

MAGAZINE

# BSD

FOR NOVICE AND ADVANCED USERS

## NEARLY ONLINE ZPOOL SWITCHING BETWEEN TWO FREEBSD MACHINES

OPENSCH JUMP SERVER WITH 2FA

OPENBSD 6.1 NEW FEATURES

DANIEL MIESSLER'S BLOG

GUI PROGRAMMING IN FREEBSD WITH PERL/TK

DEVOPS WITH CHEF ON FREEBSD

DEVELOPMENT TOOLS

VOL 11 NO 04

ISSUE 03/2017 (92)

ISSN 1898-9144



# FREENAS MINI STORAGE APPLIANCE

IT SAVES YOUR LIFE.



## HOW IMPORTANT IS YOUR DATA?

Years of family photos. Your entire music and movie collection. Office documents you've put hours of work into. Backups for every computer you own. We ask again, *how important is your data?*

## NOW IMAGINE LOSING IT ALL

Losing one bit - that's all it takes. One single bit, and your file is gone.

The worst part? **You won't know until you absolutely need that file again.**



*Example of one-bit corruption*

## THE SOLUTION

The FreeNAS Mini has emerged as the clear choice to save your digital life. **No other NAS in its class offers ECC (error correcting code) memory and ZFS bitrot protection to ensure data always reaches disk without corruption and *never degrades over time.***

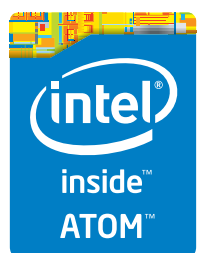
No other NAS combines the inherent data integrity and security of the ZFS filesystem with fast on-disk encryption. No other NAS provides comparable power and flexibility. The FreeNAS Mini is, hands-down, the best home and small office storage appliance you can buy on the market. **When it comes to saving your important data, there simply is no other solution.**

### The Mini boasts these state-of-the-art features:

- 8-core 2.4GHz Intel® Atom™ processor
- Up to 16TB of storage capacity
- 16GB of ECC memory (with the option to upgrade to 32GB)
- 2 x 1 Gigabit network controllers
- Remote management port (IPMI)
- Tool-less design; hot swappable drive trays
- FreeNAS installed and configured



<http://www.iXsystems.com/mini>



# FREENAS CERTIFIED STORAGE



With over six million downloads, FreeNAS is undisputedly *the* most popular storage operating system in the world.

Sure, you could build your own FreeNAS system: research every hardware option, order all the parts, wait for everything to ship and arrive, vent at customer service because it *hasn't*, and finally build it yourself while hoping everything fits - only to install the software and discover that the system you spent *days* agonizing over **isn't even compatible**. Or...

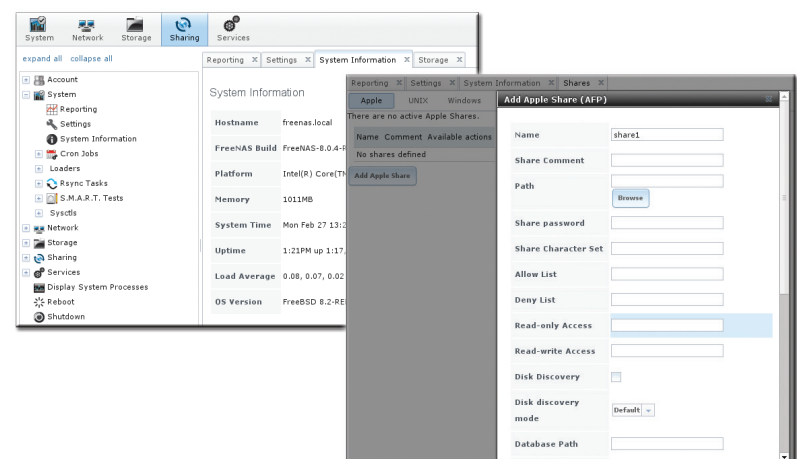
## MAKE IT EASY ON YOURSELF

As the sponsors and lead developers of the FreeNAS project, iXsystems has combined over 20 years of hardware experience with our FreeNAS expertise to bring you FreeNAS Certified Storage. **We make it easy to enjoy all the benefits of FreeNAS without the headache of building, setting up, configuring, and supporting it yourself.** As one of the leaders in the storage industry, you know that you're getting the best combination of hardware designed for optimal performance with FreeNAS.

## Every FreeNAS server we ship is...

- » Custom built and optimized for your use case
- » Installed, configured, tested, and guaranteed to work out of the box
- » Supported by the Silicon Valley team that designed and built it
- » Backed by a 3 years parts and labor limited warranty

As one of the leaders in the storage industry, you know that you're getting the best combination of hardware designed for optimal performance with FreeNAS. **Contact us today for a FREE Risk Elimination Consultation with one of our FreeNAS experts.** Remember, every purchase directly supports the FreeNAS project so we can continue adding features and improvements to the software for years to come. **And really - why would you buy a FreeNAS server from *anyone* else?**



### FreeNAS 1U

- Intel® Xeon® Processor E3-1200v2 Family
- Up to 16TB of storage capacity
- 16GB ECC memory (upgradable to 32GB)
- 2 x 10/100/1000 Gigabit Ethernet controllers
- Redundant power supply

### FreeNAS 2U

- 2x Intel® Xeon® Processors E5-2600v2 Family
- Up to 48TB of storage capacity
- 32GB ECC memory (upgradable to 128GB)
- 4 x 1GbE Network interface (Onboard) - (Upgradable to 2 x 10 Gigabit Interface)
- Redundant Power Supply

<http://www.iXsystems.com/storage/freenas-certified-storage/>





**Dear Readers,**

So, we meet again. I hope you had a great time and April was a fruitful month for you. Since all of you follow the latest news from the open-source world and there were at least some successes in the field, I think it is worth mentioning the OpenBSD release. The OpenBSD 6.1 release has a few significant changes. The most visible features are a new syspatch(8) utility for binary base system updates to supported -stable amd64 and i386 releases; the acme-client, enhancements to vmm(4), new arm64 platform, new xenodm(1) X display manager and many more. Now, thanks to all these features, you can run the OS efficiently. Also, thanks to Albert Hui, you can learn more about these features and widen your knowledge about OpenBSD release. I recommend reading his article that is published in this BSD magazine. Albert selected the most significant changes and covered them extensively in his article. You will definitely enjoy the read.

I would like to bring to your attention what we prepared for you and the contents of this BSD issue. For FreeBSD fun, I have three articles. The first article was written by an excellent and well-known writer, Mikhail E. Zakharov. His article will teach you how to use two storage pools (one pool per controller) in conjunction with the BeaST Quorum to automate zpool switching between both active controllers. If you are still interested in the FreeBSD world, then you need to read the article entitled *GUI Programming in FreeBSD with Perl/Tk* by Abdorrahman Homaei. You will learn that using Perl and Tk(PTk) for GUI programming is low dependency, fast and geek-style. Thus, It's a must read for you. And for those of you who read the BSD magazine and still don't know which OS you should switch to, you are not alone. I have good read for you too. Just scroll down to the article by David Carlier and see how many amazing tools you can use when you choose the FreeBSD OS.

As usual, I must include an emergent typical security topic. Hence from the next article, you will learn how to create a secure OpenSSH Jump server with a two-factor authentication. The article was written by the famous Leonardo Neves Bernardo, and he will present all about the OpenSSH Jump Server. Also, being in the security world, you shouldn't miss the interview with Babar Khan Akhunzada, the founder of SecurityWall.

Another author worth mentioning to you is Daniel Miessler. He is kindhearted and agreed to answer a few questions for this month's blog presentation. I hope you will enjoy reading his interview and post that he deemed fit for you.

Lastly, I would like to invite you to read the column by Rob Somerville. It is always good reading, kind of like a "dessert".

As long as we have our esteemed readers, we have a purpose. We owe you a huge THANK YOU. Additionally, we are grateful for every comment and opinion, whether positive or negative. Every word from you compels us to improve the BSD magazine and brings us closer to the ideal shape of our publication.

I hope we will share again in a month's time. We're here for you!

Best regards,  
*Ewa & The BSD Team*



# TABLE OF CONTENTS

## News\_\_\_\_\_06

*Ewa & The BSD Team*

This column presents the latest news coverage of breaking news events, product releases and trending topics of the new stories from BSD.

## BASICS

### DevOps with Chef on FreeBSD\_\_\_\_\_12

*Arun Tomar*

Chef is a powerful automation platform that transforms infrastructure into code. Also, it's supported by almost all major Linux, Unix, Windows, cloud and container platforms. Read and learn more about it from the article by Arun Tomar.

## OPENBSD

### OpenBSD 6.1 New Features\_\_\_\_\_16

*Albert Hui*

The recently released OpenBSD 6.1 is a major release and contains updates. This article will cover all the significant changes and features of the new release.

## FREEBSD

### Nearly Online Zpool Switching Between Two FreeBSD Machines\_\_\_\_\_22

*Mikhail E. Zakharov*

Although this experiment looks like a rough example of the fail-over/fail-back procedures for ZFS data pools, it is actually a breakthrough for our storage system development. After several enhancements to support reliability and workload balancing, we will use two storage pools (one pool per controller) in conjunction with the BeAST Quorum to automate zpool switching between both active controllers. Learn from Mikhail Zakharov on how to achieve this.

### GUI Programming in FreeBSD with Perl/Tk\_\_\_\_\_26

*Abdorrahman Homaei*

There are many solutions to GUI programming in FreeBSD, like JavaFX, QtCreator or U++. However, using Perl and Tk(PTk) is low dependency, fast and geek-style. Sometimes it's better to get our hands dirty than debug code constantly. Find out more with Abdorrahman Homaei.

### Become FreeBSD User: Find Useful Tools\_\_\_\_\_30

*David Carlier*

FreeBSD has made tremendous improvements over the years to fill the gaps in Linux. Whereas it still keeps its

interesting specificities, there would not be many blockers if your projects are reasonably sized to consider a migration to FreeBSD.

## UNIX

### OpenSSH Jump Server with 2FA\_\_\_\_\_34

*Leonardo Neves Bernardo*

SSH is the most convenient way to log in a remote UNIX-like machine and execute commands. Used virtually in every Unix-like box, SSH became one of the most known targets for people that usually try to invade our systems. This article will discuss how to create a secure OpenSSH Jump server with a two-factor authentication.

## UNIX BLOG PRESENTATION

### Daniel Miessler's Blog\_\_\_\_\_42

*Daniel Miessler*

The goal of A vim Tutorial and Primer tutorial is to take you through every stage of progression—from understanding the vim philosophy (which will stay with you forever), to surpassing your skill with your current editor, and finally becoming “one of those people”.

## INTERVIEW

### Interview with Babar Khan Akhunzada\_\_\_\_\_54

*Marta & The BSD Team*

Basically, SecurityWall is a team of young Security Researchers & Security Experts. They have been working in this field for years, and are also acknowledged by many high profile companies on their security knowledge.

## COLUMN

**The personal computer user interface has come a long way since the day of toggle switches and LED's. Despite all the technological advances, end users still struggle in operating software despite displays that support millions of colors and innovative software controls. What is the secret to a good GUI?**

*Rob Somerville*

56



## OpenBSD 6.1 Released

The OpenBSD project is pleased to announce the official release of OpenBSD 6.1. This is our 42nd release. They remain proud of OpenBSD's record of more than twenty years with only two remote holes in the default install. This release has several changes but the most visible are:

- New syspatch(8) utility for binary base system updates to supported -stable amd64 and i386 releases.
- The acme-client, a privilege separated ACME client for easy maintenance of Let's encrypt TLS certificates.

We expect these items will make the day to day running of OpenBSD systems significantly easier. Other improvements include:

- Several enhancements to vmm(4), including support for third-party BIOSes and Linux guests.
- New arm64 platform targeting Pine64, Raspberry Pi 3 and Opteron A1100  
Continuing SMP improvements, particularly in the network stack.
- New xenodm(1) X display manager.
- Improved capabilities in a number of IEEE 802.11 wireless network drivers.
- Updates to the package system tools as well as the package collection itself, with increased number of prebuilt packages for the more popular (and faster) architectures.
- This release has also updated versions of OpenSMTPD, OpenSSH, LibreSSL, mandoc as well as incremental improvements to all other named subprojects.

A partial list of new features and systems included in OpenBSD 6.1. can be found on the release page, and if you need a comprehensive list, see the [changelog](#) leading to 6.1.

Source: <https://www.openbsd.org/61.html>

## 2017 Cambridge DevSummit ('BSDCam')

The 2017 Cambridge DevSummit is scheduled for 2-4 August 2017. Registration is not yet open, and further information will be availed online as we draw closer to the set dates.

The event is run in an "un-conference style". In that, we brainstorm the actual session schedule on the first morning, with a focus on interactive topics that reflect the interests and exploit the knowledge of the attendees -- but there's also room for traditional talks, etc. There are plenty of break-out rooms for small groups to meet as they see fit.

Source: <https://wiki.freebsd.org/DevSummit/201708>



# TrueNAS Storage Primer on ZFS for Data Storage Professionals

If you are a storage professional but new to [TrueNAS](#) and OpenZFS, their operations and terms may be a little different for you. The purpose of this blog post is to provide a basic guide on how OpenZFS works for storage, and to review some of the terms and definitions used to describe storage activities on OpenZFS. A quick dictionary of OpenZFS terms can be found [here](#).



The TrueNAS data storage system from iXsystems uses OpenZFS as the underlying file system and volume manager. TrueNAS is based on the Open-Source software-defined storage operating system, [FreeNAS](#), which is based on the [FreeBSD](#) Open-Source operating system.

Source: <https://www.ixsystems.com/blog/truenas-storage-primer-zfs-data-storage-professionals/>

## Upcoming GhostBSD 11.0 Delayed

The upcoming version of GhostBSD will include its software package repositories instead of using those from FreeBSD, and it is the main reason for the delay of the release.

### Why Ghost Project creates their repositories:

- FreeBSD is geared more towards servers while GhostBSD targets the desktop. The Ghost project will maintain its software packages repositories, and it gives it more control and allows it to set build-time options differently.
- It makes it possible to update GhostBSD specific applications (e.g. the update station) via pkg, too.
- Creating an additional repository would lead to technical problems, and it may break some applications (packages from two different repositories could depend on various versions of the same dependency package).

### GhostBSD will provide packages for i386 and amd64. For both architectures there will be three public repositories:

- **latest:** which matches FreeBSD's ports + GhostBSD ports + changed options and perhaps patches. The devs will run this, and advanced users who are willing to test are welcome, too.
- **current:** which will be the default repository for GhostBSD.
- **previous:** which keeps the previous packages available after "current" was updated.

Source: [http://www.ghostbsd.org/UpcomingGhostBSD11.0\\_delayed\\_for\\_repositories](http://www.ghostbsd.org/UpcomingGhostBSD11.0_delayed_for_repositories)

## FreeNAS: An Ideal Storage Platform for Network Administration Education

FreeNAS works very well straight out of the box and doesn't need tinkering to get things up and running. We have a NetApp system that we also use in our coursework. I don't feel as comfortable letting students use it as any change they make could have college-wide





ramifications. This would require me to fix it, and take time away from my role as an educator. FreeNAS, on the other hand, is difficult to corrupt, especially with its snapshot capabilities that allow you to roll back the file system to previous states.

The students should just get FreeNAS. It is instinctively understandable. This means, they will spend less time drilling down on all the particulars of administering a storage solution like NetApp and more time learning the language and fundamental concepts of storage. Also, in the course of our two-year program, our students walk away with a strong foundation in storage administration.

FreeNAS ensures that our students are prepared for the workforce. Even if the organization they go to work for uses some other storage platform, working directly with FreeNAS gives them the ability to adapt with little difficulty.

We had one student in our program who was recently hired full-time for a position. The customer was a NetApp shop, but FreeNAS gave him the knowledge and confidence to be conversant in storage concepts. He is not alone in his success. Overall, students have done extremely well and have around an 85% placement within six months after participating in our two-year program. Right now, we have regional employers calling and asking us for more graduates and they even send their own people for us to train.

Source: <http://www.freenas.org/blog/freenas-ideal-storage-platform-network-administration-education/>

## **Dell EMC Releases Open-Source Storage Updates Designed for Security and Easier Consumption**

Dell Technologies today announced the latest updates in a series of open-source contributions from {code} by Dell EMC, including trust and security enhancements to REX-Ray in 0.9, the newest version of the leading container storage orchestration engine. Additionally, REX-Ray is now shipping Docker Certified managed volume plugins for cloud and storage platforms available in the Docker Store. The Docker Certification Program is a framework for partners to integrate and certify their software for the Docker Enterprise Edition (EE) commercial platform. Other announcements include collaboration on a universal Container Storage Interface and unification of Polly's features and roadmap into REX-Ray.

Container technology has made strides since last year's DockerCon, but there are still barriers to organizations consuming them in an enterprise environment—and that means it's time to put industry-standard architectures and security measures in place for this technology. Mitigating the risks of adopting containers for critical cloud native persistent workloads has proved to be a significant challenge, and REX-Ray 0.9 addresses this challenge through its controller model that ships with security enabled by default. REX-Ray can run from anywhere while protecting sensitive credentials, encrypting communication, and performing per-client authentication. The Docker Store is the enterprise user's go-to destination for trusted, enterprise-ready Docker containers, plugins and editions. Support for storage platforms that work with Docker is made available through Docker Certified managed volume plugins. These plugins greatly enhance the user experience in ensuring integration to orchestrate persistent applications with external storage. The Docker Certified REX-Ray plugins include Amazon EBS/EFS, Dell EMC ScaleIO/Isilon, Google Compute Engine PD, and any S3-compatible storage.



“Demand for storage with containers is being driven by organizations looking to use containers for all applications. Docker Enterprise Edition platform and its pluggable architecture is the enabler of a new opportunity for storage in the enterprise. REX-Ray through its features and security is a great vehicle to ensure interoperability for many storage platforms,” said Marianna Tessel, EVP Strategic Development at Docker.

Polly, named for “polymorphic volume scheduling,” is an open-source volume scheduling service centrally working with container schedulers to provide volume resources when requested. The introduction of Polly's scheduling services inside

of REX-Ray's centralized controller will enable users to deploy a single storage service that provides governance and security for critical storage services. Additionally, through its experience in building libStorage with Polly, the {code} team is collaborating with industry leaders on a Container Storage Interface initiative through the Cloud Native Computing Foundation to ensure container orchestrators, storage controllers, and agents have a universal way of interoperating in the future.

"Containers have come a long way towards becoming enterprise ready with the addition of enterprise-grade features such as security and more common interfaces. Our work with the CNCF ensures that, once the Container Storage Interface is ready, REX-Ray will bring support quickly to existing storage drivers and we can start building long-lasting value on top of this work. Collaborative projects only benefit the enterprise if users can actually consume it, and that's the thinking behind our direction and updates through libStorage and REX-Ray 0.9," said Josh Bernstein, VP of Technology at Dell Technologies.

In addition to the project updates, {code} by Dell EMC recently discussed community momentum, including the state of the {code} Catalyst program.

Source: <https://www.emc.com/about/news/index.htm>

## NetApp Showcases Cloud-Connected, Next-Generation Media and Entertainment Solutions at NAB 2017

NetApp (NASDAQ: NTAP) will showcase cloud-connected and next-generation solutions and strategies for the media and entertainment industry at the 2017 NAB Show. The company, in conjunction with its partners, will demonstrate how NetApp® data management solutions can accelerate production, modernize media data centers, and transform global repositories. The conference takes place from April 22 through April 27, 2017, at the Las Vegas Convention Center.

"Media companies are challenged to find a balance between tried-and-true production approaches, and designing global architectures for the future," said Jason Danielson, Media and Entertainment Solution Manager at NetApp. "NetApp and its partners have supported hundreds of customers in integrating the public cloud into their operations, enabling them to seamlessly manage their data from on the premises to the cloud."

### NetApp Demonstrations at NAB

NetApp cloud media services comprise several products within the NetApp Data Fabric portfolio, such as NetApp ONTAP® Cloud, NetApp Cloud Sync service, and NetApp Private Storage. These products enable seamless data management across IT environments, including Amazon Web Services (AWS). Demonstrations of NetApp cloud media services will take place in AWS Elemental Media Services booth SU2202.

NetApp StorageGRID® Webscale is an object storage solution for rich content, including videos and images, that combines a global namespace, enterprise reliability, and world-class support with industry-leading procurement and deployment options. Attendees can see StorageGRID Webscale integrations in Dalet booth SL6210, in Scale Logic booth SL5324, and in Telstra booth SU8812. A presentation will also be given at 3 p.m. Pacific Time on April 26 in the eMAM booth SL14509.

NetApp E-Series hybrid and all-flash storage arrays provide up to 12 uncompressed 4K frame-based video streams in only 2U with five-nines reliability. When paired with systems from Pixit Media, Quantum, EditShare, and Scale Logic, NetApp E-Series has become the basis for new broadcast and postproduction solutions for workgroups of every size. E-Series integrations will be on display in Scale Logic booth SL5324, in ATTO booth SL9611, and at 11:30 a.m. Pacific



Time on April 25 in a Pixit Media presentation in NetApp meeting room S101LMR. Contact the NetApp for an invitation.

Source: <http://www.netapp.com/us/company/news/press-releases/news-rel-20170419-845568.aspx>

## **The OpenStack Summit, Boston, MA May 8-11, 2017**

The OpenStack Summit will be held in Boston, May 8-11, 2017, at the Hynes Convention Center and surrounding hotels.

The world runs on open infrastructure. At the OpenStack Summit, you'll learn about the mix of open technologies building the modern infrastructure stack, including OpenStack, Kubernetes, Docker, Ansible, Ceph, OVS, OpenContrail, OPNFV, and more. Whether you are pursuing a private, public or multi-cloud approach, the OpenStack Summit is the place to network, skill up and plan your cloud strategy. Hear business cases and operational experience directly from users, learn about new products in the ecosystem and participate in hands-on workshops to build your skills. Attended by thousands of people from more than 60 countries, it's the ideal venue to plan your cloud strategy and share knowledge about architecting and operating OpenStack clouds.

The Summit will run for 4 days, Monday - Thursday, May 8-11, 2017: Comprised of presentations, panels, workshops, and educational opportunities through OpenStack Academy.

Topics span from cloud strategy and business case development to operational best practices and technical deep dives. Keynote presentations from notable OpenStack users and industry leaders will take place on Monday and Tuesday

Source: <https://www.openstack.org/summit/boston-2017/>

## **Red Hat Summit 2017 on May 2-4, 2017 in Boston, Massachusetts**

Check out the hundreds of sessions, labs, and more at this year's Red Hat Summit—all of which can be added to your agenda.

It's the power of the individual that makes open-source great—so this year we're focusing on you more than ever before. Learn, network, and experience opensource at every level with:

- Hundreds of sessions led by the best and brightest in open source.
- Hands-on experience with our newest technologies.
- Face-to-face interaction with product experts—or the actual builders—demonstrating the latest technologies.
- The chance to grow your network, reconnect with peers, and make new partnerships.

Source: <https://www.redhat.com/en/summit/2017>

# Developing Java EE Applications on Cloud

## What you will learn...

- How to use RAD to create Java EE applications.
  - Connect RAD to a PureApplication.
  - Create a Cloud application in RAD.
- Publish the cloud application onto PureApplication.
  - Use the Virtual Application Builder in PureApplication to build the Virtual Application Pattern topology.
- Deploy the Virtual Application Pattern from RAD to the private cloud.

## What you should know...

- Database and JPA concepts.
  - Basic Java EE knowledge.
- Basic concepts of cloud computing.

Free Reading

[www.SDJournal.org](http://www.SDJournal.org)



## DevOps with Chef on FreeBSD

### What you should know ...

- Knowledge of how to use version control system, preferably Git.
- Basic Knowledge of FreeBSD OS Administration.
- Basic knowledge of any scripting language, preferably Ruby.
- Working knowledge of any text/code editor.

### Resources

#### FreeBSD

[FreeBSD Documentation](#)

[Download FreeBSD](#)

#### Chef Website

[Chef Documentation](#)

[Chef IRC Channel](#)

[Chef Mailing List](#)

#### Git website

#### Ruby Website

### Hardware/Software Requirements

2 FreeBSD 11 systems/machines (Virtual or Physical), with Internet access and bash as the default shell with root login over ssh or a normal user with sudo access.

A free account on [Hosted Chef](#).

### What is DevOps?

According to Wikipedia, “**DevOps** (a clipped compound of development and operations) is a culture, movement or practice that emphasizes the collaboration and communication of both software developers and other information-technology (IT) professionals while automating the process of software delivery and infrastructure changes.”



Figure 1. DevOps

### What is automation?

It depends on the context and situation. In general, automation is a non-manual way of performing a certain set of tasks. In IT infrastructure automation, this set of tasks could be: creating new servers and infrastructure, managing existing servers, deploying applications, updating servers/applications, monitoring servers/apps, gathering metrics, graphs, scaling up or down the infrastructure and much more.

### Why is it needed?

Virtualization and Cloud has forced the need for automation. In the earlier days (The Iron Age), IT infrastructure growth was dependent on the physical hardware purchasing cycle. It used to take weeks from order to actual server being delivered and then deploying the OS and have the server ready for use.

Because of virtualization, especially the Cloud, the bottleneck is now gone. Having enough capacity, we can spawn many new servers within minutes. Now there was a new problem/bottleneck, deployment of server services (e.g.: web, db, caching, monitoring, logging, application, etc.) and that, too, with absolute speed, consistency and reliability. Another challenge was that manual server configuration was a waste of time and ultimately money. Configuring the server manually takes a lot of time, and is error prone. If the system admins are busy firefighting (Incident Management), when are they going to focus on overall improvement of the infrastructure, i.e. problem management? Scripting, e.g.: Shell, Python, Perl, etc., was of help to a certain extent. However, there was a need to have a higher level of abstraction and tools that would allow admins to focus on what needs to be done, rather than how.

What is Chef?

Chef is a powerful automation platform that transforms infrastructure into code. It’s open source. Its DSL is written in Ruby. And it’s supported by almost all major Linux, Unix, Windows, cloud and container platforms. It’s released under Apache2 license. The project was started in 2008 and has been growing ever since. It has a large community and is backed by a commercial for profit company/entity called “Chef”. Chef is being used by startups and Fortune 100 companies, like Facebook, GE Capital and more.

Overview of Chef Architecture

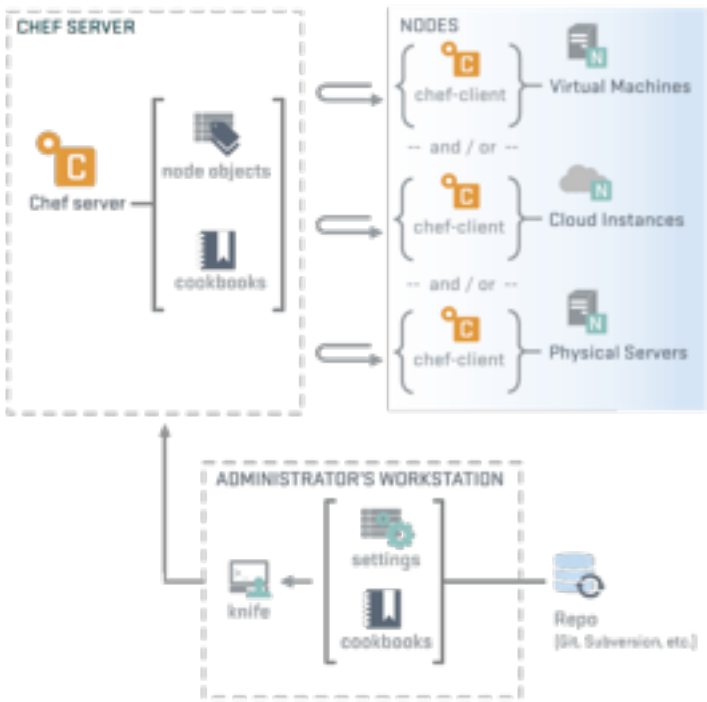


Figure 2. Overview of Chef Architecture

Let us take a look for the next Table. Now, you read the Table to see the Chef components.

Chef Workstation	A workstation is a computer configured to run various chef command line tools that interact with chef server and nodes. It’s also used to create and maintain cookbooks.
Chef Server	A chef server acts as a single point of contact for workstation as well as the nodes/client. All the code (cookbooks/recipes) are uploaded to chef server.
Chef node	A node is any machine - physical, virtual, cloud, etc. - that is managed by chef.
Chef client	Chef client is a software or agent that runs on chef nodes. It contacts chef server periodically and downloads the code (recipes, etc.) and executes them.

Chef Workstation Components

Chef repo. It’s a repository structure, used to create and maintain chef cookbooks. Cookbooks contains recipes, attributes, templates, etc. This repo could be managed by version control system such as Git.

Knife. It’s the command line tool to interact with chef server as well as nodes.

Chef DK. It’s a package that contains everything to get started with chef. Unfortunately, it’s not available for BSD as of now.

Chespec. It’s the unit test framework for chef.

Cookbook

A cookbook is a fundamental unit of configuration or policy. Following are the cookbook components:

Attributes	Attributes can be defined in files and can be overridden. They help in making the configuration dynamic.
Recipes	It’s the most fundamental configuration element. It’s authored in Ruby and it must be stored within a cookbook. It’s a collection of resources, and can be included, and can have dependencies.
Files	Files inside the files directory are copied to the target node, just like scp.
Libraries	It’s the arbitrary ruby code to be included in a cookbook.
Templates	A cookbook template is an embedded Ruby file that’s used to dynamically generate a static text file.



### Typical Chef cookbook Workflow.

- Create or Edit a cookbook.
- Upload cookbook.
- Provision machine.
- Bootstrap machine.
- Run Chef-client.
- Ssh and validate.

### Conclusion

The full course, DevOps with Chef on FreeBSD, is available on the BSD Magazine website. In this self-paced course, you'll cover lot of concepts and hands on practice. We'll focus on the core topics as mentioned in the agenda, and we assume that you already know the prerequisites. If not, you might have to learn additional topics (e.g.: Git, Ruby, FreeBSD) first.

The learning curve for Chef can get steep at times. So, just be patient and keep working on it and you'll eventually get it. We'll certainly help you throughout the course, but be prepared to do a lot of self-study as well. The best way to learn Chef is just to use it.

Without further delay, let's get started. Check the full agenda online:

<https://bsdmag.org/course/devops-chef-freebsd/>



#### About the Author

Arun Tomar is an Entrepreneur, Technology Evangelist, Solution Architect, Consultant & Corporate Trainer with deep expertise in designing and providing end to end solutions for enterprise IT Infrastructure requirements. He has more than 12 years of experience. He is currently the Director and CTO at AutomateHub Service, Inc., Canada.

# BSD Certification

**The BSD Certification Group Inc. (BSDCG) is a non-profit organization committed to creating and maintaining a global certification standard for system administration on BSD based operating systems.**

## ? WHAT CERTIFICATIONS ARE AVAILABLE?

**BSDA: Entry-level certification** suited for candidates with a general Unix background and at least six months of experience with BSD systems.

**BDSP: Advanced certification** for senior system administrators with at least three years of experience on BSD systems. Successful BDSP candidates are able to demonstrate strong to expert skills in BSD Unix system administration.

## ✓ WHERE CAN I GET CERTIFIED?

We're pleased to announce that after 7 months of negotiations and the work required to make the exam available in a computer based format, that the BSDA exam is now available at several hundred testing centers around the world. Paper based BSDA exams cost \$75 USD. Computer based BSDA exams cost \$150 USD. The price of the BDSP exams are yet to be determined.

Payments are made through our registration website:  
<https://register.bsdcertification.org/register/payment>

## i WHERE CAN I GET MORE INFORMATION?

More information and links to our mailing lists, LinkedIn groups, and Facebook group are available at our website:  
<http://www.bsdcertification.org>

Registration for upcoming exam events is available at our registration website:  
<https://register.bsdcertification.org/register/get-a-bsdcg-id>





# EMERGENCY CURING

for Windows workstations and servers  
including those running other anti-virus software



## FUNCTIONS:

- Cures Windows workstations and servers.
- Verifies the quality of the anti-virus software currently in use.

## FEATURES:

- Dr.Web CureIt! doesn't require installation and doesn't conflict with any known anti-virus; consequently there is no need to disable the anti-virus currently in use to check a system with Dr.Web CureIt!.
- Improved self-protection and an enhanced mode for more efficient countermeasures against Windows blockers.
- Dr.Web CureIt! is updated at least once an hour.
- The utility can be launched from removable media including USB storage devices.

## LICENSING FEATURES:

The utility is available for free when used for non-business purposes.



## OpenBSD 6.1 New Features

The recently released OpenBSD 6.1 includes major update, and this article will cover all of the significant changes and features of the new release. The OpenBSD 6.1 release will be the first to not provide an official CD media set for purchase. Instructions on how to create a bootable install media for OpenBSD 6.1 can be found at <https://www.openbsd.org/faq/faq4.html#MkInsMedia>

### New Platform Architecture Support

The is new release now provides functional support for ARM64 architecture and the following hardware platforms: PINE 64 (Allwinner A64), Raspberry Pi 3 (BCM2837) and the AMD Seattle. Additionally, another related hardware architecture platform is the armv7. For more details on how to install OpenBSD on the Raspberry PI 3, visit: <http://undeadly.org/cgi?action=article&sid=20170409123528>

For architecture specific installation details, please refer to the respective supported platform install instructions at: <https://www.openbsd.org/plat.html>

### OpenSSH Changes

It's important to mention that shipped version of OpenSSH 7.4 server has removed both SSH v1 protocol support and SSH client support for 3des-cbc. This might cause trouble for legacy ssh clients to connect via OpenSSH.

### Binary Patching

The syspatch(8) binary patching is now part of the base operating system and provides a utility to ease fetching, verifying, installing and reverting OpenBSD binary

patches on stable release for amd64 and i386 platforms only. A prerequisite for using syspatch(8) the /etc/installurl configuration file must be populated with a single line with the valid URL from OpenBSD FTP or HTTP mirrors: (<https://www.openbsd.org/ftp.html>).

```
# echo "# /etc/installurl" >
/etc/installurl

# echo
"https://ftp.openbsd.org/pub/OpenBSD" >>
/etc/installurl
```

Please note that /etc/installurl replaces /etc/pkg.conf(5). If you would like to specify more than one repository path, you must set PKG\_PATH environment variable (see Table 1).

Syspatch Usage	Command Description
# syspatch	Save the original release kernel, create a rollback tarball containing the files it is about to be replaced, apply all missing patches and then installing all files
# syspatch -c	List available patches
# syspatch -l	List installed patches
# syspatch -r	Revert recently installed patches

**Table 1. Syspatch usage**

After invocation of the syspatch command, a copy of the original binaries will be stored under /var/syspatch/\* . Additionally, the original kernel will be backed up and stored /bsd.syspatch\${OSrev}



# Let's Encrypt SSL/TLS Certificate Management with the Automatic Certificate Management Environment (ACME)

This feature provides a native support and a replacement for the deprecated port security/letsencrypt. The acme-client allows for automated deployment and management of Let's Encrypt certificates for one or more domains. Additionally, the acme-client will refresh the signature for an existing certificate prior to 30 days before expiry.

The full complete certificate file is located at /etc/ssl/acme/fullchain.pem and a related private key file is located at /etc/ssl/acme/private/privkey.pem/

For example, to initially configure and use acme-client for a domain running under port 80, the following commands must be executed:

- Create a new RSA account and domain key if it does not exist: # acme-client -ADv tld.com
- Setup the webserver for the challenge response process of the Let's Encrypt service.

The directory which contains the challenge is /var/www/acme , and can be served by httpd(8) and configuring /etc/httpd.conf

```
server "tld.com" {  
  
    root "/empty"  
  
    listen on * port 80  
  
    location  
    "/.well-known/acme-challenge/*" {  
  
        root "/acme"  
  
        root strip 2  
  
    }  
  
}
```

As root under the /.well-known/acme-challenge , type the following commands to generate the certificates:

```
# acme-client -vD tld.com www.tld.com
```

The above command is equivalent to the manual process of managing certificate authorities, which is comprised of the following steps:

- Generating the webserver private and public keys.
- Submitting your public key to the certificate authority.
- Providing proof of authorization to the certificate authority that you have the certificate for the domains you are requesting.
- Retrieving and installing the signed certificate.

Create and submit a new key for a single domain with a configured webserver to use challenge directory	# acme-client -vD tld.com
Renew the certificate (or can be added as cron job for automated renewal)	# acme-client tld.com www.tld.com # rcctl reload httpd

## Native Virtual Machine Support

Furthermore, this release provides a native virtual machine hyper-visor and has support for VMs with memory greater than 2GB of RAM using the vmm(4)/vmctl(8) tools and vmd(8) daemon. The vmd(8) daemon is responsible for the execution, provides VM provisioning by configuring the VM, virtual CPUs and related device layer configurations and control of the virtual machine, and is executed upon boot when the vmd\_flags="" has been enabled in the /etc/rc.conf.local configuration file. The management of the vmd(8) and vmctl(8) provides management for the VMs. OpenBSD's virtual machine fully supports privilege separation using the pledge(2) system call policy interface.

All aspects of VM configuration are defined within the global /etc/vm.conf file. Additional configurations can be loaded and specified by using the include "/etc/vm.tld.com.conf"

To setup and run an OpenBSD requires the following configuration changes to be made:

```
/etc/hostname.vether0  
  
inter 10.0.1.1 255.255.255.0 NONE
```

```

/etc/vm.conf

switch "local" {

    add vether0

    add tap0

}

vm "vm0.vm" {

    memory 1024M

    kernel "/bsd.rd"

    disk "/vmm/vm0.img"

    interface {

        switch "local"

        lladdr 00:11:22:33:44:55

    }

}

```

## VM Network Configuration

For example, we will setup a virtual machine network with the following configuration:

**Domain name: tld.com**

**IP Address: 10.0.1.1   Subnet mask: 255.255.255.0**

**Name server: 10.0.1.1**

**Default router: 10.0.1.1**

**Valid Address Range: 10.0.1.1 - 10.0.1.254**

**Subnet ID: 10.0.1.0**

**Broadcast Address: 10.0.1.255**

```

/etc/dhcd.conf

shared-network VMM-tld.com {

    subnet 10.0.1.0 netmask 255.255.255.0

{

    range 10.0.1.1 10.0.1.254;

```

```

        option subnet-mask
255.255.255.0;

        option broadcast-address
10.0.1.255;

        option routers 10.0.1.1;

        option domain-name-servers:
10.0.1.1

    }

}

/etc/sysctl.conf

net.inet.ip.forwarding=1

/etc/pf.conf

set skip on lo

block return      # block stateless traffic

pass              # establish keep-state

# By default, do not permit remote
connections to X11

block return in on ! lo0 proto tcp to port
6000:6010

ext_if="em0"

int_if="{ vether0 tap0 }"

set block-policy drop

set loginterface egress

match in all scrub (no-df random-id
max-mss 1440)

match out on egress inet from
!(egress:network) to any nat-to (egress:0)

pass out quick inet

pass in on $int_if inet

/etc/rc.conf.local

dhcpd_flags=vether0

```

```
vmd_flags=
```

To start this example, VMM executes the following commands:

- This sets up the VMM console output redirection

```
# vmctl console 1

# cu /dev/tty0
```

- This creates the virtual machine disk image and starts the virtual machine

```
# vmctl create /vmm/vm0.img -s 1024M

# vmctl start -c -k /bsd.rd -m 1024M -i 1 -d /vmm/vm0.img
```

VMM Usage Reference

To convert an existing Public Cloud Images *.img to VMM compatible raw image extension (OpenStack, LXDC, etc. )	Must install qemu using ports or packages # qemu-img convert cloudimg-amd64.img cloudimg-amd64.raw
Start the converted Cloud Image VM using VMM with NAT and starting the console	# vmctl start cloudimage -d cloudimg-amd64.raw -n nat -c
Starting the VMM	# vmctl start vmlabel
Stopping the VMM	# vmctl stop vmlabel
VMM Status	# vmctl status
Create a disk image with specific size	# vmctl create disk -s 3.0G
Create a new VM with a specified memory size	# vmctl start "vmlabel" -m 4096M -i -d disk.img -k /bsd -c

Please refer to the main pages:

vmctl(8) <http://man.openbsd.org/OpenBSD-6.1/vmctl>,  
vm.conf(5) <http://man.openbsd.org/OpenBSD-6.1/vm.conf.5> ,  
and the OpenBSD FAQ "Networking vmm guests"  
<https://www.openbsd.org/faq/faq6.html#VMMnet> for  
additional commands and configuration details.

VM Network Switch Configuration

The OpenBSD VMM allows for the virtual machines (VM) to communicate with other network interfaces on the host system by using the bridge(4) or switch(4). These interfaces for the virtual switch can be configured using the ifconfig(8) command or the hostname.if(5) configuration file option. Upon start up of the VM, all interfaces which have been assigned to the virtual switch will have the tap(4) network interfaces to be automatically created and assigned to the respective bridge(4) or switch(4) mapped virtual switch interfaces.

Software Defined Networking on OpenBSD: switchd(8) and switch(4)

The 6.1 will provide native built-in support for software defined networking (SDN) in implementing the OpenFlow\* (\*OpenFlow has a Trademark) protocol version 1.3.5.

OpenFlow protocol allows for the direct access to the data/user plane, a component of a network switch or router architecture which determines how forwarding decisions are being made. The ability to administrator, program, manipulate control and modify packet forwarding and routing capabilities at the OSI layer 3. The decoupling of the forwarding and network control functions defines the core functionality of software-defined networking architecture. The OpenFlow protocol operates using TCP and is between the switch and controller as shown in Figure 1.

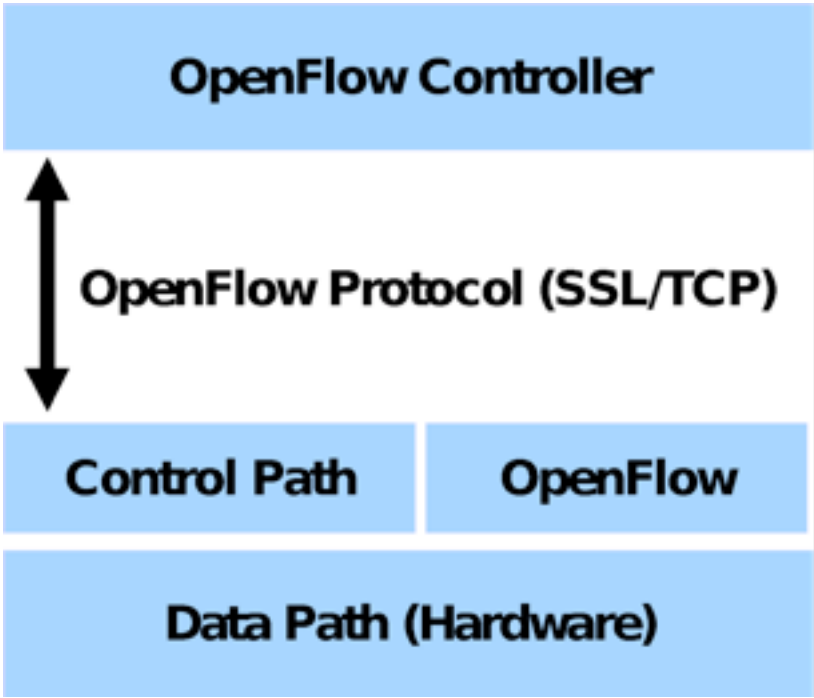


Figure 1. The OpenFlow protocol operates using TCP and is between the switch and controller



Switchd(8) functions as a proxy between the OpenFlow controller and the switch(4) comprised of the following two components:

1. The OpenFlow controller/forwarder or vswitch: switchd(8) and switchctl(8). The switchd(8) daemon functions like a local controller and operates at the user level.
2. Kernel bridge switch(4) driver and user-kernel interface /dev/switch\*. The switch(4) kernel driver is partially based upon OpenBSD's bridge(4).

This feature is implemented using a client and server model whereas the user level controller, daemon and forwarder switch(8). Later on, we will provide a tutorial of how to use the virtual switch:

- A mandatory prerequisite for using the virtual switch is to create a virtual Ethernet interface as a permanent host interface as shown previously in VMM usage,

```
/etc/hostname.vether0
```

```
inter 10.0.1.1 255.255.255.0 NONE
```

- A virtual switch interface device for use by switchd(8) or switchctl(8) must be created using the following commands:

```
1. switchd interface:      # ifconfig switch0
                           create
```

```
2. local mode switch interface: # ifconfig switch0
                                addlocal vether0
```

3. persistent switch interface startup at boot:

```
# cat /etc/hostname.switch0
```

```
up
```

```
!rcctl -f start switchd
```

```
!switchctl connect /dev/switch0
```

```
!ifconfig switch0 addlocal vether0
```

```
!ifconfig vether0 up
```

or manually started

```
# /etc/rc.d/switchd
```

```
# switchctl connect /dev/switch0
```

After creating the switch interface, a corresponding OpenFlow channel of the interface will be created under /dev/switch0. This stream device reads the OpenFlow message headers first then the entire message contents subsequently.

Switchctl(8) is used to control switchd(8) daemon which allows for controlling how the daemon interfaces with an address or switch(4) device. An alternate method of configuring switchd(8) daemon is to set default global configuration values from /etc/switchd.conf. To configure and set the listener address and port to accept connections from remote OpenFlow switches over a regular OpenFlow protocol or wrapped using tls will require the following options:

```
/etc/switchd.conf
```

```
listen on 0.0.0.0 port 6655
```

or by accepting a remote switch tls OpenFlow connection

```
/etc/switchd.conf
```

```
listen on 0.0.0.0 tls port 6655
```

The switchd(8) configuration file can be checked for errors by: /etc/rc.d/switchd -n -v

For additional details please refer to the man pages for additional details:

switchctl(8)

<http://man.openbsd.org/OpenBSD-6.1/switchctl.8> ,

switchd(8)

<http://man.openbsd.org/OpenBSD-6.1/switchd.8>

switchd.conf(5)

<http://man.openbsd.org/OpenBSD-6.1/switchd.conf.5>

## Conclusion

We've covered and highlighted several key innovative features in this new 6.1 release of OpenBSD.

### About the Author

Albert Hui has been passionate about Unix and other exotic operating systems. He has also been an OpenBSD enthusiast since 2003. To contact the author: [alberthui3@yahoo.com](mailto:alberthui3@yahoo.com)

# Conventions over Restrictions – Programming the Python Way

BY MIKE MÜLLER

Python is a powerful programming language. It supports procedural and object-oriented programming, and provides important features for functional programming. Its dynamic typing and many meta-programming features allow doing nearly everything at run time. While this freedom provides lots of opportunities for elegant solutions, Python might also be perceived as a dangerously unrestricted language. The common solution is to use conventions to solve a problem in the preferred, hence, “pythonic” way. Experienced Python programmers tend to stick to these conventions as much as possible and only diverge when the benefits are substantial compared to that of the conventional solution.

This article gives a short overview of Python features focusing on conventions and their benefits. Since Python syntax is often called executable pseudo code, a reader with solid programming experience does not need to have previous knowledge of Python to follow along.

**FREE READING \* SDJ ARTICLES \* SDJOURNAL.ORG**

[WWW.SDJOURNAL.ORG](http://WWW.SDJOURNAL.ORG)

sdjournal

SDJournal (Software Developer's Journal ISSN 1734-3933) is the monthly magazine for professional programmers. It is a technical magazine, providing articles of interest to all people who want to be an expert in the programming field. It includes highly technical articles, and non-technical articles on software development, as well as news on innovations in the field.

Our collection of articles are packed with technical tips and useful tricks to make you a better programmer.

To maintain the highest editorial standards, SDJournal is written and edited by software developers, engineers, and experts.

## Nearly Online Zpool Switching Between Two FreeBSD Machines

ZFS pools are designed to be used by a single machine at any particular point in time. When we want to access a zpool from another system, we have to export and import it to the new server. This disadvantage makes simultaneous direct usage of the pool quite impossible for reliable dual-controller storage systems.

Earlier, with the articles related to the BeaST storage system concept, I have posted several solutions on how to bypass this architectural obstacle and make failover to another storage controller. But these hints turned out dangerous to the pools. Testing has shown that in most cases, it is impossible to make fail-back when the controller boots again. Even worse, zpool often becomes permanently damaged.

Luckily, it forced me to continue my studies and finally, I have discovered a plain way to make a safer online or, honestly saying, near-online failover/failback operations for ZFS storage pools.

And yes, it now looks simpler compared to what I have seen before. Also if everything goes well, this method with several essential improvements regarding ZIL mirroring, will be implemented in the BeaST storage system for ZFS.

Such operations became possible due to CTL HA implementation, which controls access to the LUNs and prevents simultaneous attempts to use zpool at any moment, including the time of switching controllers.

Now, let's call Ockham to cut off all unnecessary stuff from the configuration and leave only the essence of the idea.

I have built a lab similar to what I normally use for the BeaST storage experiments. It is an Oracle VM VirtualBox environment with one virtual machine, clnt-1, for a client host and two virtual servers for storage controllers: ctrl-a and ctrl-b, respectively.

For the sake of simplicity on storage machines, I use only three **shareable, fixed-sized** (see Oracle VM VirtualBox documentation for the virtual storage:

<https://www.virtualbox.org/manual/ch05.html>) virtual drives: ada1/ada2 connected with SATA virtual controller for client data, and da0 on SAS controller for ZFS Intent Log. FreeBSD 11.0 Release is installed on ada0 – a **normal** from the VirtualBox point of view **dynamically allocated** virtual drive on all of these three machines.

Two networks are configured: 192.168.55.0/24 for client access and 192.168.56.0/24 for cross-controller data path and system communications.



The dual-controller configuration

First of all, let's configure the basics of a FreeBSD system. The /etc/rc.conf on controllers is elementary:

ctrl-a	ctrl-b
<pre>hostname="ctrl-a"  # Cross-controller link ifconfig_em0="inet 192.168.56.103 netmask 0xffffffff00" # Client network ifconfig_em1="inet 192.168.55.103 netmask 0xffffffff00"  # VirtualBox guest additions vboxguest_enable="YES" vboxservice_enable="YES"</pre>	<pre>hostname="ctrl-b"  # Cross-controller link ifconfig_em0="inet 192.168.56.104 netmask 0xffffffff00" # Client network ifconfig_em1="inet 192.168.55.104 netmask 0xffffffff00"  # VirtualBox guest additions vboxguest_enable="YES" vboxservice_enable="YES"</pre>

Then, edit /boot/loader.conf to configure boot time parameters of CTL HA:

ctrl-a	ctrl-b
<pre>ctl_load="YES"  kern.cam.ctl.ha_id=1 kern.cam.ctl.ha_mode=2 kern.cam.ctl.ha_role=0</pre>	<pre>ctl_load="YES"  kern.cam.ctl.ha_id=2 kern.cam.ctl.ha_mode=2 kern.cam.ctl.ha_role=1</pre>

And reboot both controllers:

```
# reboot
```

You can see, I have set "kern.cam.ctl.ha\_mode=2" which means that secondary CTL HA node accepts all requests and data to the LUN but then forwards everything to the primary node, so it works in ALUA (Asymmetric Logical Unit Access) mode. Full active-active access "kern.cam.ctl.ha\_mode=1" may be dangerous especially when dealing with our dual-controller architecture and ZFS.

Now, we can create a zpool (beast) with separated ZIL (da0) and a volume (v0) on the ctrl-a machine. The appropriate task on ctrl-b controller is only to import the "beast" pool. The options, "-m none" on ctrl-a and "-N" on ctrl-b prevent pool mounting:

ctrl-a	ctrl-b
<pre>zpool create -m none beast mirror /dev/ada1 /dev/ada2 zpool add beast log /dev/da0 zfs create -V 100M beast/v0 zfs set sync=always beast</pre>	<pre>zpool import -N beast</pre>

The "zfs set sync=always beast" option is needed to push all transactions through ZIL device as we do not want to lose client data in the RAM memory of the dead controller in case of failure.

When talking about production systems, ZIL device must be secure and as fast as possible. Ideally, it should be a non-volatile cache memory (connected to- or mirrored between- both controllers) or shared by controllers' fast NVMe, SSD drive at least. The BeaST storage system has a ZIL mirroring function for that case, but for simplicity of our small Virtual Box lab, we created a shared SAS drive for that ZIL device.

Now, setup CTL HA interlink. Run:

ctrl-a	ctrl-b
<pre>sysctl kern.cam.ctl.ha_peer="listen 192.168.56.103:7777"</pre>	<pre>sysctl kern.cam.ctl.ha_peer="connect 192.168.56.103:7777"</pre>

Check if sysctl kern.cam.ctl.ha\_link value equals 2. It means that the link is established:

```
# sysctl kern.cam.ctl.ha_link

kern.cam.ctl.ha_link: 2
```

Prepare a simple /etc/ctl.conf configuration for a single LUN definition with our ZFS volume:

ctrl-a	ctrl-b
<pre>portal-group pg0 {     discovery- auth-group no- authentication     listen 192.168.55.103 }  target ign. 2016-01.local.beast:t arget0 {     auth-group no- authentication     portal-group pg0      lun 0 {         path /dev/ zvol/beast/v0     } }</pre>	<pre>portal-group pg0 {     discovery- auth-group no- authentication     listen 192.168.55.104 }  target ign. 2016-01.local.beast:t arget0 {     auth-group no-authentication     portal-group pg0      lun 0 {         path /dev/zvol/beast/ v0     } }</pre>

Finally, start ctld daemon:

ctrl-a	ctrl-b
<pre>service ctld onestart</pre>	<pre>service ctld onestart</pre>

## The Client

This part is the least interesting one to create and to read. Thus, I posted obvious configuration files and commands here.

Update /etc/rc.conf on the client:

```
hostname="cln-1"

ifconfig_em0="inet 192.168.55.111 netmask
0xffffffff00"

# Public LAN

# VirtualBox guest additions

vboxguest_enable="YES"

vboxservice_enable="YES"
```

```
# iSCSI

iscsid_enable="YES"

# Initiators

Set this kernel variable so that the client is able to lose
iSCSI paths. Contents of /etc/sysctl.conf are:

kern.iscsi.fail_on_disconnection=1

Now, connect the controllers by running:

# iscsictl -A -p 192.168.55.103 -t
ign.2016-01.local.beast:target0

# iscsictl -A -p 192.168.55.104 -t
ign.2016-01.local.beast:target0

Then, create a multipathing device to access the same
LUN from both controllers:

# gmultipath label beast /dev/da0 /dev/da1

The command above sets an active-passive mode on the
multipath device. For testing purposes, you may want to
create it in active-active:

# gmultipath label -A beast /dev/da0
/dev/da1

From the client's point of view, all paths are active.
However, the path for ctrl-b will respond slowly since
CTL HA forwards all data to ctrl-a and on production
systems. Therefore, active-active mode may have a
negative effect on client's performance.

Create and mount a filesystem:

# newfs /dev/multipath/beast

# mount /dev/multipath/beast /mnt

And lastly, put some load on it:

# dd if=/dev/random of=data.rnd bs=1M
count=90

# sh -c "while true; do cp data.rnd /mnt;
md5 /mnt/data.rnd; done"

Now, we can consider the system ready for
failover/failback tests.
```

## Fail There and Back Again

In the BeaST storage system, we use simple BeaST Quorum (BQ) software to detect controller failure and react accordingly. But today, we will do everything by our hands just to see what is going on during our experiments.

Let's go! Switch off the power of ctrl-a controller and immediately, run to console of ctrl-b and type:

```
# zpool export beast && zpool import -f -N  
beast && sysctl kern.camctl.ha_role=0
```

These commands will reattach the “beast” pool to ctrl-b and make this controller primary for CTL HA operations.

The client has to wait for these operations to complete. And in real life, it may take quite a long period! It mostly depends on the pool size and volumes, workload and other options, so be careful with it and check the timeout settings. Nonetheless, we should not worry about our tiny configuration as it will be fast for sure.

Nevertheless, if everything is done properly, client will continue to work with the volume and pool through the ctrl-b controller.

To restore ctrl-a, boot it and run:

```
# sysctl kern.camctl.ha_role=1  
  
# zpool status  
  
# sysctl kern.camctl.ha_peer="listen  
192.168.56.103:7777"  
  
# service ctld onestart
```

It makes ctrl-a to start operating and to be the secondary member of the CTL HA cluster, so the controller will serve requests by forwarding data to the ctrl-b controller.

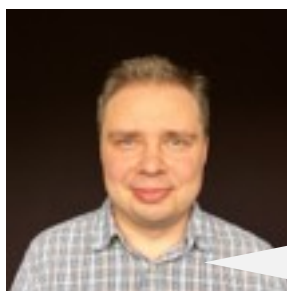
The same procedure applies to the ctrl-b controller if we decide to fail it. We can have fun crushing controllers, one after another, all day long, but it's better to use the BeaST Quorum or some other more advanced quorum software to automate the recovering process. Details on how-to install and configure the BeaST Quorum are fully described in the BSD Magazine vol 10 no 13, issue 12/2016 (89).

## Conclusion

Although this experiment looks like quite a rough example of the fail-over/fail-back procedures for ZFS data pools, it is actually a breakthrough for our storage system development.

After several enhancements to support reliability and workload balancing, we will use two storage pools (one pool per controller) in conjunction with the BeaST Quorum to automate zpool switching between both active controllers.

**Now, cross your fingers for good luck, as I am one step from implementing it in the BeaST storage system.**



### About the Author

Mikhail E. Zakharov is an honored SAN/storage IBMer. He has 10 years of experience in large SAN and storage environments: mainly Hitachi, HP and Brocade. Empty – expect-like tool author and a FreeBSD enthusiast.

Contact the author at:  
[zmey20000@yahoo.com](mailto:zmey20000@yahoo.com)



## GUI Programming in FreeBSD with Perl/Tk

### What is Perl?

Perl officially stands for Practical Extraction and Report Language. Perl was originally developed by Larry Wall in 1987 as a general-purpose Unix scripting language to make report processing easier. It was optimized for scanning arbitrary text files, extracting information from those text files, and printing reports based on that information. It's widely used for everything, from quick "one-liners" to full-scale application development. Its general-purpose programming facilities support procedural, functional and object-oriented programming paradigms, making Perl a comfortable language for any purpose.

### What is CPAN?

The Comprehensive Perl Archive Network (CPAN) is a repository of over 250,000 software modules and accompanying documentation for 39,000 distributions, written in the Perl programming language by over 12,000 contributors. You can easily add a new library to Perl with CPAN that's going to make programming easier.

### How to deal with CPAN in FreeBSD 11?

Cpanminus is Perl extension to get, unpack, build, and install modules from CPAN. To install cpanminus, issue these commands:

```
#pkg install p5-App-cpanminus
#rehash
```

After that, you can install CPAN module with cpanm command.

### What is TK?

The Perl/Tk module, also known as pTk or ptk, is a Perl module designed to create widgets and other commonly

used graphical objects to form a graphical user interface (GUI). Using the module to create a GUI enhances the look and feel of a program and helps the end user navigate through the program and its functions. One major advantage of using the Perl/Tk module is that the resulting application can be cross-platform, meaning the same GUI application can be used on UNIX®, Linux®, Macintosh, Microsoft® Windows®, or any other operating system that has Perl and the Perl/Tk module installed.

### Install Tk On FreeBSD

The installation process is very straightforward and fast, and cpanm will take care of everything. Just issue this command: `#cpanm Tk` and Tk is installed successfully.

### Create First GUI Application

It's time to write our first gui program in Perl/Tk or PTK. Create a file named ptk.pl, and enter the following text in the file:

```
#!/usr/bin/perl -w

# this is bsdmag ptk tutorial

use Tk;

use strict;

my $mw = MainWindow->new;

$mw->geometry("200x100");

$mw->title("bsdmag!!!");

$mw->Label(-text => 'bsdmag')->pack();

$mw->Button(-text => "Exit", -command
=>sub{exit})->pack();

MainLoop;
```



Let's dig into the code, line by line:

```
#!/usr/bin/perl -w
```

The first part ( /usr/bin/perl ) defines the location where the Perl executable resides on the

computer. The second part of this line ( -w ) enables warnings when executing the script, informing the end user of any possible errors found.

Comments and text that shouldn't be evaluated at execution are preceded with an octothorpe or # symbol:

```
# this is bsdmag ptk tutorial
```

For a Perl script to use the Tk module, it must be included—in other words, use Tk. Also, adding the use strict statement to a Perl script helps find any possible typos or logic errors:

```
use Tk;
```

```
use strict;
```

MainWindow: creates the primary window of the application and assigns it to \$mw . \$mw acts as the parent to all other widgets:

```
my $mw = MainWindow->new;
```

Set the main window size to 200 x 100 and title the window "bsdmag!!!:"

```
$mw->geometry("200x100");
```

```
$mw->title("bsdmag!!!");
```

Create a label inside the main window with the caption *bsdmag*. Finally pack function, which is the geometry manager. It's used on widgets to calculate the space allocated on the widget's parent; it also displays the widget:

```
$mw->Label(-text => 'bsdmag')->pack();
```

This line creates an Exit button inside the main window that exits the Perl script:

```
$mw->Button(-text => "Exit", -command  
=>sub{exit})->pack();
```

Because this button takes up space on the main window, you need to use the pack function to calculate the space used.

When MainLoop is called, all functions and data read prior to that point are executed, and the GUI is displayed.

**MainLoop;**

**Widgets**

There are many widgets out there but we will only cover entry, buttons and labels.

## What is a widget?

A widget is a graphical object that performs a specific function. Any graphical object in the Perl/Tk module can be considered a widget. When you think of a GUI application, the buttons, entry, frames, and scrollbars are all widgets.

## Entry

The entry widget lets you enter one line text input.

```
#!/usr/bin/perl -w
```

```
use Tk;
```

```
use strict;
```

```
my $mw = MainWindow->new;
```

```
$mw->geometry("200x100");
```

```
$mw->title("Entry Test");
```

```
$mw->Entry(-background => 'white',  
-foreground => 'black')->pack(-side =>  
"top");
```

```
MainLoop;
```



All the elements described before except for this line:

```
$mw->Entry(-background => 'white',  
-foreground => 'black')->pack(-side =>  
"top")
```

This line creates an entry with a white background and black foreground at the top of window.

## Label

A *label* is a non-editable text widget. Labels can be useful prior to entry boxes. Enter the following example script:

```
#!/usr/bin/perl -w

use Tk;

use strict;

my $mw = MainWindow->new;

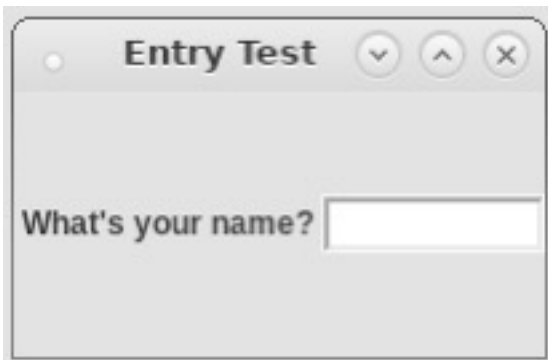
$mw->geometry("200x100");

$mw->title("Entry Test");

$mw->Label(-text => "What's your
name?")->pack(-side => "left");

$mw->Entry(-background => 'black',
-foreground => 'white')->pack(-side =>
"right");

MainLoop;
```



The first line creates the label with the text, *What's your name?*:

```
$mw->Label(-text => "What's your
name?")->pack(-side => "left");

$mw->Entry(-background => 'white',
-foreground => 'black')->pack(-side =>
"right");
```

To position the label to the left of the entry widget, you set pack to align to the left side. As before, the background is set to white and foreground set to black, and pack will create everything just like before. Entry widget can only show one line and pressing enter can't create a new line.

## Button

Buttons are the most important element of event-driven programming. You can execute functions or commands

when activated by the end user. The next example uses two buttons. The first, labeled BSDMag, displays the message "This is BSDMag" with an OK button. The second button, labeled close, displays the message "Do you want to exit?". If the user clicks on Yes, the application will close everything, and if No, it will return the user to the main window. Here's the example script:

```
#!/usr/bin/perl -w

use Tk;

use strict;

my $mw = MainWindow->new;

$mw->geometry("200x200");

$mw->title("Button Test");

my $button1 = $mw->Button(-text =>
"BSDMag", -command =>
\&button1_sub)->pack();

my $button2 = $mw->Button(-text =>
"Close", -command =>
\&button2_sub)->pack();

sub button1_sub {

    $mw->messageBox(-message => "This is
BSDMag", -type => "ok");

}

sub button2_sub {

    my $yesno_button =
$mw->messageBox(-message => "Do you want
to exit?",

-type => "yesno", -icon => "question");

    if ($yesno_button eq "Yes") {

        exit;

    }

}

MainLoop;
```



We can split this script to 2 main sections.

- Creating Button widget.
- Creating subroutine.

The codes below create two buttons. Text value with a display caption and a command value which binds on-click subroutine. Pack will create widgets.

```
my $button1 = $mw->Button(-text =>
"BSDMag", -command =>
\&button1_sub)->pack();
```

```
my $button2 = $mw->Button(-text =>
"Close", -command =>
\&button2_sub)->pack();
```

When \$button1 is clicked on, the subroutine button1\_sub is executed. Inside this function, a message box is created that displays the text, *This is BSDMag*, with a single button labeled OK. Because no further evaluations are performed on the OK button, the message box is destroyed and the main window regains focus.

```
sub button1_sub {
    $mw->messageBox(-message => "This is
BSDMag", -type => "ok");
}
```



This subroutine shows a message box that displays the text “Do you want to exit?”. If the user clicks on “Yes” button, the program will close everything. Type of message box with “yesno” means only two buttons will be displayed, and the icon will be a “question” mark.

The return value of what the user clicked will be stored in \$yesno\_button variable and \$yesno\_button eq "Yes" to compare it to Yes value. If this comparison is true, then the program will exit.

```
sub button2_sub {
    my $yesno_button =
$mw->messageBox(-message => "Do you want
to exit?",
```

```
-type => "yesno", -icon => "question");

    if ($yesno_button eq "Yes") {
        exit;
    }
}
```



## Conclusion

There are many solutions to GUI programming in FreeBSD, like JavaFX, QtCreator or U++. However, using Perl and Tk(PTk) is preferred for its low dependency, fast and geek-style. Sometimes it's better to getting our hands dirty instead of debugging code constantly.

## Useful Links

<http://www.perl.com/pub/1999/10/perlTk/>

<http://meetbsd.ir>

<http://in4bsd.com>



### About The Author

Abdorrahman Homaei has been working as a software developer since 2000. He has used FreeBSD for more than ten years. He became involved with the meetBSD dot ir and performed serious training on FreeBSD. He is starting his company in Feb 2017. You can visit his site to view his CV: <http://in4bsd.com>



## Become FreeBSD User: Find Useful Tools

If you're usually programming on Linux and you consider a potential switch to FreeBSD, this article will give you an overview of the possibilities.

### How to Install the Dependencies

FreeBSD comes with either applications from binary packages or compiled from sources (ports). They are arranged according to software types (programming languages mainly in lang (or java specifically for Java), libraries in devel, web servers in www ...) and the main tool for modern FreeBSD versions is pkg, similar to Debian apt tools suite. Hence, most of the time if you are looking for a specific application/library, simply

```
pkg search <name>
```

without necessarily knowing the fully qualified name of the package. It is somehow sufficient. For example pkg

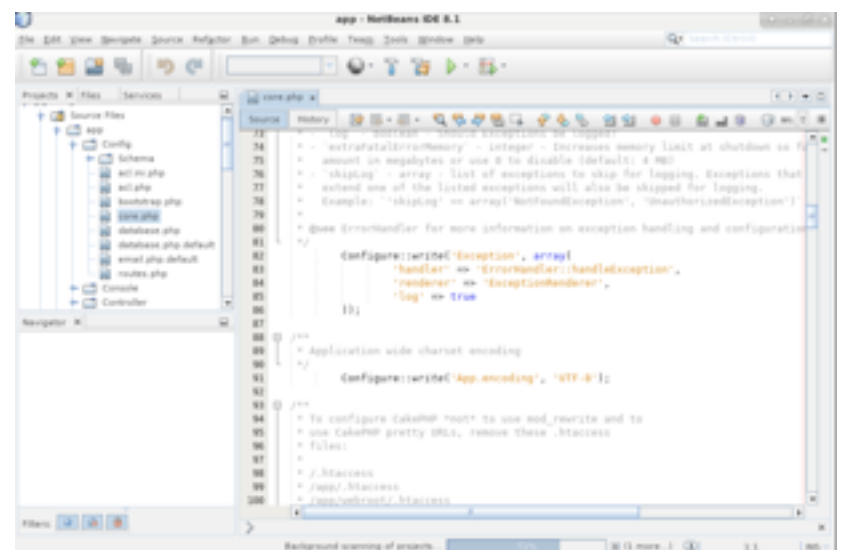
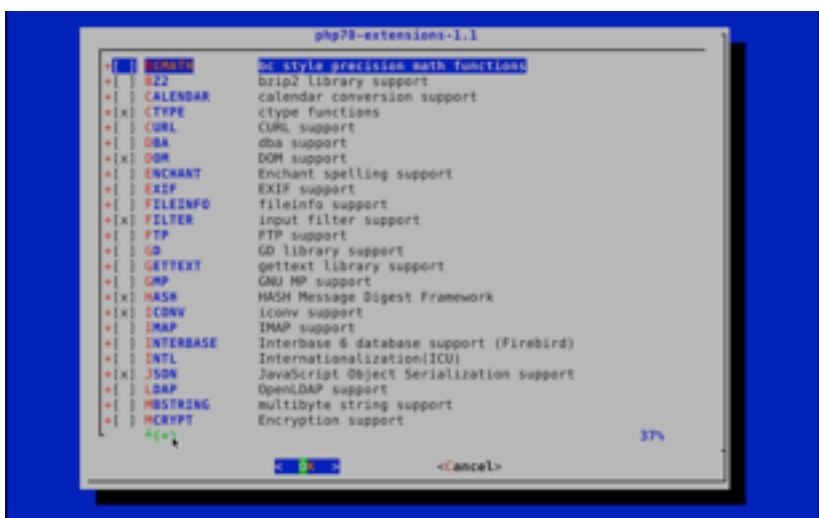
search php7 will display php7 itself and the modules. Furthermore, php70 specific version and so on.

The main difference is, you are not forced to either choose the binary or the port but can have both if it suits your need. Nonetheless, bear in mind that compiling from the source can take a certain amount of time to achieve if it's an important point for you. If the ports tree is not already present on your server, portsnap fetch extract will fetch the ports tree for you by default in /usr/ports. Then, related to the software type described above, you need to go to the related folder. For example, for installing php7:

```
cd /usr/ports/lang/php70
```

```
make config-recursive
```

```
make install clean
```



The second command, depending on which options you are going to choose, will display all the options available for each dependency (for example, if gd support is enabled, the options for graphics/gd library will appear).

However, most of the time the binary packages are sufficient to cover most of the needs.

## Web Development

Basically, this is the easiest area to migrate to. Most Web languages do not use specific platform features. Thus, most of the time, your existing projects might just be “drop-in” use cases.

If your language of choice is PHP, you are lucky as this scripting language is workable on various operating systems, on most Unixes and Windows. In the case of FreeBSD, you have even many different ports or binary package versions (5.6 to 7.1). In this case, you may need some specific PHP modules enabled, luckily they are available atomically, or if the port is the way you chose, it is via the [www/php70-extensions's](http://www.php70-extensions's) one.

Of course developing with Apache (both 2.2 and 2.4 series are available, respectively [www/apache22](http://www/apache22) and [www/apache24](http://www/apache24) packages), or even better with Nginx (the last stable or the latest development versions could be used, respectively [www/nginx](http://www/nginx) and [www/nginx-devel](http://www/nginx-devel) packages) via php-fpm is possible.

Besides PHP, the same applies for Python / Django ([www/py-django](http://www/py-django)) and Ruby on Rails ([www/rubygen-rails](http://www/rubygen-rails)),

Golang 1.8.1, Python 2.7 and 3.6 ([lang/python<version>](#)) are available as Ruby until 2.4 ([lang/ruby<version>](#)).

In terms of databases, we have the regular RDMBS like MySQL and PostgreSQL (client and server are distinct packages ...

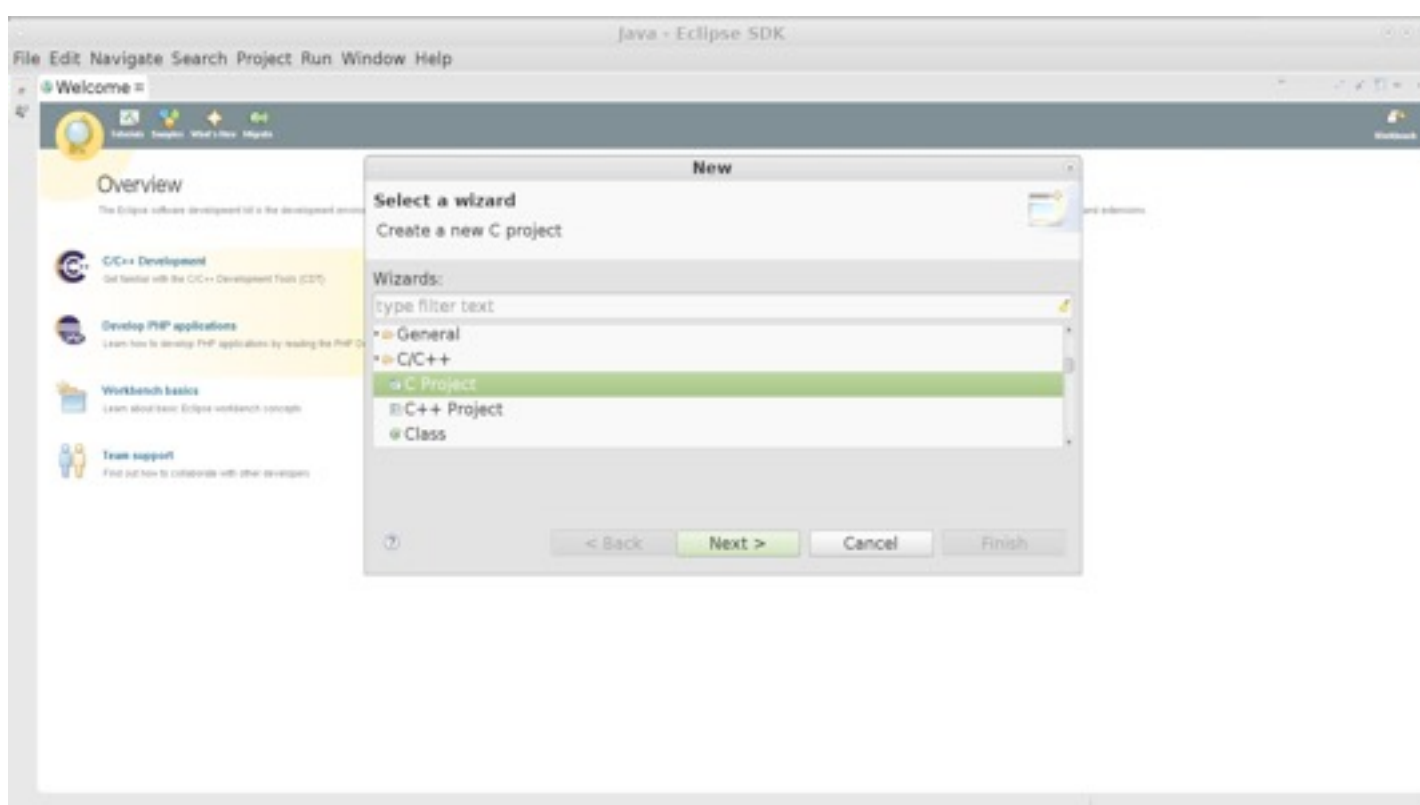
[databases/\(mysql/postgresql\)<version>-client](#), and [databases/\(mysql/postgresql\)<version>-server](#)).

Additionally, a more modern concept of NoSQL with CouchDB, for example ([databases/couchdb](#)), MongoDB ([databases/mongodb](#)), and Cassandra ([databases/cassandra](#)), to name but a few.

Also, if you need to perform efficient Map / Reduce for Big Data work; you can have either the well-known Apache Hadoop or Apache Spark (respectively [devel/hadoop](#) and [devel/spark](#)). Lastly, in case you ever need any search engine, Apache Solr/Lucene ([textproc/apache-\(solr/lucene\)](#)), Xapian ([databases/xapian](#)) and their various language bindings are available.

Is it only Java Web or any language which is based on the Java VM platform? In FreeBSD, Java 8 (either [java/openjdk8](#) or [java/linux-oracle-jdk18](#)), various popular frameworks and J2EE servers, servlet engines like Spring ([java/springframework](#)), Jboss ([java/jboss<version>](#)), Tomcat ([www/tomcat<version>](#)), Jetty ([www/jetty](#)), etc. are available. Even the more modern languages like Scala ([lang/scala](#)), Groovy ([lang/groovy](#)) can be found.

Two languages described above, Python and Ruby, have their Java VM counterparts, Jython ([lang/jython](#)) and Jruby ([lang/jruby](#)), available as well,



In terms of Integrated Development Environment, there are still several choices. For instance, the venerable Netbeans (java/netbeans or java/netbeans-devel) and Eclipse (java/eclipse ... side note, FreeBSD needs to have Kerberos support enabled, NO\_KERBEROS is /etc/make.conf or /etc/src.conf presence needs to be checked) with their numerous popular plugins.

## Low-level Development

The BSD are shipped with C and C++ compilers in the base. In the case of FreeBSD 11.0, it is clang 3.8.0 (in x86 architectures) otherwise, modern versions of gcc exist for developing with C++11. Examples are of course available too (lang/gcc<version> ... until gcc 7.0 devel).

Numerous libraries for various topics are also present, web services SOAP with gsoap through User Interfaces with GTK (x11-toolkits/gtk<version>), QT4 or QT 5 (devel/qt<version>), malware libraries with Yara (security/yara), etc.

In terms of IDEs, Eclipse and Netbeans described above allow both C/C++ development. Anjuta and Qtcreator are also available for important projects. If you prefer, FreeBSD has in base vi and Vi Improved which can be found in ports / packages (editors/vim or editors/vim-lite without X11 support).

FreeBSD is a POSIX system, hence porting C/C++ code to this platform depends on the degree of portability of your projects, so the usage of specific “linuxisms” and such.

In case you need more information about porting software in FreeBSD and its specific tools, I would recommend you to read the BSDMag issues no 66 and 68.

## Android / Mobile Development

To be able to do Android development, to a certain degree, the Linux's compatibility layer (aka linuxulator) needs to be enabled. Also, x11-toolkits/swt and linux-f10-gtk2 port/package need to be installed (note that libswt-gtk-3550.so and libswt-pi-gtk-3550.so are necessary. The current package is versioned as 3557 and can be solved using symlinks).

In the worst case scenario, remember that bhyve (or Virtualbox) is available, and can run any Linux distribution efficiently.

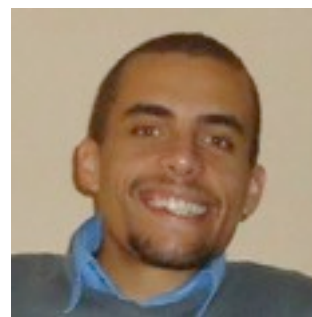
## Source Control Management

FreeBSD comes in base with a version of subversion. As FreeBSD source is in a subversion repository, a prefixed svnlint command prevents conflicts with the package/port.

Additionally, Git is present but via the package/port system with various options (with or without a user interface, subversion support).

## Conclusion

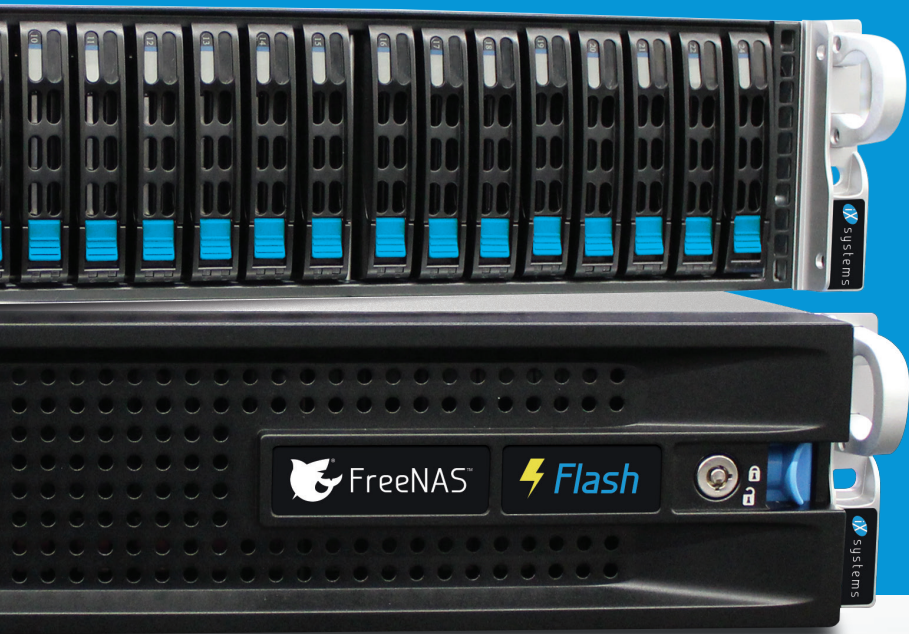
FreeBSD has made tremendous improvements over the years to fill the gap created by Linux. FreeBSD still maintains its interesting specificities; hence there will not be too much blockers if your projects are reasonably sized to allow a migration to FreeBSD.



### About the Author

David Carlier is a software developer since 2001, with several languages from C/C++ to Java, Python and Golang. He is working and living in Ireland since 2012's fall, co-organiser of Dublin BSD Group meetup.





# IS AFFORDABLE FLASH STORAGE OUT OF REACH?

***NOT ANYMORE!***

## IXSYSTEMS DELIVERS A FLASH ARRAY FOR UNDER \$10,000

**Introducing FreeNAS® Certified Flash.** A high performance all-flash array at the cost of spinning disk.

### KEY ADVANTAGES

- ⚡ 10TB of all-flash storage for less than \$10,000
- ⚡ Unifies SAN/NAS for block and file workloads
- ⚡ Runs FreeNAS, the world's #1 software-defined storage solution
- ⚡ OpenZFS ensures data integrity
- ⚡ Scales to 100TB in 2U
- ⚡ Perfectly suited for Virtualization, Databases, Analytics, HPC, and M&E
- ⚡ Performance-oriented design provides maximum throughput/IOPs and lowest latency
- ⚡ Maximizes ROI via high-density SSD technology and inline data reduction

The all-flash datacenter is now within reach. Deploy a FreeNAS Certified Flash array today from iXsystems and take advantage of all the benefits flash delivers.

For more information, visit [ixsystems.com/FreeNAS-Certified-Servers](http://ixsystems.com/FreeNAS-Certified-Servers) today.



# OpenSSH Jump Server with 2FA

## What you will learn...

- How to use install a Linux box to be used as a SSH proxy server (or jump server).
- How to use 2FA to improve the security level of the service.

## What you should know...

- Basic understanding of Linux and SSH protocol.
- Basics of IT security.

This article will discuss how to create a secure OpenSSH Jump server with two-factor authentication.

## Introduction

SSH is the most convenient way to log into a remote UNIX like machine and execute commands. Used virtually in every Unix like box, SSH became one of the most known targets for people that usually try to invade our systems.

The authentication phase of SSH is a key point where we can verify if the users are valid or not. In this phase, we could use methods like password, asymmetric encryption (public and private keys) to check which user is trying to access our service. Nowadays, those methods are not enough anymore, and it is important to use stronger methods of authentication to keep bad people out of our systems.

Using two-factor authentication methods (2FA), we can protect the SSH services from many kinds of attacks, but the use could be not so convenient. A way to solve this problem and continue to be safe is to install SSH jump servers on our environment.

In this article, I will show you how to install a secure jump server using 2FA suitable to access your servers from the Internet in a safe way. The installation and examples will be based on Debian 8.7, but the ideas and configuration could be used in other Linux flavors or even other Unix like operating systems.

## Why Do I Need a Jump Server?

The need to access our servers from everywhere is pretty obvious. We have people cases like the analysts who are on call duty, traveling or those who moved to some

paradisiac beach in some Caribbean country and work from there.

Virtual private networks (VPNs) are a good solution for most of cases. However, to have a Jump server can be a simpler and cheaper way to connect to your servers maintaining the same level of security. You can also use both solutions in a complementary way. In doing so, the Jump server will reject direct connections from the Internet and people will have to connect to the VPN before, thus increasing the overall level of security.

Another case that Jump servers are useful is when it is required to connect to our cloud servers. In this case, we will create a safe bridge between the company's network and the cloud environment without the necessity of a VPN service.

## Defining the Architecture?

Before you start installing the new servers and services, it is necessary to define your architecture. First of all, to avoid that your SSH Jump server becomes the single point of failure, probably you will need to install the Jump servers in pairs. If your environment is split between on-premises and cloud environment, you will need more than a pair. If you have VPN solutions that safely connect your networks, just a pair is sufficient.

For each pair of servers, you will also need a pair of public IP addresses if you don't have a solution of VPN that allows roaming users to connect to your environment safely. It is desirable that the Jump servers be as close as possible to the servers that they intend to protect. The closer they are, the faster you can transfer files and more restrict the SSH daemon of the servers can be kept.

It's important to notice that your pair of Jump servers are at a potential risk in your network, especially if they have both interfaces connected to the Internet and to our internal servers. If the Jump server is not well protected, we will be decreasing the overall security level of our network.

## First Steps

First of all, we need to harden the servers before installing them and configuring the SSH services on them. The OS hardening is out of scope of this article because the subject is comprehensive to be covered. We can see the summary of the most important steps related to the hardening of the operating system:

- Keep the operating system up to date, avoiding to have known vulnerabilities that can expose your system to attackers.

- Install a HIDS (Host Intrusion Detection System), like samhain or tripwire. It's important to keep track of any important change in your system.

- Enable strong logging and send the logs to an external log server. It is important to configure your HIDS to send logs remotely using syslog or by another way. If you are creating a pair of Jump servers, you can send logs from one to the other and vice versa. Some syslog services like rsyslog and syslog-ng support the TLS protocol. Use it instead of plain text messages to send the log messages;

- Disable routing if your Jump server has more than one interface (desired configuration).

- Authenticate your users in an external LDAP server, keeping there the SSH public keys. Doing that, you can better monitor the users in your environment. For example, creating a notification whenever your `/etc/shadow` or `/etc/passwd` is changed. If your LDAP contains people that don't need to use the Jump server, create some control to avoid them being able to log in, use ldap filters, `Allow|Deny User|Groups` at `sshd_config` or at least false shells;

- Configure a firewall in your operating system. Even if you have a firewall in the border of your network, you cannot trust an external control. Block all inbound traffic except from the port that the SSH daemon is listening;

- Remove the compilers and other development tools. It becomes more difficult for someone to download and compile some code to exploit some vulnerability and elevate privileges;

- If you have some http proxy in your network, you can allow the use of http, https and ftp only. To do that, you can block the outbound traffic using these protocols directly to the Internet using your local and/or network firewall. In the proxy, it allows just a connection to apt-get or yum sources. Also, it makes it difficult for someone trying to download some exploit to elevate privilege. If someone could download the exploit, at least, you have some log to know exactly what he got from the Internet. Block all the other protocols that are not managed by your proxy. Keep the outbound traffic unblocked to your managed servers. If possible, only to the SSH protocol;



- Compile your bash with the support to send logs to syslog. This option is available in bash version 4, and unfortunately, it's necessary to change the code and compile it to use the functionality;

- Configure SSH to keep your common users inside a chroot. In this case, there is a big advantage of using a BSD\* as a SSH Jump server instead of a Linux distro. All BSDs have the jail, so it's possible to protect one user from the other. Using Linux, we are limited to use Chroot. Thus, someone can look at the process that another user is running. Besides, this user cannot navigate to the OS and other users' directories;

- Uninstall all unnecessary service, software or protocol. Less software means less vulnerabilities and the safer your servers will be;

- Configure sysctl.conf to protect the TCP/IP stack, memory, syslog, etc. The Linux kernel has a lot of securities that can avoid many kinds of attacks, and are disabled by default;

- Run your SSH daemon in a non-default port. It will avoid the annoying brute force attack on the scripts kiddies, and your log of authentication will be cleaner.

- After the installation of your Jump servers, block the entire SSH traffic directly to your servers in the external firewall.

## Create Your Duo Admin Account and Application

First of all, get the cellphone that you will use as the 2FA and install Duo Mobile downloaded from Google Play Store or Apple Store.

Thereafter, go to <https://duo.com/> and click on Sign Up. Fill the form with your personal information. After that, you will be prompted for the Password. The next step will be to activate the account. Duo page will show the QRCode to be scanned by the Duo app in order to activate the account. Open the Duo app and push the button with a key logo (with the plus sign beside it) at the top right corner. Give permission to the Duo app to use the camera, if needed and scan the QRCode displayed at the page. Hence after, you are prompted to set a backup phone number. You can use the main phone number also as the backup one if you want to.

Finally, you are prompted to confirm your identity to Log In. As you can see, the access to the admin page

requires the 2FA and if you don't change the configuration, there are four options to use as a 2FA method:

- 1) You can type the passcode manually, getting it after opening the Duo app and pushing the button with the key logo beside the related applications. This option doesn't require Internet nor telephony connection on your cell phone.

- 2) Duo push. This is the most convenient way to authenticate. Duo will push a notification to your cell phone, and you will need Internet connection in your cell phone.

- 3) Passcode via SMS.

- 4) Duo can call you to say the passcode.

The methods 3 and 4 require credits of telephony. I suggest that you use Duo push whenever your cellphone has some Internet connection and the method 1 (manually) otherwise.

Log in with your new account and Duo 2FA at <http://www.duo.com>, go to Applications and push "Protect an Application". Look for "Unix Application", change the default name to the hostname of your Jump server and push "Save Changes". Your Duo admin account and application will have been created.

## Install pam\_duo in your Linux box

The following procedures can be used to install duo on Debian 8 (Jessie) boxes. You can find information for other operating systems at:

<https://duo.com/docs/duounix#overview>

Create `/etc/apt/sources.list.d/duosecurity.list` with the following contents:

```
# deb http://pkg.duosecurity.com/Debian
jessie main
```

It will configure the duo repository. Add a new key to the list of trusted keys of apt-get:

```
# curl -s https://duo.com/APT-GPG-KEY-DUO
| sudo apt-key add -
```

re-synchronize the package index files of apt-get from their sources:

```
# apt-get update
```

And install duo-unix package

```
# apt-get install duo-unix
```

Log in at <http://www.duo.com>, go to Applications and push “Protect an Application”. Look for “Unix Application”, change the default name to the hostname of your Jump server and push “Save Changes”. Open the application that you just created. At the top of the page, you can see the section Details.

Use the information in this section, Details, to configure `/etc/duo/pam_duo.conf` file, `ikey`, `skey` and `host` fields.

Change `pushinfo` to `yes` at `/etc/duo/pam_duo.conf`. Push is one interesting feature of DUO. Instead of typing the code, you will receive a notification on your cellphone and will be authenticated after the push of two buttons (I have no idea why Duo requires that you push confirm rather than authenticating after the first push of a button. One push makes more sense for me).

At this point, your system is ready to authenticate using duo through pam modules. You can use it to virtually authenticate any software with pam support. This article will show how to use it to authenticate SUDO and SSH.

## Creating users

Before the user is ready to be utilized, it needs to be created and enrolled. Users can be created manually or

imported, and Duo also can synchronize the users with some Active Directory service. The procedure of enrollment needs to be done manually because the user needs to add the account inside the Duo mobile app.

Once created and enrolled, the user will be available for all the services that authenticate at duo on that company. Different from other solutions of 2FA like google-authenticator, Duo is a centralized authentication system. So, it's not necessary to use more than one account or to copy the `.google-authenticator` file (unsafe).

The procedure of creating a user is not a requirement in the procedure of using Duo. Once you authenticate a service that uses Duo, all the users that will use the service can receive the message inviting to be enrolled and do the procedures by themselves.

The following example shows the message that a user that was never enrolled at Duo receives after using a SUDO service configured to use DUO:

```
$ sudo su -
```

We trust you have received the usual lecture from the Local System

Administrator. It usually boils down to these three things:

```
#1) Respect the privacy of others.
```

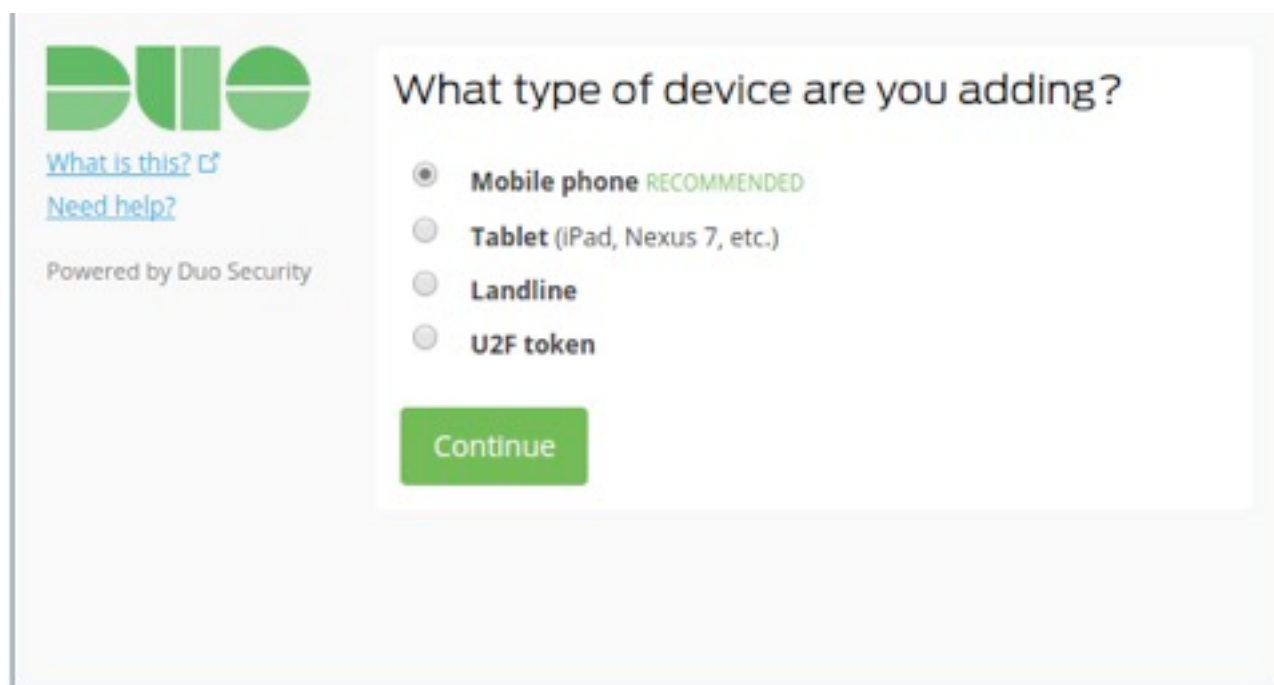


Figure 1. Start setup

#2) Think before you type.

#3) With great power comes great responsibility.

Authenticated with partial success.

Please enroll at

<https://api-f11be55e.duosecurity.com/portal?code=afd3b14279146a15&akey=DAS59J8HUTY7CT3JOHME>

Please enroll at

<https://api-f11be55e.duosecurity.com/portal?code=f2dd530d75e76851&akey=DAS59J8HUTY7CT3JOHME>

Please enroll at

<https://api-f11be55e.duosecurity.com/portal?code=64ec2d88884618f0&akey=DAS59J8HUTY7CT3JOHME>

sudo: 3 incorrect password attempts

As you can see, the enrollment URL is given to the user at the first login and he/she can proceed by himself/herself. For security reasons, the enrollment URL expires, so it's important to enroll immediately after you get the invitation message.

If you prefer to create the user manually, you can simply go to the admin page, click on Users, after that Add Users. Fill the username field remembering that this username needs to be the same as the username in the operating system. Click in Add User and fill the email field. If you click on Send Enrollment Email, the user will receive an email with the URL to proceed with the enrollment.

In Duo, the licensing model is based on the number of users using the system. Until 10 users, you can use the free version. Therefore, it's important to create a necessary number of users.

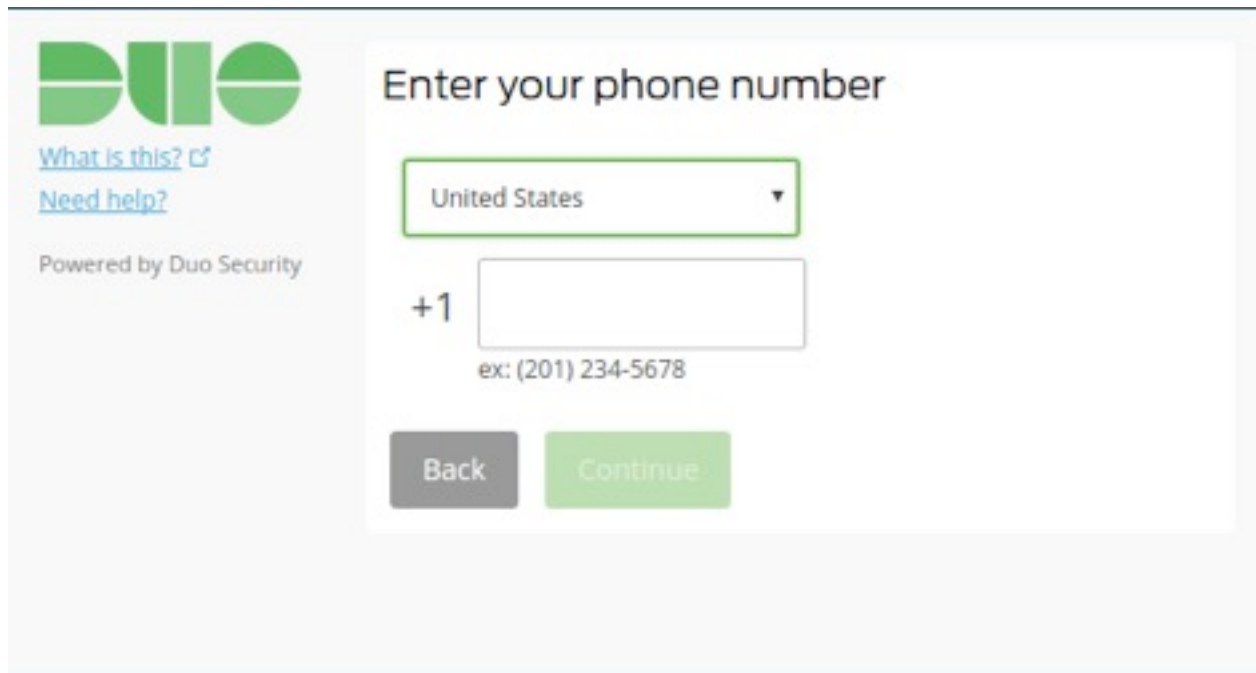
The image shows a Duo Security enrollment interface. On the left, there is a Duo logo and links for 'What is this?' and 'Need help?'. Below that, it says 'Powered by Duo Security'. The main section is titled 'Enter your phone number'. It features a dropdown menu for the country, currently set to 'United States'. Below the dropdown is a text input field for the phone number, preceded by a '+1' sign. An example number 'ex: (201) 234-5678' is shown below the input field. At the bottom of the form are two buttons: 'Back' and 'Continue'.

Figure 2. Phone number

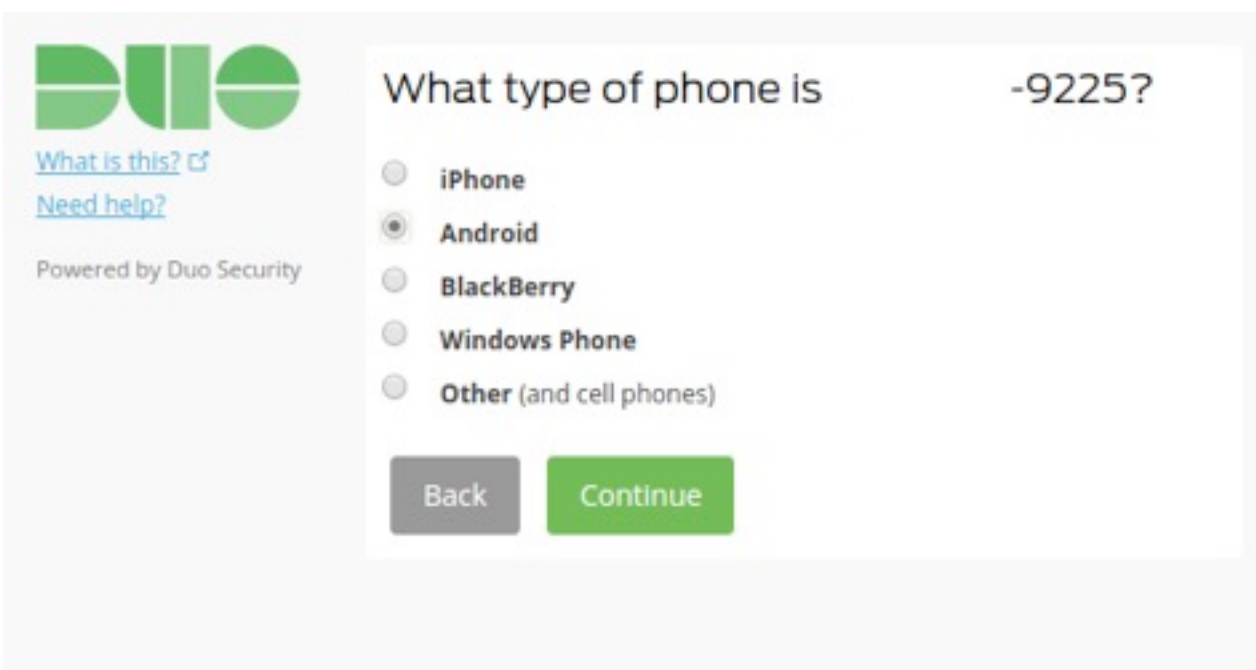
The image shows a Duo Security enrollment interface. On the left, there is a Duo logo and links for 'What is this?' and 'Need help?'. Below that, it says 'Powered by Duo Security'. The main section is titled 'What type of phone is -9225?'. It features a list of radio button options: 'iPhone', 'Android', 'BlackBerry', 'Windows Phone', and 'Other (and cell phones)'. The 'Android' option is selected. At the bottom of the form are two buttons: 'Back' and 'Continue'.

Figure 3. Choose the operating system of your cell phone



## Enrollment process

The enrollment process is very simple. Follow the link of enrollment and fill the fields as shown below.

First step, a greetings message is displayed. Just click on Start setup (see Figure 1).

Second step: Besides, you can use even a landline to authenticate, but a mobile phone gives you more security controls and allows the feature duo push. Press Continue:

Third step: Fill in your cell phone number information. The cellphone number is required and can be used to receive the passcode by SMS or call. This number is not important for the push method, I tested using a wrong number and it works anyway (see Figure 2).

Fourth step; choose the operating system of your cell phone (see Figure 3).

Fifth step, click in I have Duo Mobile installed if you already have it installed. Otherwise, keep this screen open and do it.

Sixth step; open your Duo mobile app, click on the button with the key and the '+' beside it and scan the QRCode (see Figure 4).

Seventh step; It's showed that the code was successfully added. Just click Continue.

Eighth step, choose between the option that sends a Duo Push automatically when you try to login or if you select

your preferred method of receiving the passcode. This option depends on your preference. Click Finish Enrollment after your choice.

Last step, a congratulations screen is displayed.

## Authenticating SUDO Using DUO

In most cases, using Duo to authenticate could be safer than using a password. A good example is SUDO. Besides, you can use PASSWD inside /etc/sudoers to force people to use password, but it's possible to store the password and use some script to break this control (if you are interested, search for non-interactive sudo using expect). Also, it's difficult to block some automation software that use sudo to elevate privileges, like Ansible. Duo can be the solution for this problem.

Let's change our pam configuration in order to require Duo to authenticate. Open

/etc/pam.d/sudo and change

```
@include common-auth
```

to:

```
#@include common-auth
```

```
auth [success=1 default=ignore]
/lib64/security/pam_duo.so
```

```
auth requisite pam_deny.so
```

```
auth required pam_permit.so
```

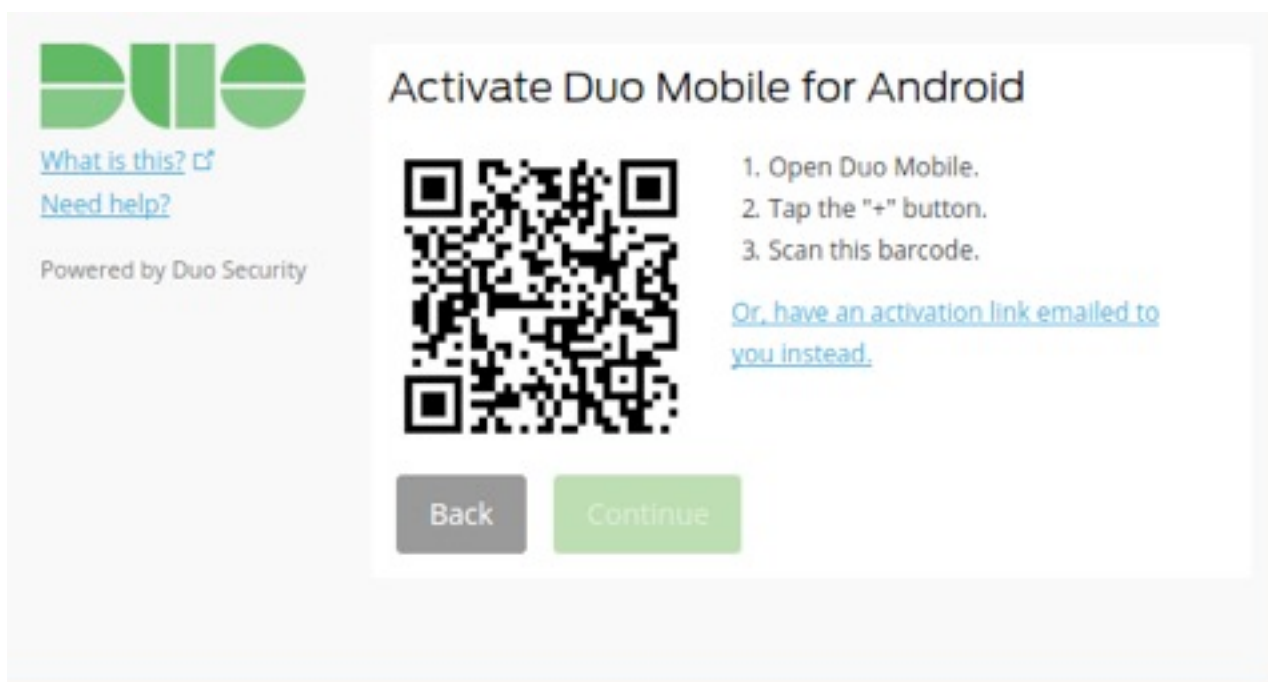


Figure 4. Scan the QRCode

Our system is now prepared to prompt for Duo 2FA whenever we use sudo, let's test it.

Become a user with sudo permissions to run su and run the following:

```
$ sudo su -
```

```
Duo two-factor login for neves
```

Enter a passcode or select one of the following options:

1. Duo Push to XXX-XXX-9225
2. Phone call to XXX-XXX-9225
3. SMS passcodes to XXX-XXX-9225

Passcode or option (1-3): 1

Pushed a login request to your device...

Success. Logging you in...

```
# id
```

```
uid=0(root) gid=0(root) groups=0(root)
```

```
#
```

As you can see, I typed 1 when prompted for the passcode. At this moment, I received the notification on my cellphone, with information about the company, name of the server, IP, city and country information based on GeoIP, user, and date. The first screen you need to push Approve, the next screen has the same information and a button Confirm (no idea why the necessity of the second screen). After the second screen, you are authenticated and a notification of success is sent to your cell phone.

Using Duo to authenticate SUDO is safer and more convenient than passwords. You can still bypass the pam authentication of sudo using NOPASSWD at sudoers when needed.

## Authenticating SSH Using DUO

Now, let's go to the most important part of the article, how to authenticate the SSH server using Duo. First, it can work together with password and with SSH public keys. I strongly recommend that SSH public keys are used. It's possible to have both the SSH private key and the soft token for the same gadget, what theoretically cannot be the desired scenario regarding security. Imagine if someone installs a malware in your gadget and is able to

authenticate in some server using the private key stored there and can handle the push notification at the same time to accept and confirm it. Besides being perfectly possible to happen, the current malwares are not so advanced. I still prefer the security that SSH public keys give me than the insecurity of having both SSH private keys and Duo mobile push in the same device. If you are a security paranoid, never store the SSH key in your cellphone.

Let's change our pam configuration to require Duo to authenticate for our SSH service. Open /etc/pam.d/sshd and change:

```
@include common-auth
```

to:

```
#@include common-auth
```

```
auth [success=1 default=ignore]
/lib64/security/pam_duo.so
```

```
auth requisite pam_deny.so
```

```
auth required pam_permit.so
```

Edit /etc/ssh/sshd\_config and make changes keeping the configuration like that:

```
UsePAM yes
```

```
ChallengeResponseAuthentication yes
```

```
UseDNS no
```

If you will use passwords + duo mobile, that PasswordAuthentication is set to yes. If you will use SSH public keys, change also the following configuration:

```
PubkeyAuthentication yes
```

```
PasswordAuthentication no
```

```
AuthenticationMethods
publickey,keyboard-interactive
```

Let's try a login now and see what happens:

```
$ ssh myserver
```

```
Authenticated with partial success.
```

```
Duo two-factor login for neves
```

Enter a passcode or select one of the following options:

1. Duo Push to XXX-XXX-9225
2. Phone call to XXX-XXX-9225
3. SMS passcodes to XXX-XXX-9225

Passcode or option (1-3): 1

Success. Logging you in...

The programs included with the Debian GNU/Linux system are free software.

The exact distribution terms for each program are described in the individual files in /usr/share/doc/\*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent

permitted by applicable law.

Last login: Sun Apr 2 19:01:19 2017 from 24.52.229.66

neves@myserver:~\$

First, the message 'Authenticated with partial success.' is showed and is the way we know that the authentication using SSH public keys worked. After that, the behavior is pretty much the same as the authentication of SUDO. You are asked to choose the method to give the passcode and it's authenticated thereafter.

## Conclusions

Using Duo mobile, you can create secure SSH Jump servers to access your environment using 2FA. You can install duo mobile on all servers in the environment, but it's not so practical to push buttons whenever you login to any server. With SSH Jump servers and some session multiplexer like screen or byobu, you have access to the entire environment in a safer and practical way.

You can explore Duo and use it to authenticate dozens of services like VPN, Wordpress and so on. You can even install the login\_duo and run it from some script or binary.

Duo offer lots of other security control possibilities, like the possibility to control the login based on the country using GeoIP, uses of hard tokens, fixed or temporary passcodes (can be used in case someone is temporarily without a cellphone), bypass of 2FA, blocking of the user, etc.

Using a Jump server with Duo can increase the security level of your network considerably. However, don't forget to research and do the items in the First Steps section. Like any security solution, Duo cannot fully protect your environment alone. Both Duo and your Jump server are only some additional security controls that can help you to increase a little bit the overall level of security of your network.



### About the Author

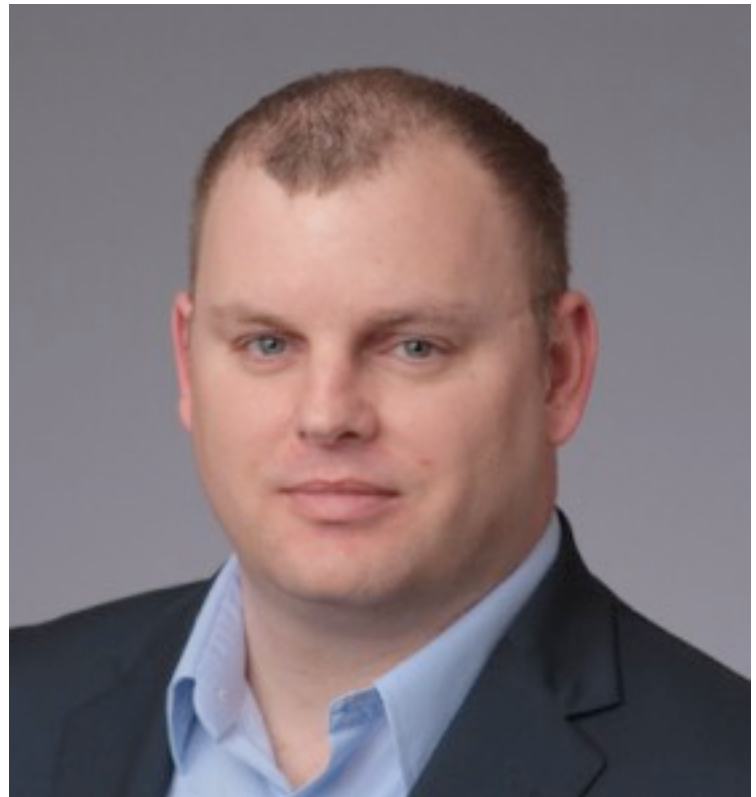
Leonardo Neves Bernardo started using Unix in 1996, when he considered it to be more interesting than any other system at that time. For more than twenty years, he has worked in several IT areas but has always been focused on Unix operating systems.

Leonardo holds a Bachelor's degree in Computer Science from the Universidade Federal de Santa Catarina, Florianópolis, Santa Catarina, Brazil as well as various certifications including LPIC-3, LPIC-300, LPIC-302 and LPIC-303, RHCSA and the ITILv3 Foundation. Linkedin profile at:

<https://www.linkedin.com/in/leonardoneves>



# \_UNIX BLOG PRESENTATION



## Daniel Miessler's Blog

*The goal of A vim Tutorial and Primer tutorial is to take you through every stage of progression—from understanding the vim philosophy (which will stay with you forever), to surpassing your skill with your current editor, to becoming “one of those people”.*

## Meet Daniel Miessler

***Daniel Miessler is the Director of Advisory Services at IOActive and has 17 years of experience in information security. His background is in technical security testing and enterprise defense, including network, web, application, mobile, IoT testing, and adversary-based risk management.***

***He is the leader of the OWASP IoT Security project and speaks regularly at conferences, on panels, and to the media on the topics of information security and technology trends. He also produces a blog, podcast, and newsletter with similar themes.***

**Can you tell our readers about yourself and your blog?**

I've been blogging since 1999, which is also when I started in information security.

**How you first got involved in with blogging?**

I first used blogging as a record of what I had learned. So I would study a topic and write myself tutorials so that it would be hard for me to forget those concepts again. And those tutorials became popular over time.

**What's the best thing a blogger can give to his readers?**

Concise value is the best thing. People are busy and they need to be able to get key points very quickly.

**Everyone has a favourite post. Name yours and why?**

I'd have to say my vim primer. It is the most comprehensive primer I've done, and I think it's organized in an intuitive way. I have got lots of emails saying that the Vim as Language section helped a lot of people learn vim in a way they could not before. <https://danielmiessler.com/study/vim/#language>

**What is your favourite OS and why?**

My favorite OS is probably Ubuntu. The reasons are its universal nature, the support in the community, and its constant but predictable evolution.

**What is your advice to anyone who wants to advance their UNIX knowledge?**

I have a few pieces of advice there:

1. Use UNIX all the time, for whatever you do. If you have jobs that require you to do that you will learn a lot whether you want to or not.
2. Consider getting a UNIX certification of some type that will force you to learn fundamentals that you might have otherwise missed.
3. Focus on the core Unix concepts of the output of one command becoming the input to another. Go into the various binary directories /bin/, /usr/bin, etc. and become familiar with every command in there does. Understand how they could be used together.

**What do you think what makes UNIX so beloved?**

It's the combination of simplicity, transparency, and power when commands are linked with each other.

**What is the future of UNIX? What do you think?**

I think UNIX will continue to dominate the backend of the Internet, as we're seeing with AWS. Its simplicity, predictability, stability, and pricing model make it hard to beat.

**Do you have any specific goals for the rest of this year?**

I am going to try to finish my book on Tcpdump.

**Thank you**

# A vim Tutorial and Primer

There are dozens of Vim references online, but most of them either go ninja straight away, or start basic and don't go much deeper.

The goal of this tutorial is to take you through every stage of progression—from understanding the vim philosophy (which will stay with you forever), to surpassing your skill with your current editor, to becoming “one of those people”.

In short, we're going to learn vim in a way that will stay with you for life. Let's get started.

## WHY VIM

I believe people should use [vim](#) for the following three reasons: It's ubiquitous. You don't have to worry about learning a new editor on various boxes.

It's scalable. You can use it just to edit config files or it can become your entire writing platform.

It's powerful. Because it works [like a language](#) vim takes you from frustrated to demigod very quickly.

In short, I believe you should consider competence with vim the way you consider competence with your native language, or basic maths, etc. So much in technology starts with knowing your editor.

## APPROACH

[Kana the Wizard](#) says there are five (5) levels to vim mastery:



Level 0: not knowing about vim

Level 1: knows vim basics

Level 2: knows visual mode

Level 3: knows various motions

Level 4: not needing visual mode

I don't know about that, but I thought it was worth mentioning. Kana's a wizard, after all. My approach to showing you vim is based around four main areas:

- Intro/Basics: this is a basic pitch/prop to get you up and running and thinking the right way.
- Getting Stuff Done: this is the meat. Bring a fork. And probably a napkin. You seem messy.
- Advanced: this is where I show you how to become one of “those people” with vim. Frequent Requests: this is where I give you the tricks to do that one thing you need.

In other words, if you're already up and running you should be able to jump to [Getting Stuff Done](#) and start knocking stuff out. If you're already solid on those bits, then head over to the [Advanced](#) section to learn Kung Fu. And if you're here to solve a specific “forgot how to do that one thing”, check out the [Frequent Requests](#) area.

So, setup, basic usage, ninja stuff, and then frequently asked tasks—and you basically just go where you need to within those.

## CONFIGURATION

As I said, I'm not looking to turn this into the uber-vim-config piece. There are many of those out there. This is a primer / tutorial designed to make you strong with vim's concepts, and to build a long-term power with the tool. But we'll talk through some configuration basics as part of that.

First, I recommend you go with a (mostly) self-managed vim install. I used to be into [Janus](#), but didn't like the fact that I wasn't sure what it was doing exactly. My favorite configs are simple and elegant, and it should be the same with vim.

So, to that end, I use a straight `~/.vim` directory under home, and a `~/.vimrc` file as my configuration.

## A FEW KEY ~/.VIMRC CHANGES

Firstly, the `<Esc>` key for leaving insert mode is, in my opinion, rather antiquated. Vim is about efficiency, and it's hardly efficient to leave the home keys if you don't have to. So don't.

```
inoremap jk <ESC>
```

[ NOTE: Some like to change the `<ESC>` key to `jj`, but I don't find that as natural as rolling from `j` to `k`. ]

## Changing the leader key

The leader is an activation key for shortcuts, and it's quite powerful. So if you are going to do some shortcut with the letter “c”, for example, then you'd type whatever your leader key is followed by “c”.

The default leader (`\`) key seems rather out of the way as well, so I like to remap the leader key to Space.

```
let mapleader = "<Space>"
```

Now when you're executing your nifty shortcuts that you're about to learn, you can do so with either thumb, since your thumbs are always on the Space bar.

[ NOTE: Thanks to Adam Stankiewicz for the Space as Leader recommendation. ]

## Remapping CAPSLOCK

This one isn't in your conf file, but it's an important deviation from the defaults. The CAPSLOCK key on a keyboard is generally worthless to me, so I remap it to `Ctrl` [at an operating system level](#). This way my left pinky can simply slide to the left by one key to execute `Ctrl-whatever`.

Then there are just a few basics that are recommended by most and make things much easier overall.

```
filetype plugin indent on  
syntax on  
set encoding=utf-8
```



Remember, you can spend a lifetime optimizing your `~/.vimrc` file; these are just a few things to get you started. For full configs check out [my setup](#) or look at the links in [the references section](#).

## PLUGIN MANAGEMENT

[ NOTE: If you're not already familiar and comfortable with plugins, skip this section for now and come back to it another time. ]

### Native management

I have upgraded to Vim 8.x, which now supports native plugin management. You simply drop your plugins under:

```
~/.vim/pack/plugin/foldername/
start/pluginname
```

Done and done. Now you can play with any plugins you want using the method above and they will be loaded automatically when Vim starts. I much prefer this to the third-party plugin management options we needed with 7.x

## LEVERAGING GITHUB FOR BACKUP AND PORTABILITY

One thing I do with my Vim setup is I keep my entire `~/.vim` directory within a [git](#) repository stored [here](#). What this does is give me the ability to go to a shiny new box and say `git clone https://github.com/danielmiessler/vim` and have my entire vim environment exactly the way I want it.

You may want to do the same.

Simply clone to your new box and then symlink `~/.vimrc` to `~/.vim/vimrc` and you're done.

## VIM AS LANGUAGE

Arguably the most brilliant thing about vim is that as you use it you begin to *think* in it. vim is set up to function like a language, complete with nouns, verbs, and adverbs.

Keep in mind that the terms I'm going to use here are not technically correct, but should help you understand better how vim works. Again, this guide is not meant to replace a full book or the

help—it's meant to help you get what doesn't come easily from those types of resources.

## VERBS

Verbs are the actions we take, and they can be performed on nouns. Here are some examples:

d: delete

c: change

y: yank (copy)

v: visually select (V for line vs. character)

## MODIFIERS

Modifiers are used before nouns to describe the way in which you're going to do something. Some examples:

i: inside

a: around

NUM: number (e.g.: 1, 2, 10)

t: searches for something and stops before it

f: searches for that thing and lands on it

/: find a string (literal or regex)

## NOUNS

In English, nouns are objects you do something *to*. They are objects. With vim it's the same. Here are some vim nouns:

w: word

s: sentence

): sentence (another way of doing it)

p: paragraph

}: paragraph (another way of doing it)

t: tag (think HTML/XML)

b: block (think programming)

## NOUNS AS MOTION

You can also use nouns as motions, meaning you can move around your content using them as the size of your jump. We'll see examples of this below in the moving section.

## BUILDING SENTENCES (COMMANDS) USING THIS LANGUAGE

Ok, so we have the various pieces, so how would you build a sentence using them? Well, just like English, you combine the **verbs**, **modifiers**, and **nouns** in (soon to be) intuitive ways.

For the notation below, just remember RGB (**red**, **green**, **blue**, which I still remember as "roy-gee-biv") is VMN (verb, modifier, noun):

# Delete two words

**d2w**

# Change inside sentence (delete the current one and enter insert mode)

**cis**

# Yank inside paragraph (copy the paragraph you're in)

**yip**

# Change to open bracket (change the text from where you are to the next open bracket)

**ct<**

Remember, the "to" here was an open bracket, but it could have been anything. And the syntax for "to" was simply t, so I could have said dt. or yt; for "delete to the next period", or "copy to the next semicolon".

Isn't that beautiful? Using this thought process turns your text editing into an intuitive elegance, and like any other language the more you use it the more naturally it will come to you.

## GETTING THINGS DONE

Now that we've handled some fundamentals, let's get tangible and functional.

## WORKING WITH YOUR FILE

Some quick basics on working with your file.

vi **file**: open your file in vim

:w: write your changes to the file

:q!: get out of vim (quit), but without saving your changes (!)

:wq: write your changes and exit vim  
:saveas ~/some/path/: save your file to that locationvim

[ NOTE: While :wq works I tend to use ZZ, which doesn't require the ":" and just seems faster to me. You can also use :x ]

ZZ: a faster way to do :wq

## SEARCHING YOUR TEXT

One of the first things you need to be able to do with an editor is find text you're looking for. vim has extremely powerful search capabilities, and we'll talk about some of them now.

### Searching by string

One of most basic and powerful ways to search in vim is to enter the "/" command, which takes you to the bottom of your window, and then type what you're looking for and press ENTER.

# Search for include

/include<CR>

That'll light up all the hits, as seen below:



Once you've done your search, you can press "n" to go to the next instance of the result, or "N" to go to the previous one. You can also start by searching backward by using "?" instead of "/".

### Jumping to certain characters

One thing that's brutally cool about vim is that from anywhere you can search for and jump to specific characters. In this article, for example, because I'm

editing HTML, I can always jump to the "<" character to be at the end of the sentence.

# Jump forward and land on the < character

f<

# Jump forward and land right before the < character

t<

You can think of this as "find" for the first one, which lands right on it, and "to" for the second one, which lands right before it.

What's really sick, though is that you can use these as nouns for commands. So just a second ago while editing this sentence I did:



# Change to the next "<"

ct<

This works for whatever character, e.g. periods, open brackets, parenthesis, regular letters—whatever. So you can just look forward in your text and jump to things or you can know that it's somewhere up there and just got to it wherever it is.

[ NOTE: You can use the ";" to move forward to the next instance of what you searched for—whether you used "t" or "f" to search for it. Also, a comma "," does the same, but backward. ]

### A search reference

/ {string}: search for string

t: jump up to a character

f: jump onto a character

\*: search for other instances of the word under your cursor

n: go to the next instance when you've searched for a string

N: go to the previous instance when you've searched for a string

:: go to the next instance when you've jumped to a character

,: go to the previous instance when you've jumped to a character

## MOVING AROUND IN YOUR TEXT

Getting around within your text is critical to productivity. With vim this is both simple and elegant, as it leverages the core principal of [vim as language](#) that we talked about above. First, some basics.

### Basic motions

We start with use of the home row. Typists are trained to keep their right hand on the j, k, l, and ";" keys, and this is the starting point for using vimas well.

j: move down one line

k: move up one line

h: move left one character

l: move right one character

This is a bit strange at first, and it just takes a few minutes of practice to get functional with, but it'll quickly become so natural that you'll be doing it in Microsoft Word and Outlook (it doesn't work there, by the way).

So your right index and middle fingers move you up and down lines, and your index and ring fingers move you left and right by one character.

### Moving within the line

You can easily move within the line you're on.

O: move to the beginning of the line

\$: move to the end of the line

^: move to the first non-blank character in the line

t": jump to right before the next quotes  
f": jump and land on the next quotes  
[ NOTE: , and ; will repeat the previous  
t and f jumps. ]

### Moving by word

You can also move by word:

w: move forward one word  
b: move back one word  
e: move to the end of your word

When you use uppercase you ignore  
some delimiters within a string that  
may break it into two words.

W: move forward one big word  
B: move back one big word

This uppercasing of a given command  
having different and more powerful  
effects is something we'll see  
frequently.

### Moving by sentence or paragraph

): move forward one sentence  
}: move forward one paragraph

### Moving within the screen

H: move to the top of the screen  
M: move to the middle of the screen  
L: move to the bottom of the screen  
gg: go to the top of the file  
G: go to the bottom of the file  
^U: move up half a screen  
^D: move down half a screen  
^F: page down  
^B: page up

### Jumping back and forth

While you're in normal mode it's  
possible to jump back and forth  
between two places, which can be  
extremely handy.

Ctrl-j: jump to your previous navigation  
location

Ctrl-o: jump back to where you were

### Other motions

:\$line\_numberH: move to a given line  
number

M: move to the middle of the screen

L: move to the bottom of the screen  
^E: scroll up one line  
^Y: scroll down one line  
^U: move up half a page  
^D: move down half a page  
^F: move down a page  
^B: move up a page

So let's package that all up into one  
place:

### Motion command reference

j: move down one line  
k: move up one line  
h: move left one character  
l: move right one character  
0: move to the beginning of the line  
\$: move to the end of the line  
w: move forward one word  
b: move back one word  
e: move to the end of your word

): move forward one sentence  
}: move forward one paragraph  
:line\_number: move to a given line  
number

H: move to the top of the screen  
M: move to the middle of the screen  
L: move to the bottom of the screen  
^E: scroll up one line  
^Y: scroll down one line  
gg: go to the top of the file  
G: go to the bottom of the file  
^U: move up half a page  
^D: move down half a page  
^F: move down a page  
^B: move up a page

Ctrl-j: jump to your previous navigation  
location

Ctrl-o: jump back to where you were

[ NOTE: I map my CAPSLOCK to Ctrl  
so I can use it for these various  
Ctrl-based movements, among other  
things. ]

### CHANGING TEXT

Ok, so we've done a bunch of moving  
within our text; now let's make some  
changes. The first thing to remember is

that the motions will always be with  
us—they're part of the language  
(they're modifiers in the [vocabulary  
above](#)).

### Understanding modes

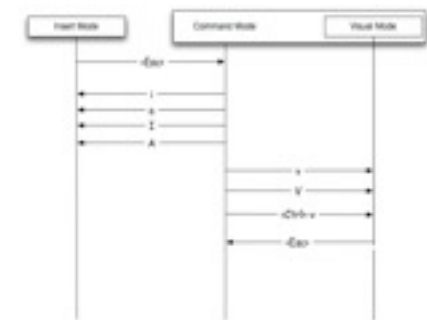


Image from [Michael Jaki](#)

The first thing we need to grasp is the  
concept of modes. It's a bit  
counterintuitive at first but it becomes  
second nature once you grok it. Most  
guides start with this bit, but I find it a  
bit obtuse to lead with, and I think the  
transition point from Normal to Insert is  
a great place to introduce it.

You start in Normal Mode. One of the  
most annoying things about vim for  
beginners is that you can't just open it  
up and start typing. Well, you can, but  
things go sideways pretty quick if you  
do.

Normal Mode is also known as  
Command Mode, as it's where you're  
usually entering commands.  
Commands can be movements,  
deletions, or commands that do these  
things and then enter into Insert Mode.

Insert Mode is where you make  
changes to your file, and there are tons  
of ways of entering Insert Mode from  
Normal Mode. Again, don't worry, this  
all becomes ridiculously simple with a  
bit of practice.

Visual Mode is a way to select text. It's  
a lot like Normal Mode, except your  
movements change your highlighting.  
You can select text both character-wise  
or line-wise, and once in one of those  
modes your movements select more  
text.

The purpose of Visual Mode is to then  
perform some operation on all the  
content you have highlighted, which  
makes it very powerful.



Ex Mode is a mode where you drop down to the bottom, where you get a “:” prompt, and you can enter commands. More on that later. Just know that you can run some powerful command-line stuff from there. There are some other modes as well, but we won’t mess with them here as they tend to live outside primer territory.

## Remembering your language

Let’s recall our language: Verb, Modifier, Noun. So we’re assuming we’re starting in Normal Mode, and we’re going to switch into Insert Mode in order to change something.

Our verb is going to start us off, and we have a few options. We can change (c), insert (i), or append (a), and we can do variations on these, as seen below.

## Basic change/insert options

Let’s start with the options here.

i: *insert* before the cursor

a: *append* after the cursor

I: *insert* at the beginning of the line

A: *append* at the end of the line

o: *open* a new line below the current one

O: *open* a new line above the current one

r: *replace* the one character under your cursor

R: *replace* the character under your cursor, but just keep typing afterwards

cm: change whatever you define as a *movement*, e.g. a word, or a sentence, or a paragraph.

C: *change* the current line from where you’re at

ct?: *change* change up to the question mark

s: *substitute* from where you are to the next command (noun)

S: *substitute* the entire current line

# Change inside sentence

cis

# Go to the beginning of the line and enter insert mode

I

# Start typing right after the cursor

a

As you can see, there are lots of ways to start entering text. There are also some shortcuts (shown above as well) for doing multiple things at once, such as deletion and entering Insert Mode.

# Delete the line from where you’re at, and enter insert mode

C

# Delete the entire line you’re on, and enter insert mode

S

## Changing Case

You can change the case of text using the tilde (~) command. It works as you’d imagine—either on the letter under the cursor, or on a selection.

## Formatting Text

It’s sometimes helpful to format text quickly, such as paragraphs, and this can easily be done with the following command:

# Format the current paragraph

gq ap

gq works based on your textwidth setting, which means it’ll true up whatever you invoke it on to be nice and neat within those boundaries.

[ NOTE: The “ap” piece is the standard “around paragraph” text object. ]

## DELETING TEXT

Now that we know how to change text, let’s see how to do straight deletes. As you’re probably getting now, it’s very similar—just a different action to start things off.

## Basic deletion options

x: *exterminate* (delete) the character under the cursor

X: *exterminate* (delete) the character before the cursor

dm: delete whatever you define as a *movement*, e.g. a word, or a sentence, or a paragraph.

dd: *delete* the current line

dt.: *delete* delete from where you are to the period

D: *delete* to the end of the line

J: *join* the current line with the next one (delete what’s between)

Simple enough.

## UNDO AND REDO

You can’t have a text editor without undo and redo. As you’ve probably noticed, vim does its best to make the keys for the actions feel intuitive, and undo and redo are not exceptions.

u: undo your last action.

Ctrl-r: redo the last action

[ NOTE: Remember that you should have already [remapped your CAPSLOCK](#) to Ctrl so that you can do this quickly with your left pinky. ]

Both commands can be used repeatedly, until you either go all the way back to the last save, or all the way forward to your current state.

## REPEATING ACTIONS

One of the most powerful commands in all of vim is the period “.”, which seems strange, right? Well, the period “.” allows you to do something brilliant—it lets you repeat whatever it is that you just did.

## Using the “.” to repeat your last action

Many tasks you do will make a lot of sense to repeat. Going into insert mode and adding some text, for example. You can do it once and then just move around and add it again with just the “.” Here are a couple of other examples.

```
# delete a word
```

```
dw
```

```
# delete five more words
```

5.

Whoa. And wait until you see it combined with Visual Mode.

## COPY AND PASTE

Another text editor essential is being able to quickly copy and paste text, and vim is masterful at it.

[ NOTE: Another really powerful repeat command is `&`, which repeats your last excommand. ]

### Copying text

vim does copying a bit different than one might expect. The command isn't `c`, as one might expect. If you'll remember, `c` is already taken for "change". vim instead uses `y` for "yank" as it's copy command and shortcut.

`y`: yank (copy) whatever's selected

`yy`: yank the current line

Remember, just like with any other copy you're not messing with the source text—you're just making another...copy...at the destination.

### Cutting text

Cutting text is simple: it's the same as deleting. So whatever syntax you're using for that, you're actually just pulling that deleted text into a buffer and preparing it to be pasted.

### Pasting text

Pasting is fairly intuitive—it uses the `p` command as its base. So, if you delete a line using `dd`, you can paste it back using `p`.

One thing to remember about pasting is that it generally starts right after your cursor, and either pastes characters/words or lines or columns—based on what you copied (yanked). Also remember that you can undo any

paste with the universal undo command "`u`".

## A copy and paste reference

`y`: yank (copy) from where you are to the next command (noun)

`yy`: a shortcut for copying the current line

`p`: paste the copied (or deleted) text after the current cursor position

`P`: paste the copied (or deleted) text before the current cursor position

### # Switching lines of text

```
ddp
```

This is a quick trick you can use to swap the position of two lines of text. The first part deletes the line you're on, and the second part puts it back above where it used to be.

## SPELLCHECKING

We'd be in pretty bad shape if we couldn't spellcheck, and vim does it quite well. First we need to set the option within our conf file.

```
# Somewhere in your ~/.vimrc
```

```
set spell spellang=en_us
```

### Finding misspelled words

When you have set spell enabled within your conf file, misspelled words are automatically underlined for you. You can also enable or disable this by running `:set spell` and `:set nospell`.

Either way, once you've got some misspellings you can then advance through them and take action using the following commands:

```
# Go to the next misspelled word
```

```
]s
```

```
# Go to the last misspelled word
```

```
[s
```

```
# When on a misspelled word, get some suggestions
```

```
z=
```

```
# Mark a misspelled word as correct
```

```
zg
```

```
# Mark a good word as misspelled
```

```
zw
```

I like to add a couple of shortcuts to my `~/.vimrc` file related to spelling. The first just makes it easy to "fix" something:

```
# Fix spelling with <leader>f
```

```
nnoremap <leader>f 1z=
```

This one gets rid of spellchecking when I don't want to see it—like when I'm in creative mode. I can then re-toggle it with the same command.

```
# Toggle spelling visuals with <leader>s
```

```
nnoremap <leader>s :set spell!
```

## SUBSTITUTION

Another powerful feature of vim is its ability to do powerful substitutions. They're done by specifying what you're looking for first, then what you're changing it to, then the scope of the change.

The basic setup is the `:%s`

```
# Change "foo" to "bar" on every line
```

```
:%s /foo/bar/g
```

```
# Change "foo" to "bar" on just the current line
```

```
:s /foo/bar/g
```

[ NOTE: Notice the lack of the `%` before the "`s`" ]

There are many other options, but these are the basics.

## ADVANCED

Brilliant. So we've covered a number of basics that any text editor should have, and how vim handles those tasks. Now

let's look at some more advanced stuff—keeping in mind that this is advanced for a primer, not for Kana the Wizard.

## MAKING THINGS REPEATABLE

We talked [a bit ago](#) about being able to repeat things quickly using the period “.”. Well, certain types of commands are better for this than others, and it's important to know the difference.

In general, the idea with repetition using the period “.” (or as Drew Neil calls it—the dot command) is that you want to have a discreet movement action combined with a repeatable command captured in the “.”.

So let's say that you're adding a bit of text to the end of multiple lines, but you're only doing it where the line contains a certain string. You can accomplish that like so:

# Search for the string

/delinquent

Now, whenever you press the “n” key you'll teleport to the next instance of “delinquent”. So, starting at the first one, we're going to append some text.

# Append some text to the end of the line

A[DO NOT PAY] [Esc]

Ok, so we've done that once now. But there are 12 other places it needs to be done. The “.” allows us to simply re-execute that last command, and because we also have a search saved we can combine them.

# Go to the next instance and append the text to the line

n.

Remember, the idea is to ideally combine a motion with the stored command, so you can jump around and re-execute it as desired.

## TEXT OBJECTS

Text Objects are truly spectacular. They allow you to perform actions (verbs) against more complex targets (nouns).

So, rather than selecting a word and deleting it, or going to the beginning of a sentence and deleting it, you can instead perform actions on these... objects...from wherever you are within them.

Hard to explain; let me give you some examples.

### Word Text Objects

Let's look first at some word-based objects.

iw: inside word

aw: around word

These are targets (nouns), so we can delete against them, change against them, etc.

# Delete around a word

daw

[ NOTE: The difference between “inside” and “around” an object is whether it gets the spaces next to it as well. ]

### Sentence Text Objects

is: inside sentence

as: around sentence

Those work pretty much the same as with word objects, so imagine you're knee deep into a sentence that you decide suddenly you hate. Instead of moving to the beginning of it and figuring out how to delete to the end, you can simply:

# Change inside a sentence

cis

This nukes the entire sentence and puts you in Insert Mode at the beginning of your new one.

### More object types

There are also a number of other object types, which I'll mention briefly.

paragraphs: ip and ap

single quotes: i' and a'

double quotes: i" and a"

I use these constantly when editing code or HTML. Remember the key is that you don't even have to be inside the section in question; you just tell it ci" and it'll delete everything inside the double quotes and drop you inside them in Insert Mode. It's wicked cool. The same works for a few other types of items, including parenthesis, brackets, braces, and tags (think HTML). Think about editing an HTML link, where there is the URL within double quotes, and then the link text within tags; this is handled elegantly by vimby doing two commands: ci" and then cit.

### A text object reference

Here a list of the objects for your reference:

words: iw and aw

sentences: is and as

paragraphs: ip and ap

single quotes: i' and a'

double quotes: i" and a"

back ticks: i` and a`

parenthesis: i( and a(

brackets: i[ and a[

braces: i{ and a{

tags: it and at

FWIW, the ones I use the most are word, double quote, and tag.

## USING VISUAL MODE



Many tricks of the vim wizard can attract attention, but few create as many pleasurable expletives as skillful

use of Visual Mode. Perhaps the best thing to say about Visual Mode is that it magnifies the power of everything you've learned so far. It does this by allowing you to apply commands to the text that's currently highlighted. So let's start with how to enter Visual Mode and light up some text. You enter Visual Mode with the "v" key, and there are three different options.

character-based: v

line-based: V

paragraphs: Ctrl-v

### Selecting inside containers

Often time you'll be inside some content that is surrounded on both sides by something, such as , . ( { [. You can visually select these things by issuing commands like these:

# Select inside of parenthesis

vi(

# Select inside of brackets

vi[

You can also add a number to that to select two levels out (if you're inside a nested set.

# Select everything inside the second tier braces

v2i{

[ NOTE: You can also use va to select *around* instead of *inside*. Remember not to burden your mind with these. They're the same exact nouns and verbs we know from everywhere else. ]

### Character-based visual select

Starting with character-based (using v to enter from Normal Mode), you can use this to select characters, sets of characters, words, etc. I use this far less frequently than line-based (V), but I still use it often.

The main thing to understand here is that now that you're in Visual Mode, *your motions are changing what's being highlighted. This means you can*

*do motions like w or ) to expand your selection.* The highlighted area is then going to become the target for an action.

### Line-based visual select

You enter this mode by pressing the V key from Normal Mode, and from here you then take the actions we'll discuss in a moment.

### Column-based visual select

Another option is to select text vertically, which is great for pulling columns of data.

### Actions you can perform on visually selected text

It's really your choice, but the most common operations are simply deletion, copy, and paste. Just think of it as highlighting with your mouse—back when you used such things.

# Enter visual mode, select two more words of text, and copy them

vwwy

Then you simply go where you want to put them and type p to paste them there.

Or you can do some line-based action.

# Enter line-based visual mode and delete a couple of lines below

Vjld

You can also use text objects, which is seriously sick.

# Visually select an entire paragraph

vip

# Visually select an entire paragraph then paste it down below

vipyjip

Don't panic about how big that command is. Remember, it's language. You can rattle off:

I want to go to the store.

...without any problem, and it's the same with:

Copy the paragraph, move down two lines, and paste it.

### Combining visual mode with repetition

Another wicked thing you can do with Visual Mode is apply the .command to execute a stored action against the selection. Let's take the text below for example.

foo  
bar  
thing  
other  
yetanother  
also

If we want to prepend a colon in front of every line, you can simply put one in front of foo, visually select all the lines below it, and then hit the . key.

:foo  
:bar  
:thing  
:other  
:yetanother  
:also

[ NOTE: One thing that makes this possible is having vnoremap . :norm.<CR> in my ~/.vimrc. ]

BAM!

Not feeling it yet? How about this: your file is 60,000 lines, each with a line like the above, and you have to append the ":" to each of them. What do you do?

# Add the colon to the whole file

Oi:j0vG.

wut

Ease up, killer. Here are the steps:

Go to the beginning of the first line and insert a colon

Go down one line and go to the beginning of the line



Visually select all the way down the end of the file

Add the colon to the selection

Done. For the entire file. And remember, you're not going to have to remember to type "ALPHABET AMPERSAND GOBBLYGOOK 25" — no, it's just going to come to you, like falling off a bike. Trust me.

## USING MACROS

People think macros are scary. They're really not. They really come down to one thing: recording EVERYTHING you do and then doing it again when you replay. Here's a simple reference:

qa: start recording a macro named "a"

q: stop recording

@a: play back the macro

Simple, right? You can have multiple macros stored in multiple registers, e.g. "a", "b", "c", whatever. And then you just play them back with @a or @c or whatever.

## Why macros

You may be asking:

If visual selection and repetition with the dot command are so powerful, why use macros at all?"

Great question, and the answer is complexity. Macros can do just about anything you can do, so check out this workflow:

Search within the line for "widget"

Go to the end of the word and add "-maker"

Go to the beginning of the line and add a colon

Go to the end of the line and add a period.

Delete any empty spaces at the end of the line.

That's a lot of work, and if your file is 60K lines like the last one, it's going to

be somewhat painful. Try doing that in Microsoft Word, for example.

With vim, however, you simply perform those actions once and then replay it on each line.

[ NOTE: You can actually replay a macro on a visual selection by executing :normal @a(or whatever your macro register is) which will temporarily switch you into normal mode, for each line, and then execute the macro there. ]

## TRICKS

Let's go through a few tasks that get asked about a lot and/or just save a considerable amount of time.

## REMOVE WHITESPACE FROM AT END OF A LINE

Based on the type of file you're in, you might have some [line drama](#). Here's how to delete those annoying Ctrl-M characters from the end of your lines.

# Delete the Ctrl-M characters from the end of files

```
:%s/\s\+$//
```

## CHANGING FILE TYPE

```
set ft=unix
```

```
set ft=html
```

```
set ft=dos
```

[ NOTE: To show the current filetype, run or put :set filetype into your ~/.vimrc ]

## WRAPPING CONTENT

Using the [Surround](#) Plugin you can do some seriously epic stuff in terms of wrapping text with markup.

cs'': for the word you're on, change the surrounding quotes from double to single.

cs'<q>: do the same, but change the single quotes to <q>

ds'': delete the double quotes around something

ysiw[: surround the current word with brackets

ysiw<em>: emphasize the current word (it works with text objects!) Want to know what's crazier about that? It's dot repeatable!.

Visual Mode: select anything, and then type S. You'll be brought to the bottom of the window. Now type in what you want to wrap that with, such as <a href="/images">, and then press enter.

## CONCLUSION

So that's it then. There are two things I'd like one to come away with from this guide:

- vim is learnable
- vim is powerful

If you are able to become even partially comfortable with the basics covered here I think you will simply enjoy text more—and that's not a minor thing. The more comfortable you are dealing with text, the more comfortable you'll be dealing with ideas, and I think that's nothing less than epic.

More than anything else, *this* is why you should be competent with your text editor. You want to feel native and powerful when capturing ideas—not hobbled or encumbered.

Or you can sweep all that rubbish aside and just be one of those people who make others smile orgasmically when they watch you edit a config file—either way, I hope you found this helpful.

# FREE READING

## SDJOURNAL

# Using Python Fabric to AGNU/Linux Server Configuration Tasks

### What you will learn...

In this article, we will run through the basics of using Fabric to run local and remote commands and transfer files between servers. Through some examples, we will try to show some of its capabilities and use cases.

### What you should know...

You are not required to have any prior experience to follow this article. However, if you are used to performing SSH system administration or application deployment tasks, it will be easier to realize some practical use cases.

### About the author...

Renato Candido is a free (as in freedom) {software, hardware, and culture} enthusiast, who works as a technology consultant at Liria Technology, Brazil. He tries to solve technical's problems using these sorts of tools (he thinks the world would be a little better if all resources were free, as in freedom). He is an electronics engineer and enjoys learning things related to signal processing and computer science (and he thinks that there will be self-driving cars and speaking robots designed exclusively with free resources). Contact the author on

<http://www.renatocandido.org>

[www.sdjournal.org](http://www.sdjournal.org)

## Interview with Babar Khan Akhunzada

**Hello Babar, how have you been doing? Can you introduce yourself to our Readers?**

My name is Babar Khan Akhunzada. I am a seventeen years old lad from Pakistan, a college student and the Founder of SecurityWall, a Cyber Security based startup.

**Tell us something about your company, Security Wall...**

Basically, SecurityWall is a team of young Security Researchers & Security Experts. They have been working in this field from years, and also acknowledged by many high profile companies on the basis of their security knowledge. Last year in September, we participated in StartupIstanbul. This is a well-known conference in Istanbul, Turkey. There, 100 Startups were invited to participate in a competition and SecurityWall stood as finalists. Moreover, it was the youngest team to ever participate. As a result, we received huge support from everyone in Istanbul and Pakistan.



**Can you tell us more about this competition? What kind of challenges did you face there?**

StartupIstanbul is one of the great startups events around EU. Mostly, more developed startups are allowed to participate since it is not easy to get into the competition and then qualify for the next stage. It was a great feeling representing Pakistan in the competition, and being there as the only cyber security startup. People loved our AI products ideas. As a matter of fact, many startups and organizations acquired our services for their products' penetration. From that acquisition, we benefited in terms of traction and helped us in establishing good grounds in EU.

**You have a broad range of services. Which one is the most popular?**

We offer Vulnerability Assessment, Penetration Testing, Data Analysis, Malware Protection, Bug Fixing, Web Optimization. The most popular services we offer until now are Penetration Testing, Bug Fixing and Malware Protection. This is because, nowadays, most of the applications developers are unaware of new attacks, new methodology and techniques into hacking.

**Is your company based in Pakistan or in USA?**

We are based in Pakistan. Nevertheless, we have clients from US, UAE, Turkey, and some from Europe. Our mission is to spread our services globally through partnering. In doing so, we can reach anyone who seeks security testers and wants to be secure in this vulnerable cyber world.

**Tell us more about this partnering. Are you planning on finding other startup companies from Pakistan to work together?**

Yeah! We would love to be part of other startups, not only locally but also internationally to contribute to the cyber world. We want to make a crowd of hackers by making cool partnerships since most of the time; some teams have experience in some part of security. SecurityWall team has partnered with international startups which are mostly based on network security. We offer them our application consultation and in return, they build our Network. Therefore, it is a win-win relationship.

**You have been working for the biggest companies, like Apple, Yahoo! and Sony. But your experience with them is not very long. Have you been working with them as a freelancer?**

We were not working as an employee. However, it was a type of work where we were required to secure their applications. The companies' record an influx of users and customers over there, and many black hat hackers are also active to attack on such big names in the industry. Therefore, the task entailed looking for security bugs and vulnerabilities in their systems' applications to make internet a safe place for all. Later on after reporting the vulnerabilities, they appreciated our work and listed us as a security contributor to their Hall of Fame list.

**Did you enjoy that kind of work, looking for security bugs and vulnerabilities?**

Believe me; it evokes a kind of satisfaction all the time. I can even sit and work for hours. This is what I suggest to beginners, do what you love. But then, they ask how does one know what they love. I advise them that it is the thing you can do for hours, the task you can do while you are hungry. Being thirsty is your passion, and this is the thing you need to stick by. It isn't important if it is hacking, or it can be in any profession.

**You are The Youngest Pakistan Security Researcher Who Helped World Top Companies. What does it mean? Who gives this reward?**

I am not the only one, there are many guys who are working in the security field and doing great in Pakistan. I started this when I was in school, made contributions to both Adobe and eBay for the first time at the age of fourteen. To me, it was a start which helped me to move to greater ranks. To be a security researcher (or any sort of researcher), you need to pick a security subject and ace it. Learn every single aspect about the subject, and in the event that you investigate this point sufficiently, you will discover something new. To land a position as an infiltration analyzer, you have to demonstrate the ability to break programming. If you don't know how to break programming in any case or you can't carry out the employment, then you won't be enlisted. Currently, the web applications in the security business are above all in other sectors, they nearly took after mobile applications. Ace the OWASP Best 10, hack DVWA and chase for bugs in open-source web applications. Compose misuses for these imperfections and report them to developers or companies.

**What do you think about cyber security industry in Pakistan? Is it growing? Are people getting more and more interested in cyber threats or not yet?**

The global market of Cyber Security has been estimated to be worth \$170.21 Billion by 2020. This estimated growth will be as a result of the new technologies and new techniques since no one feels secure in this digital era. Pakistan is also growing because some great organizations are working for Cyber Security education, awareness to people and companies. The main problem we are facing is lack of awareness of Cyber Security in the industry. Usually, they come to know about it when they get breached or affected by hackers. Now, many techies from Pakistan are taking an interest in it and spreading awareness of it. Amazingly, youngsters are focusing and taking part in security which is a green signal. The thing which lacks here is the government taking no interest in cyber security field. If everyone plays their part, Pakistan will take a much greater role in the Cyber Security industry.

**What would you say to all these youngsters who are focusing on cyber security? What kind of advice would you give them?**

First, my advice to them would be to keep on learning, they should not expect to learn all stuff within a day, a week, a month or a year. It is a path of learning a whole new skill. So, all you guys have to do is to establish good grounds there.

**Your company is a startup. Is Pakistan a startup/entrepreneurial friendly?**

Yes, we are. We are focusing on Cyber Security which is missing. I hope we and others who took part in changing its shape will play a vital role in it. I will say a big "YES", because everything has changed from the past. If we talk about the industry, everything has a digital shape and everyone is focusing on entrepreneurship and looking for brilliant ideas to fulfill the need. We can judge that potential from a statistic revealed by Freelancer.com. The site declared that Pakistanis were the third-biggest user base of its platform, closely trailed by India and the United States.

**What are the challenges your company has to face at the moment? Any plans for the future?**

The lack of awareness be it in public, industry, organizations, companies, employees etc., and that's the challenge everyone has to work on. We at SecurityWall conduct seminars and encourage events' participation in Digital conferences for the audience to be aware of Cyber Security. Most of the time, universities officials take interest and invite us to raise awareness of Cyber Security to students. That is how Security grows and how they can choose Cyber Security as a profession. Until now, many students are working on it.

**Any piece of advice for our readers?**

If anyone is interested and has passion in any field of Cyber Security, I suggest you do what you love. Work not on all but focus on one and be a master, increase your coding skill, think logically and connect with cyber security experts, ensure to follow their instructions to the latter.

**Thank you**



*The personal computer user interface has come a long way since the day of toggle switches and LED's. Despite all the technological advances, end users frequently still struggle operating software despite displays that support millions of colors and innovative software controls. What is the secret to a good GUI?*

*Rob Somerville*

First of all, a confession. I am unrepentant, died in the wool, stoically intransigent CLI user. I probably only use 5% of the functionality of my word-processor. Apart from a few specialized applications I regularly use like graphics and desktop publishing, if I cannot do what I need at the shell prompt or via a web browser, a general sense of foreboding descends upon me when I need to learn to use a new application. So much of this is because I know – irrespective of the operating system or hardware platform – that I will have not one, but two mountains to climb. Firstly, I will have to get my head around what the software does and doesn't do (functionality), and secondly, I will need to understand where in the myriad of menus, popups, forms and dialogue boxes that the functionality actually lives, if it actually does exist at all. I must admit, over the years I am much more smitten with the quality of web interfaces rather than desktop designs, as the web truly is point and click – and straightforward.

The historical response to this is usually a dismissive RTFM (Read The Friendly Manual). However, with high-quality documentation pertaining to the user interface generally being the exception rather than the rule, after a cursory glance, I normally abandon this approach. The number of times I have encountered software where last minute revisions have not made it to the manual are too numerous to mention. So strike one to the CLI, man and info are your friends. Some \*nix commands are even flexible enough if you get the switches the wrong way round (and not follow the example to be found at the start of the man page) to parse your erroneous request correctly, or as a worse case to report something vaguely sensible back. So once

your application is written, at the very least, put some screenshots and user documentation together covering the most common (and not so common) use cases, or better still, an online video. New users will thank you for it.

The other hurdle is consistency. Unless your application is particularly flash or you are designing as part of a team, you will generally be responsible for the color scheme, fonts used, size of dialogue boxes, form fields, etc. Random sizes etc. do not inspire confidence. Pick a set of standards and stick to them. Error messages should be clear and right at the top of the form, unless you have the pleasure of using callbacks where you can highlight the offending field. If you are not validating user input, you are making a rod for your own back – invalid data input (or poor form logic) will generate a lot of support calls. Make messages informative - "Error code 76 at line 12275" might mean something to you – but to an end user on a steep learning curve and a tight deadline, this may mean printing off a copy of your online profile and throwing darts at it.

Complexity is a big problem with large applications. How do you incorporate a lot of functionality into a straightforward UI, especially if the screen estate is at a minimum? One useful design tip is to have a standard and expert mode. The former being a condensed and functional subset of the latter that is pre-populated with the most common defaults. Form design can be simplified by keeping the most commonly used functions on display, and allowing the user to add the relevant controls and additional functionality if required.

Icons, if used, must have a working title that is displayed when hovered over, better still, give the user a choice of text or icons. Often, icons do not accurately represent the functionality they perform. Keep clutter to a minimum. If there are lots of toolbars, split the functionality into groups and allow the user to show or hide the operations they most use.

Even before you start sketching out a design on paper, it is important to grasp how the intended audience thinks and works. An intuitive interface for Application A will be very different from Application B. People have different workflows depending on the task at hand, so resist the temptation to port a good design from one application to another. While the fundamentals will remain the same, subtle differences that make life easier may mean major re-coding – it is often better to start from scratch with a good brief and a clear understanding of what makes an excellent environment than trying to bend something else to fit. After all, the controls for a Boeing 747 are very different from a radio controlled drone. While both sets of controls fly planes, one is considerably more intuitive than the other. Always consider that what appears intuitive and second nature to you may not apply to your user base.

How resilient is your application interface? The GUI is the gateway between the end user and the chaos of software and technology beneath. The best analogy I can think of from a defensive programming standpoint is that the GUI is like a bouncer or steward outside a rowdy nightclub on a Friday night. Helpful, informative, authoritative but firm and fair. It is the GUI's responsibility to ensure that no undesirables get past, eject any troublemakers that do, but most importantly to ensure everyone has a good night out. Sadly, you cannot legislate or code against fools, but the more effort you put into this the higher the quality your application will be. Depending on the tool-kit you are using and the specific application you are designing, the onus for input validation may reside outside any form controls, and you may have to write extensive error trapping routines.

Finally, test, test, test and test again. Try to break your design at every stage and iron out any flaws. Even before you start writing code per se, sit a sample of users down in front of your GUI, and listen to their feedback and ideas. Get continual feedback throughout the major milestones of the project. They are the people you are trying to make life easy for, and you will be surprised with the challenges that they often raise. Once you have successfully repeated this, you can proudly add the most important part of the UI – the Easter egg with your copyright notice and prevailing wisdom.

# MAGAZINE BSD

## **Editor in Chief:**

Ewa Dudzic  
[ewa@bsdmag.org](mailto:ewa@bsdmag.org)  
[www.bsdmag.org](http://www.bsdmag.org)

## **Contributing:**

Renan Dias, Rob Somerville, Hubert Feyrer, Kalin Staykov, Manuel Daza, Abdorrahman Homaei, Amit Chugh, Mohamed Farag, Bob Cromwell, David Rodriguez, Carlos Antonio Neira Bustos, Antonio Francesco Gentile, Randy Ramirez, Vishal Lambe, Mikhail Zakharov, Pedro Giffuni, David Carlier, Albert Hui, Marcus Shmitt, Aryeh Friedman

## **Top Betatesters & Proofreaders:**

Daniel Cialdella Converti, Eric De La Cruz Lugo, Radjis Mahangoe, Daniel LaFlamme, Steven Wierckx, Denise Ebery, Eric Geissinger, Luca Ferrari, Imad Soltani, Olaoluwa Omokanwaye, Radjis Mahangoe, Katherine Dizon and Mark VonFange.

## **Special Thanks:**

**Denise Ebery**  
**Annie Zhang**  
**Katherine Dizon**

## **Senior Consultant/Publisher:**

Paweł Marciniak

## **Publisher:**

Hakin9 Media SK,  
02-676 Warsaw, Poland Postepu 17D Poland worldwide  
publishing [editors@bsdmag.org](mailto:editors@bsdmag.org)

Hakin9 Media SK is looking for partners from all over the world. If you are interested in cooperation with us, please contact us via e-mail: [editors@bsdmag.org](mailto:editors@bsdmag.org)

All trademarks presented in the magazine were used only for informative purposes. All rights to trademarks presented in the magazine are reserved by the companies which own them.



## Rack-mount networking server

Designed for BSD and Linux Systems



Designed. Certified. Supported

Up to **5.5Gbit/s**  
routing power!

### KEY FEATURES

- ▶ 6 NICs w/ Intel igb(4) driver w/ bypass
- ▶ Hand-picked server chipsets
- ▶ Netmap Ready (FreeBSD & pfSense)
- ▶ Up to 14 Gigabit expansion ports
- ▶ Up to 4x10GbE SFP+ expansion

### PERFECT FOR

- ▶ BGP & OSPF routing
- ▶ Firewall & UTM Security Appliances
- ▶ Intrusion Detection & WAF
- ▶ CDN & Web Cache / Proxy
- ▶ E-mail Server & SMTP Filtering



# Among clouds Performance and Reliability is **critical**

Download syslog-ng Premium Edition  
product evaluation [here](#)

Attend to a free logging tech webinar [here](#)



**BalaBit**  
IT Security

[www.balabit.com](http://www.balabit.com)

## **syslog-ng log server**

The world's first High-Speed Reliable Logging™ technology

### **HIGH-SPEED RELIABLE LOGGING**

- above 500 000 messages per second
- zero message loss due to the  
Reliable Log Transfer Protocol™
- trusted log transfer and storage