challenge was to come up with a screen editor which allowed the user to view more than one line of information at a time. The other major challenge in designing a screen editor is that it had to be small enough to be effective over a phone line as it would be run from a main frame and displayed on a terminal screen; not run from a local computer as is done today. And so vi was born, very easily meeting its design goals. Somehow I doubt that Bill Joy really expected his creation to still be in use today, some 30 years later.

vi is a multi-modal editor that relies on keyboard commands to get the job done. Though it often appears somewhat confusing and mysterious to the uninitiated it is really quite simple and extremely fast. As I mentioned earlier vi is multi-modal, it has two modes: command mode and insert mode. Command mode allows you to enter commands, execute macros or use expressions, while insert mode allows you to enter text into your document. Moving between the two modes is easy as we will see as we create a text file.

To start vi from the command line we type vi and up pops the editor. To open a file we would use the command:

vi *nameoffile*

You can also use this same technique to create a new file. The first time you save the file is when the file is physically created. Immediately upon opening the file, you are in command mode. Here you can issue a number of commands. We should first practice closing and saving the file. To save the file you issue the following command:

:w

The colon tells vi that you are issuing a command. The 'w' stands for write as saving a file is really the physical task of writing a file to disk.

To quit vi you have several options. You could simply use the following command:

:q

This allows you to exit straight out vi. However it is always nice to save and exit at the same time. To do this you can combine the two above commands like so:

:wq

Alternatively you could also enter the following command to achieve the same thing:

ZZ

Note that the 'ZZ' must be capitalized.

Reopen the file you created and insert some text. To get into insert mode you have two options. You can press 'i' or 'a'. Pressing 'i' allows you to enter text before the cursor while pressing 'a' allows you to enter text immediately to the right of the cursor. To move back to command mode in order to save your work, press the Escape key. You are now in command mode. Make sure to save your work using the commands above. If you notice that the text isn't wrapping you can use the escape key to move back to command mode and issue the following command:

set wrapmargin=10

Being able to navigate around a document is crucial to your productivity. Therefore vi provides several methods to get around your document. Each of these methods is executed from command mode.

When vi was created there wasn't such a thing as arrow keys on the keyboard. Instead you used the h, j, k, and l keys to navigate around the screen. The h key allows you to move to the left, the j key allows you to move downwards, the k key allows you to move upwards, and the l key facilitates movement to the right. The good news is that if you are using one of the modern vi variants such as vim you can simply use the arrows keys.

However, it pays to be aware of the more traditional navigation keys in the event that you are stuck using the original vi or one its derivatives. Besides, they are much faster when you consider that you must physically move you hand to use them. The traditional keys allow you to stay in the proper typing position.

Naturally there are more methods of navigation that are more powerful and more flexible. To move to the end of a line you can simply press the $ key. To move to the beginning of a line press the 0 (zero). There are also methods for moving from word to word and sentence to sentence. To move to the beginning of the current sentence press the '('. To move to the beginning of the next sentence you use the ')'. To move backwards by word just press 'b', to move forwards by word press 'w'. To move to the end of a word press 'e' or 'E'. You can also move to specific lines in a document use the goto command which represented by the 'G' key. If you want to move to line 15 you use the following command:

15G

You can move to the last line in the file by pressing 'G' by itself. To move to the first line in the file you would just type: 1G. If you are moving around in this manner and haven't edited the file and want to return to your previous position you enter a two single back-quotes (``).

Like most editors, vi lets you cut, copy, and paste text. This is accomplished using three commands: delete, yank, and put. As with navigation these commands are executed in command mode.

To delete text you use ‘d’. If I want to cut text from the cursor position to the end of the line I can combine the delete command with the end of line command by pressing ‘d$’. To cut single characters use the ‘x’. If I want to just copy the text rather than cut or delete it I use ‘y’ instead. To copy more than one line I can use the command: 4yy. This tells vi to copy four lines from the position of the cursor. You can apply the same methodology to the delete command. In order to place the text we have copied or deleted back into the document, move your cursor to the desired position. Press ‘p’.

The text is now pasted into the document. If you make a mistake you can use the undo command (‘u’).

We have now covered the basics of using vi. You have learned to open, create, and save files. You have also learned the basics of navigating a document and how you can edit it using the basic editing commands. These are but a few of the command available to you when using vi. Next month we will look at search and replace, pattern matching and ways to customize your vi environment. Until then, keep practicing.

Regal Goes With Linux

Regal Entertainment Group, operator of America’s largest chain of movie theatres, has begun using new IBM point-of-sale systems and open-source Linux technology to serve millions of customers at its high-volume Regal Cinemas concession stands in one of the first and largest rollouts of Linux in the retail point-of-sale environment, IBM announced on September 25.

Regal is also testing a new, in-theatre, Linux-based kiosk that will enable movie patrons to purchase tickets or retrieve tickets purchased from an online service.

“After carefully examining a variety of options for our concession stand and kiosk needs, we selected IBM as our strategic technology partner. IBM’s reputation for reliability and customer service to the Retail sector, coupled with the excellence of their SurePOS technology design, made our choice easy,” said J.E. Henry, Senior Vice President and Chief Information Officer, Regal Entertainment Group. “With the nationwide scale of our business and our need to handle millions of concession and kiosk transactions annually, we needed a strong, reliable partner like IBM to support our growth.”

Regal Cinemas is using rugged IBM SurePOS 500 point-of-sale systems at its concession stands, running Linux from Red Hat and linked with Regal’s IBM eServer iSeries, a mainframe-class midrange server, at Regal’s headquarters in Knoxville, Tennessee. An average of 13 SurePOS 500s will be used at each theatre complex.

“We chose Linux because it is an open, affordable operating system, giving us lower cost and relative ease of programming with the Java2 projects we already had in place,” said Todd King, Vice President of Technical Information Services, Regal Entertainment Group. “We need a rugged, reliable technology solution for the time-sensitive, high-volume traffic in our concession stands, as well as a platform on which to build future applications, su h as our ticket kiosk.”

“The concession stand is a significant profit center for a theater and is one of the most important and demanding areas of theater operations due to the need to provide service to a large number of customers in a very short time. They cannot afford downtime or rebooting when customers are in line for concessions and the movie is about to start,” said Tom Peterson, General Manager, IBM Retail Store Solutions. “The fact that Regal is using Linux in such a critically important way demonstrates that Linux running on retail-hardened IBM point-of-sale systems can be an attractive alternative for retailers.”

Sending Us E-Mail?

Due to the amount of e-mail MUUG receives, we’ve set up an auto-reply to give you immediate feedback, and redirect some of the e-mail to the appropriate places. Why not look at http://www.muug.mb.ca/about.html#contacts first?

Share Your Thoughts

E-mail us with your comments on the newsletter, whether it’s criticisms or commendations, and continue to send in articles or ideas for same. If you have a How-To or other idea, and aren’t ready to give a presentation at MUUG, an article is a great alternative! If you can write better than the editor, that’s terrific; if you can’t, submit it anyway and we’ll get it into shape for publication. We know that many of you have some great ideas and lots of knowledge. Why not share? Mailto: editor@muug.mb.ca.