# APT-IPFS

## Development of software using The InterPlantary File System for enhancing features of Linux package manager

**Wisit Tipchuen (wisit_t@sci.tu.ac.th) and Asst. Prof. Dr. Kasidit Chanchio**

Computer Sciences, Thammasat University, Thailand

## Abstract

Today Linux is commonly deployed on virtual machines in the cloud, IoT devices, and personal computers. Updating software on Linux usually requires downloading software packages from remote repositories. Downloading and updating software on multiple devices at the same time can cause heavy network usages and slow down software updating progress in an organization.

In this paper, we developed a new software tool, namely APT-IPFS, using The InterPlanetary File System (IPFS) for enhancing features of the apt Linux package manager to address the above challenges. IPFS is a distributed file system that collectively uses peer-to-peer resources to store data. APT-IPFS extends the Debien's APT tool by caching the newly downloaded packages on IPFS storages of an organization. Upon receiving a download request, APT-IPFS searches for the requested package in the IPFS storages first before retrieving it from a remote repository.

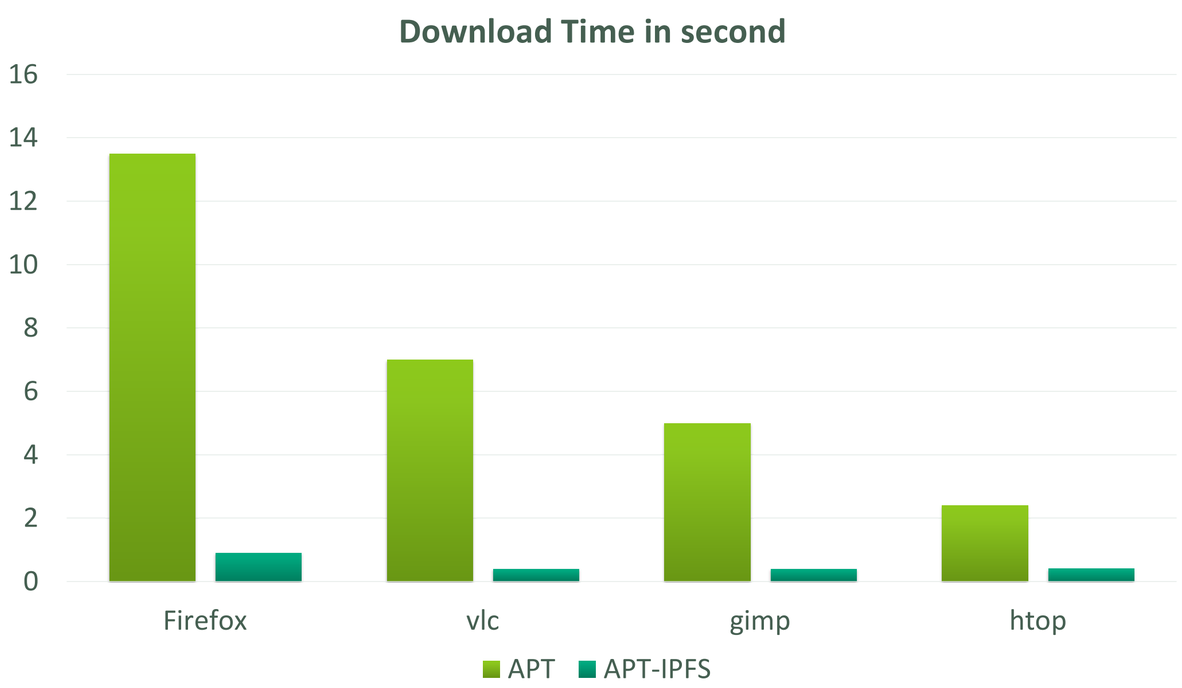
APT-IPFS has been implemented for the Debian-based distribution of Linux software packages. In this paper, we address the key design issues and discuss the self-hosted Linux software repository framework implemented using IPFS. We have also conducted a preliminary experiment to show the reductions of response times of APT-IPFS over the traditional APT software.

## Introduction

In a large organization, a significant amount network bandwidth must be used for software updates since many computers within the organization may perform software updates at the same time. For examples, the organization may have a private or public cloud that hosts a large number of virtual machines (VMs), containers, and bare medal servers. Moreover, the organization may operate a large number of IoT devices in its networks. In some cases, applications may consist of many software components or services, which require updates at the same time. As a result, the organization must pay a lot of money to provision high networking bandwidths for updating their applications. If the bandwidths are not enough, the updates can take a long time or fail. Since software updates compete for network bandwidths, they may also interfere with networking performances of the applications. To overcome these problems, a powerful caching solution is needed.

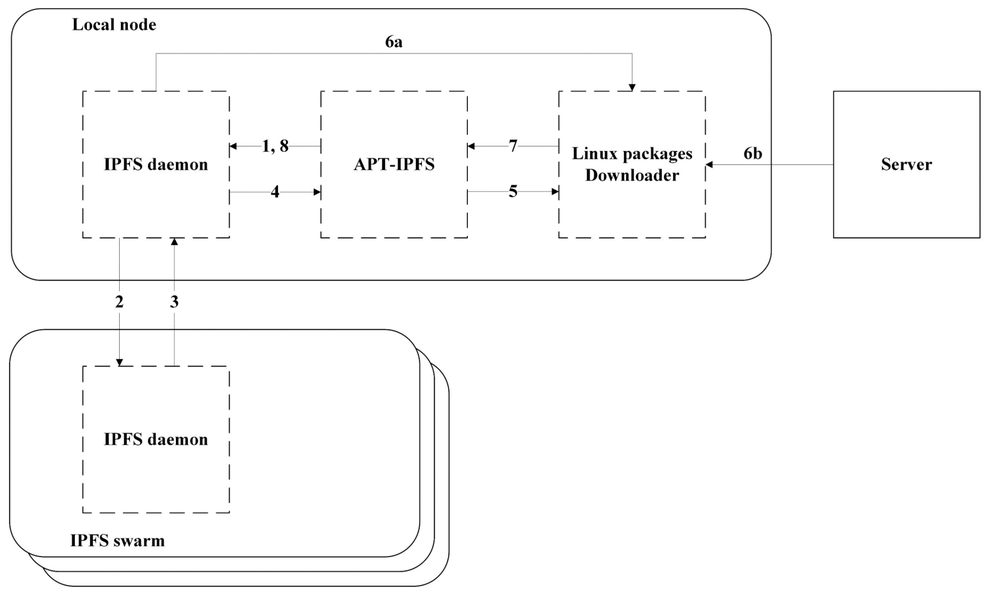
This paper proposed a new solution for caching software packages, namely APT-IPFS, based on the InterPanetary File System (IPFS). IPFS is a decentralized file system based on a peer-to-peer protocol. It adds contents by broadcasting a unique torrent file to every peer and starts to seed the file. IPFS could be seen as a single bittorent swarm which can be implemented in the network of the organization.

## Preliminary Performance Comparisons




We have conducted a preliminary evaluation by comparing APT-IPFS performance against that of the APT software. In the experiment, we uses APT and APT-IPFS to download 4 software packages as illustrated in the graph below. The packages are downloaded from the default ubuntu repository to a Docker container running ubuntu 16.04. In case of APT-IPFS, we assume that some users in an organization have already used APT-IPFS to download these packages to their containers. As a result, the packages are already available in IPFS. The IPFS swarm framework used in this experiment has its storage nodes in 5 containers. The APT-IPFS performance results reported in the graph show significant performance gains over APT when a user use APT-IPFS to download software that have been downloaded before.

## Proposed mechanism




In the Figure above, APT-IPFS is installed on a local computer (the "local node" in the Figure) in a network of an organization. This "local node" may be a virtual machine (VM), a container, a physical host, or an IoT device. The "server" represents a remote repository server on the Internet. Finally, the "IPFS swarm" represents the IPFS peer-to-peer framework in the organization.

The APT-IPFS mechanism operates as follows.

- APT-IPFS searches packages from IPFS swarm, a cluster of IPFS storage nodes, using IPFS daemon (Step 1 in the Figure).
- IPFS daemon lookups the cryptographic hash of the package in the IPFS swarm (2) and returns search result to the IPFS daemon (3). IPFS daemon, in turn, returns search result to APT-IPFS (4).
- Base on the search result, APT-IPFS decides to use either IPFS swarm or a remote APT repository server to download the packages and sends decision information to Linux packages downloader. (5)
- If the packages are founded in the IPFS swarm, they would be downloaded from multiple IPFS storage nodes in parallel using Linux packages downloader. (6a)
- Otherwise, IPFS falls back to download the packages from the remote APT repository server using Linux packages downloader (6b).
- After APT-IPFS receives the packages from Linux packages downloader (7), it publishes the packages to IPFS swarm (8) for future package searching and downloads.

## Conclusions

To this end, we have presented APT-IPFS a novel peer-to-peer distributor that sits between client and server, providing efficient and transparent downloading and updating services for software packages. We have addressed the key design issues and discuss the self-hosted Linux software repository framework implemented using IPFS. We have also conducted a preliminary experiment to show the reductions of response times and network usages of APT-IPFS over the traditional APT software. APT-IPFS has been used in conjunction with Debian-based distribution of Linux software packages and is also available in the latest release of Ubuntu. Existing user statistics have suggested that it interacts well with clients and substantially reduces server cost.

## References

[1] Benet, J. (2014). IPFS - Content Addressed, Versioned, P2P File System.
Retrieved from https://github.com/ipfs/papers/raw/master/ipfs-cap2pfs/ipfs-p2p-file-system.pdf

[2] Protocol Labs. (2017). IPFS is the Distributed Web. Retrieved April 11, 2017, from https://ipfs.io/

[3] Wang, L., & Kangasharju, J. (2013). Measuring large-scale distributed systems: case of bittorrent mainline dht. 2013 IEEE Thirteenth International Conference (pp. 1-10). IEEE.

[4] Mazieres, D., & Kaashoek, M. F. (1998). Escaping the evils of centralized control with self-certifying pathnames. n Proceedings of the 8th ACM SIGOPS European workshop on Support for composing distributed applications (pp. 118-125). ACM.

https://github.com/MasterTos/apt-ipfs