**Install ELK On Ubuntu 18.04 Bionic Beaver Linux**

**For ELK SYSMON**

Original:

<https://linuxconfig.org/install-elk-on-ubuntu-18-04-bionic-beaver-linux>

**Modifications: JJK made several changes to prepare it for receiving Winlogbeat data from several Window 7 and Windows Server 2012 computers.**

**What is ELK**

If you're in a situation where you manage large amounts of data logs, the ELK stack is exactly what you're looking for. The ELK stack combines Elasticsearch, Logstash, and Kibana into a simple, yet powerful, open source stack that lets you manage large amounts of logged data from a convenient graphical web interface. All three tools are developed by Elastic, so they work in tandem perfectly, and they're very easy to get set up on your Ubuntu system.

**Install the Dependencies**

Begin by installing the dependencies. These are all fairly common, though there are a couple of notes that you need to take into account. Of course, these are Nginx-based, so disable Apache or switch the port, if you're using it.

An Ubuntu 18.04 server set up including a non-root user with sudo privileges and a firewall configured with ufw. The amount of CPU, RAM, and storage that your Elastic Stack server will require depends on the volume of logs that you intend to gather. For this tutorial, we will be using the following specifications for our Elastic Stack server:

**Install Ubuntu 18.04 LTS**

**Download the Desktop ISO via** [**https://ubuntu.com/download/desktop/thank-you?version=18.04.3&architecture=amd64**](https://ubuntu.com/download/desktop/thank-you?version=18.04.3&architecture=amd64)

**We will be installing the VM into a VMware vSphere cluster. Use 2 vCPUs, 8GB RAM, at least 300GB vDisk, and put the VM on the same network (LAN) as the Windows PCs that will be sending in data through Winlogbeat.**

**Set Static IP**

#### Ubuntu Server

To configure a static IP address on your Ubuntu 18.04 server you need to modify a relevant netplan network configuration file within /etc/netplan/ directory. The file is 01-network-manager-all.yaml:

# This file describes the network interfaces available on your system

# For more information, see netplan(5).

network:

version: 2

renderer: networkd

ethernets:

ens160:

dhcp4: no

addresses: [192.168.1.158/24]

gateway4: 192.168.1.1

nameservers:

addresses: [192.168.1.248,192.168.1.249]

Once ready apply changes with:

$ sudo netplan apply

In case you run into some issues execute:

$ sudo netplan --debug apply

Logstash doesn't support Java 10, which is available on Bionic from openjdk-11-jre. If you have it installed on your system, remove it. Use the older version until Logstash gets support.

$ sudo apt install openjdk-8-jre apt-transport-https wget nginx

**Add the Elastic Repository**

Elastic provides a complete repository for Debian based systems that includes all three pieces of software. You just need to add it to your system. Begin by importing their GPG key.

wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -

Next, add the repository. Create a file at /etc/apt/sources.list.d/elastic.list, and paste the following line into it.

deb https://artifacts.elastic.co/packages/6.x/apt stable main

Save that file, and exit. Update Apt.

$ sudo apt update

**Install Elasticsearch and Kibana**

You're now ready to install Elasticsearch and Kibana. They're available through Apt, so get them like you normally would.

$ sudo apt install elasticsearch kibana

You need to edit the Kibana configuration file at /etc/kibana/kibana.yml to tell it that the host server is localhost. The line is already there. Uncomment it.

server.host: "localhost"

Restart Kibana and start up Elasticsearch, and both will be ready to go.

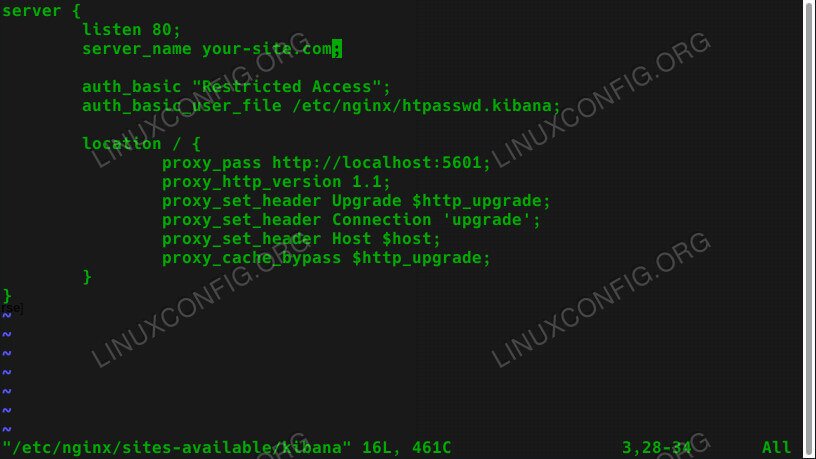
$ sudo systemctl restart kibana

$ sudo systemctl start elasticsearch

**Set Up Nginx**

Kibana is served through Nginx, so you need to set up a basic Nginx configuration to get it to serve your instance of Kibana. Start by creating a password for Kibana. This way, your server isn't accessible openly on the Internet. Use OpenSSL to generate the password, and place it in/etc/nginx/htpasswd.kibana. You can change the username to anything you want. In this instance, it'sadmin.

$ echo "admin:`openssl passwd -apr1 <password goes here>`" | sudo tee -a /etc/nginx/htpasswd.kibana

[](https://linuxconfig.org/images/ubuntu-18.04-elk-nginx-config.jpg)

**Kibana Nginx Configuration On Ubuntu 18.04**

After you have your password, create an Nginx configuration similar to the one below at /etc/nginx/sites-available. Make sure to use your actual server url or IP. The defaults should be good for everything else.

server {

listen 80;

server\_name your-site.com;

auth\_basic "Restricted Access";

auth\_basic\_user\_file /etc/nginx/htpasswd.kibana;

location / {

proxy\_pass http://localhost:5601;

proxy\_http\_version 1.1;

proxy\_set\_header Upgrade $http\_upgrade;

proxy\_set\_header Connection 'upgrade';

proxy\_set\_header Host $host;

proxy\_cache\_bypass $http\_upgrade;

}

}

After you have your configuration, remove the existing default config, and create a new symlink in sites-enabled for Kibana.

$ sudo rm /etc/nginx/sites-enabled/default

$ sudo ln -s /etc/nginx/sites-available/kibana /etc/nginx/sites-enabled/kibana

Restart Nginx for the changes to take effect.

$ sudo systemctl restart nginx

**Install Logstash**

The last thing that you need to do is install Logstash. Just install it with your package manager.

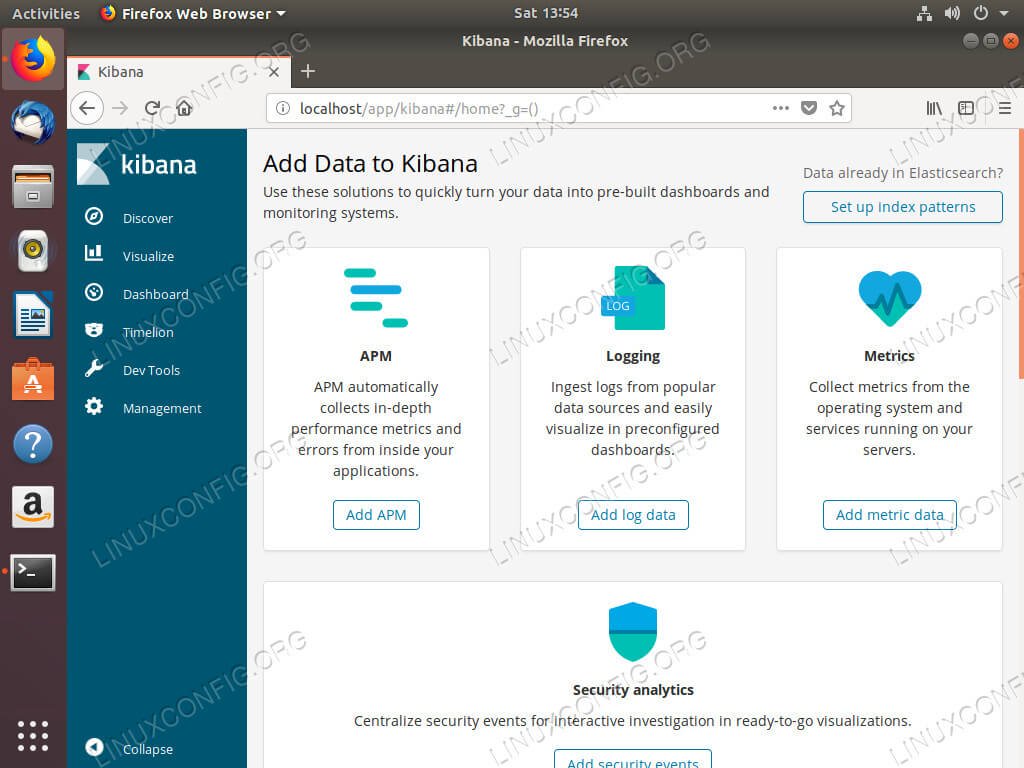
$ sudo apt install logstash

$ sudo systemctl start logstash

$ sudo systemctl enable logstash

**Sign In To Kibana**

Open up your browser, and go to the address that you assigned to your Kibana instance in the Nginx configuration. You should be prompted to enter the username and password that you set up for Kibana. Enter them.

[](https://linuxconfig.org/images/ubuntu-18.04-elk-kibana.jpg)

**Kibana Running On Ubuntu 18.04**

Once you performed the configuration in the Appendices you will see data showing up in Kibana. You can begin using Kibana and setting up your preferences.

Some example dashboards are included in the Appendices.

**Appendix A**

**Logstash Config Files**

**Path /etc/logstash/conf.d/**

**File: 02-beats-input.conf:**

output {

elasticsearch {

hosts => [“localhost:9200”]

sniffing => true

manage\_template => false

index => “%{[@metadata][beat]}=%{+YYYY.MM.dd}”

document\_type => “%{[@metadata][type]}”

}

}

**File: 30-elasticsearch-output.conf:**

output {

beats {

port => 5044

ssl => false

# ssl\_certificate => “/etc/pki/tls/certs/ELK-Stack.crt”

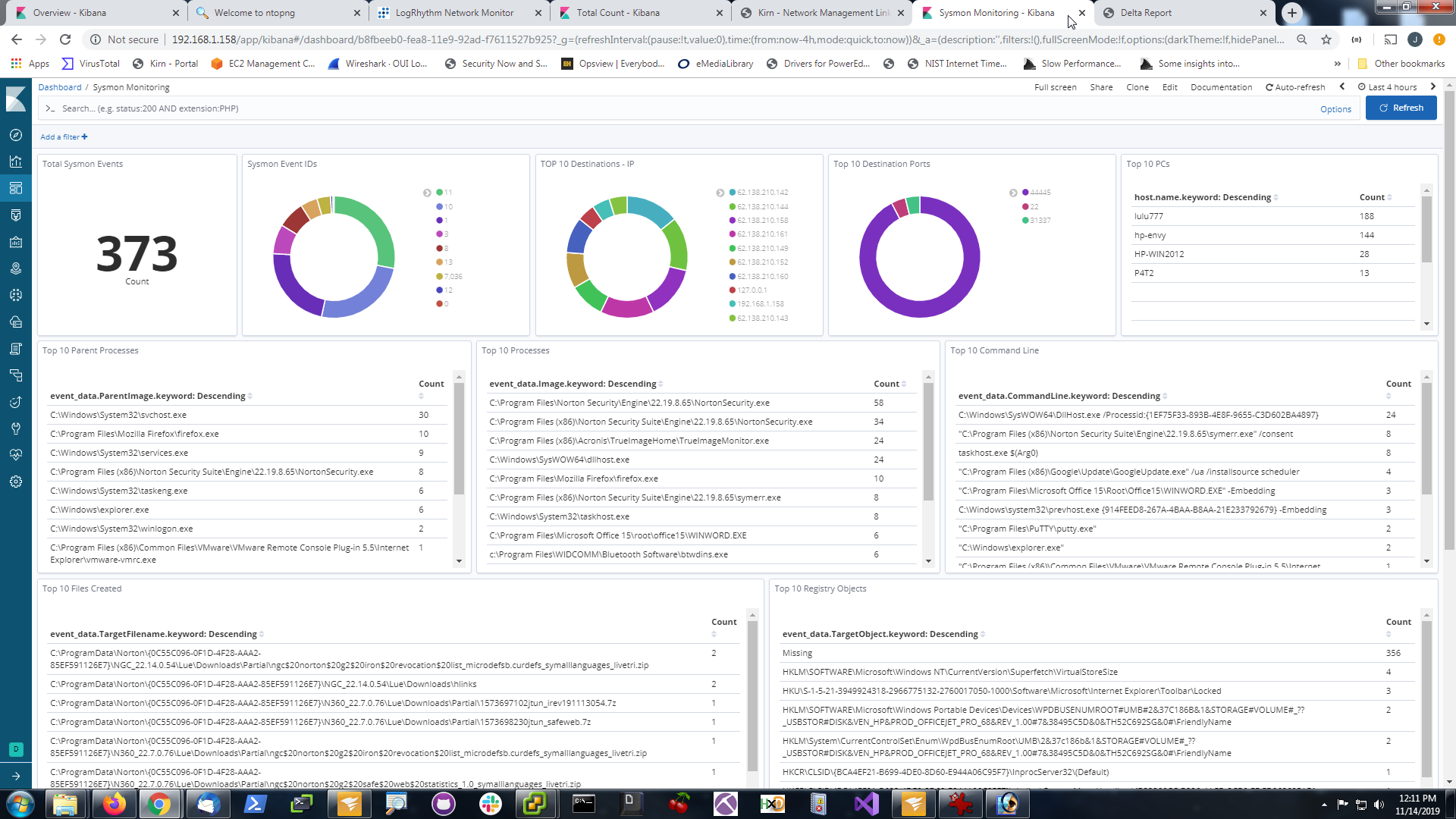
# ssl\_key => “/etc/’pki/private/ELK-Stack.key”

}

}

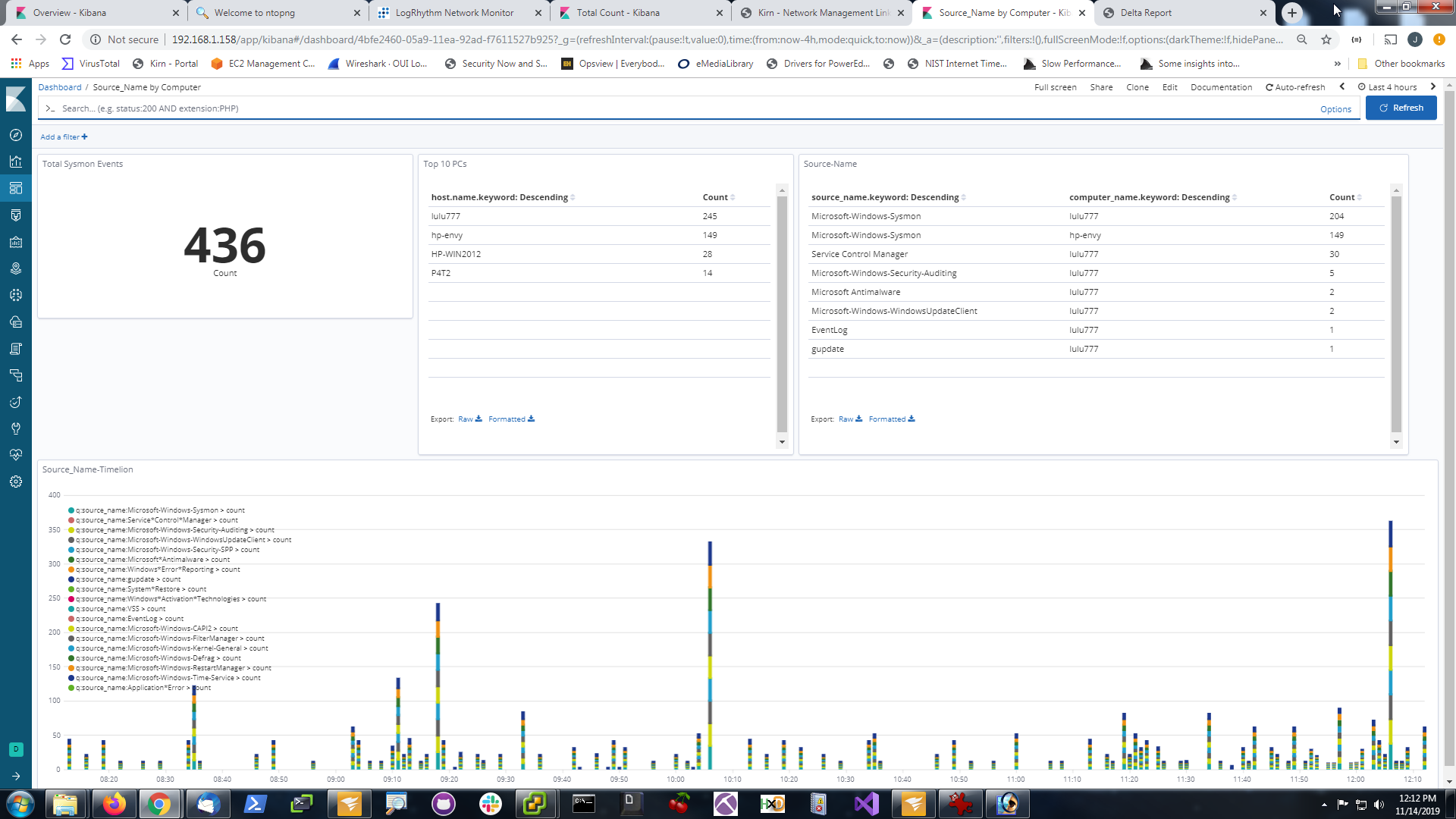
**Appendix B**

**Sysmon Monitoring – Dashboard**



**Appendix C**

**Source\_Name by Computer – Dashboard**



**Appendix D**

**Source\_Name –Timelion visualization creation:**

Copy the below to the “Timelion Expression” panel to get a running timelion of stacked events:

.es(source\_name:Microsoft-Windows-Sysmon).bars(stack=true),

.es(source\_name:Service\*Control\*Manager).bars(stack=true),

.es(source\_name:Microsoft-Windows-Security-Auditing).bars(stack=true),

.es(source\_name:Microsoft-Windows-WindowsUpdateClient).bars(stack=true),

.es(source\_name:Microsoft-Windows-Security-SPP).bars(stack=true),

.es(source\_name:Microsoft\*Antimalware).bars(stack=true),

.es(source\_name:Windows\*Error\*Reporting).bars(stack=true),

.es(source\_name:gupdate).bars(stack=true),

.es(source\_name:System\*Restore).bars(stack=true),

.es(source\_name:Windows\*Activation\*Technologies).bars(stack=true),

.es(source\_name:VSS).bars(stack=true),

.es(source\_name:EventLog).bars(stack=true),

.es(source\_name:Microsoft-Windows-CAPI2).bars(stack=true),

.es(source\_name:Microsoft-Windows-FilterManager).bars(stack=true),

.es(source\_name:Microsoft-Windows-Kernel-General).bars(stack=true),

.es(source\_name:Microsoft-Windows-Defrag).bars(stack=true),

.es(source\_name:Microsoft-Windows-RestartManager).bars(stack=true),

.es(source\_name:Microsoft-Windows-Time-Service).bars(stack=true),

.es(source\_name:Application\*Error).bars(stack=true)

