Breaking encryptions using GPU accelerated cloud instances.

Thomas Roth

January 6, 2011

Contents

Author

Thomas Roth - https://stacksmashing.net/ - input@stacksmashing.net

1 About GPU accelerated computing in the cloud

On the November 15, 2010 Amazon launched what they call "Cluster GPU Instances", an instance type in the Amazon Elastic Computing Cloud (EC2) that is equipped with two NVIDIA Tesla "Fermi" M2050 computing modules. The M2050 is a PCI-Express 16x slotcard which is equipped with a graphic processing unit (GPU). Even though GPUs were originally developed for speeding up certain parts of the rendering pipeline, they evolved into very flexible and programmable devices that can also be used for high performance general purpose computing (GPGPU - General purpose computing on graphic processing units). What's special about GPUs is that they are entirely developed for parallel computing - because that's what rendering a screen of pixels is all about. Modern high-end GPUs feature more than 400 cores which can be used to accelerate a lot of applications in the science sector, in the medical sector, in the financial sector and in the encryption sector - practically everywhere where the same program needs to run over a large amount of independent data elements.

With the launch of this new instance type Amazon provides the possibility to boot up a cluster of GPU accelerated computers with just a few clicks and with a price of $2.10/h for every instance.

2 Specifications

These are the technical specifications of the cloud instance and the computing module as of January 6, 2011.

2.1 Specifications of the "Cluster GPU" instance

Cluster GPU instance specification[1]:

- 22 GB of memory
- 33.5 EC2 Compute Units (2 x Intel Xeon X5570, quad-core "Nehalem" architecture)
- 2 x NVIDIA Tesla "Fermi" M2050 GPUs
- 1690 GB of instance storage
- 64-bit platform
- I/O Performance: Very High (10 Gigabit Ethernet)
- API name: cg1.4xlarge

2.2 Specifications of the M2050 GPU

NVIDIA Tesla "Fermi" M2050 GPU specification[2]:

- 448 CUDA Cores
- 3GB of GDDR5 memory (at 1.55GHz, 384-bit interface)
- 148 GB/sec memory bandwidth
- 225W TDP

2.3 Benchmarking with Pyrit

Pyrit[3] is a tool that allows the creation of massive databases with pre-computed pairwise master keys (PMKs) of the IEEE 802.11 WPA/WPA2-PSK authentication phase. It supports a lot of computing platforms like CUDA, OpenCL and VIA Padlock.

[1] https://aws.amazon.com/ec2/hpc-applications/
[2] http://www.nvidia.com/object/product_tesla_M2050_M2070_us.html
[3] http://code.google.com/p/pyrit/

2.3.1 Description

These benchmarks were done on a "Cluster GPU instance" and a "Cluster Compute instance". Booth have the same specifications besides the fact that the last one is not equipped with GPUs. For the benchmarks the machine image "ami-42a2532b", which is provided by Amazon as "Cluster Instances HVM CentOS 5.5", was used.

The Pyrit benchmarks were done using revision 288 from the official SVN repository. Pyrit and the CUDA support for Pyrit were used in this benchmark. The benchmark call was "pyrit benchmark_long".

2.3.2 Results

Cluster Compute Instance	PMKs/s	7003.52
Cluster GPU Instance	PMKs/s	47301.44
Factor:		6.75

3 The Cloud Cracking Suite

Note: Because of the early state of the cloud cracking suite there are no command line options and usage samples provided here, as they are subject to change very often right now.

3.1 Introduction to the Cloud Cracking Suite

The "Cloud Cracking Suite", also reffered to as CCS, is a tool for distributing and managing encryption breaking jobs in the Amazon EC2 cloud. It allows one to start up multiple instances in the cloud which are then running self-organized until the job is done.

The Cloud Cracking Suite consists of two parts, the ccs-server and the ccs-client:

3.1.1 Serverside

The ccs-server manages the cloud instance and supervises the actual cracking job as well as the communication and coordination with other instances that are working on the same job. It also configures the node after booting up for the job and gives informations about the progress and the state of the node back to the client.

The ccs-server is integrated into an Amazon Machine Image (AMI) and get's automatically launched after the startup.

3.1.2 Client

The css-client is used to start, show and stop jobs. It connects to the Amazon Web Services infrastructure and starts up EC2 instances, uploads data to S3 or communicates directly with nodes depending on what it should do.

It's important to understand that the client doesn't have to run while jobs are running, it can always reconnect to the job later to get the status or to stop the job.

3.2 The ccs-server

3.2.1 Description

The ccs-server is packed in an Amazon Machine Image containing several Python programs for the supervisor as well as cracking engines, default wordlists and other tools that might be used by the server for doing jobs.

3.2.2 Startup

The ccs-server is started via /etc/rc.local to ensure that it starts at the end of the boot progress. The first thing that it does is getting the *userdata* that was supplied by the client. Depending on the content of the data, it will setup the right cracking engine, download additional data and start the job. It also sets up an RPC server that is used to communicate with the client or other nodes, for instance to get the progress and the speed of a job.

3.2.3 During a job

During a job the server is controlling the cracking engine, ensures RPC communication, handles certain events (like successful encryption) and ensures that recovery data is spread onto the other nodes, so that if the job crashes it can still be resumed later using informations from other nodes.

3.2.4 On success or failure

On completion of a job the result is uploaded into an S3 bucket and a notification is sended to all other nodes that might be still working on the job notifying them about the successful job completion or about the unsuccessful job completion. The system tries to shutdown nodes as fast as possible to keep the costs as low as possible.

3.3 The ccs-client

The ccs-client is a command line program that's used to manage jobs.

3.3.1 Creating a job

Most of the time creating a job using the client is very easy. The client needs several informations to set up the instances:

- Instance Type, i.e. 'cg1.4xlarge' for cluster GPU instances
- Number of nodes
- Cracking engine to use
- Options to the cracking engine
- File to crack (Wi-Fi dump, Hash etc.)
- Additional data (Extra data for certain cracking engines)

- EC2 Credentials
- File to safe the job informations into

If all informations are valid, the client will generate the userdata that's passed to the instances and then try to launch the nodes. To ensure that all nodes are configured properly, the client waits until all instances are bootet up and verifies the configuration of the nodes.

3.3.2 Getting job informations

If the client is started in view mode, it will try to use the supplied job information file to connect to the nodes that are taking part in it and retrieve progress information.

3.4 Cracking engines

Cracking engines are the cipher implementations used by the CCS. They are controlled by the server using a Python API which every cracking engine has to provide. The cipher implementation might be in any language, as long as a Python module is provided for controlling it.

3.4.1 Default engines

- MD5 (incl. CUDA support)
- SHA1 (incl. CUDA support)
- NTLM (incl. CUDA support)
- WPA/WPA2-PSK (incl. CUDA support, based on Pyrit)

3.5 WPA-PSK Benchmark

This benchmark used 8 cluster GPU instances and a 39 million word dictionary to which the right key was passed at the end for testing purposes.

3.5.1 Results

Per node information:		
Node[0]	PMKs/s	47301.44
Node[1]	PMKs/s	49141.28
Node[2]	PMKs/s	45983.85
Node[3]	PMKs/s	50524.59
Node[4]	PMKs/s	48198.19
Node[5]	PMKs/s	47591.84
Node[6]	PMKs/s	45295.47
Node[7]	PMKs/s	49794.91
Sum	PMKs/s	383831.57

Seconds total: 132

4 Sources and further reading

- Cloud Cracking Suite: https://stacksmashing.net/cloud-cracking-suite/
- Amazon HPC Site: https://aws.amazon.com/ec2/hpc-applications/
- NVIDIA CUDA Zone: http://www.nvidia.com/object/cuda_home.html
- CUDA programming guide: http://developer.download.nvidia.com/compute/cuda/1_1/NVIDIA_CUDA_Programming_Guide_1.1.pdf
- The Pyrit project: http://code.google.com/p/pyrit/