2D-06

使用済みパソコンを再利用した分散ストレージシステム

川戸　聡也†　　本村　真一‡　　東野　正幸‡　　川村　尚生†
鳥取大学大学院工学研究科†　　鳥取大学総合メディア基盤センター‡

1. はじめに

データを保存するためのストレージは必要不可欠な存在である．容量が不足する場合は新たなストレージの導入により対応可能だが，予算や資源の限られる状況では難しい．一方，本来は利用価値があるにもかかわらず利用されていない遊休資源が多く存在しており，資源が有効に活用されていない．身近では，機器の更新などにより用途がなくなり放置や廃棄されてしまう，本来であれば継続して利用可能な使用済みパソコンが挙げられる．

そこで本研究では，使用済みパソコンを分散ストレージとして再利用することを検討する．小容量，数多く利用可能といった使用済みパソコンの特性を踏まえ，既存のディスクを換装して必要な容量を確保した上で，数多くの台数を利用した導入費用が安価で可用性の高い分散ストレージシステムとすることを想定する．データを分散配置や冗長配置することで使用済みパソコンの故障に備え，高い可用性を実現する．ディスクを換装する場合には費用が発生するが，パソコン用のディスクはストレージ製品やサーバ向けのディスクに比べて安価に導入できる．

ここで，使用済みパソコンを分散ストレージとして再利用する上で必要な作業を全て手動で行うと，多大な労力を要してしまう．このため，使用済みパソコンに分散ストレージとして利用するための環境を自動的に構築するシステム（以下，「自動構築システム」という）を実装する．また，使用済みパソコンに対して分散ストレージとして利用するための環境を実際に構築し，実環境に組み込んで運用することで性能や導入費用の評価を行う．

2. 自動構築システム

使用済みパソコンを分散ストレージとして再利用する上で必要な以下の処理を自動化した．

2. 1　OS の自動インストール

使用済みパソコンを分散ストレージシステムのノードとして利用するために，Kickstart[1]とPXE ブートにより OS を新規にインストールすることとした．PXE ブートさせるためには使用済みパソコンに IP アドレスの設定が必要であるが，今回は固定 IP アドレスを自動的に払い出すスクリプトを作成した．構築した PXE サーバによる OS の自動インストールの流れを図 1 に示す．クライアントである使用済みパソコンから PXE ブートの要求が行われると，DHCP サーバが前述のスクリプトにより固定 IP アドレスを払い出す．その後，TFTP サーバからブートイメージが，HTTP サーバから OS のイメージや Kickstart 関連のファイルが読み込まれることで OS の自動インストールが実行される．これにより，使用済みパソコンをネットワークに接続して PXE ブートさせる作業のみで，OS のインストールが自動的に完了されるようにした．

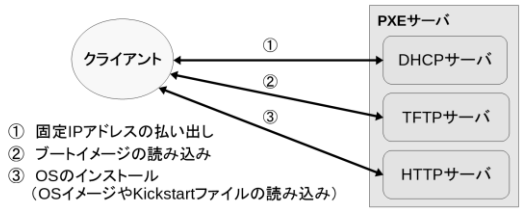

図 1　OS の自動インストール

2. 2　分散ストレージソフトウェアの自動インストール

分散ストレージの種類としては，データをオブジェクトという単位で管理するオブジェクトストレージとした．オブジェクトは保存場所を示す固有の識別子と，様々な情報を記録可能なメタデータが付与された上で，階層構造ではない平坦な空間に格納される．固有の識別子が実際に保存されるディスク上の場所に依存しないことで，データの配置に制約が少ない．このため，データの移動や分散配置が容易である．

オブジェクトストレージを構築するためのソフトウェアとして，OpenStack Swift[2]（以下，「Swift」という）を利用した．Swift の基本構成を図 2 に示す．オブジェクトはコンテナで，コンテナはアカウントで一覧を管理される．アカウントを担う Account Server，コンテナを担う Container Server，オブジェクトを格納する Object Server はまとめてストレージノードと呼ばれ，ゾーンという単位でグループ化される．ゾーンは複数作成することができ，それぞれに同じアカウント，コンテナ，オブジェクトが格納される

Distributed Storage System reusing Used-PCs
†Toshiya Kawato and Takao Kawamura, Department of Information and Electronics, Graduate School of Engineering, Tottori University
‡Shin-ichi Motomura and Masayuki Higashino, Center for Information Infrastructure & Multimedia, Tottori University

ため，冗長化や耐障害性の向上を実現できる．また，ストレージノードへの通信はプロキシノードと呼ばれる Proxy Server が中継する．

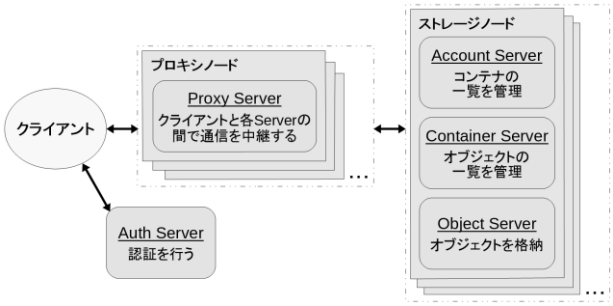

図 2　Swift の基本構成

Swift の自動インストールには Ansible[3]を利用した．プログラミングのようなコードの記述によりインフラ構築が可能であり，複数回実行しても同じ結果となる冪等性を持つ．構築対象で Python が動作し，且つ SSH 接続できれば利用可能であり，構成が簡素なため容易に導入できる．

まず，前述の PXE サーバと同一のサーバ上に Ansible をインストールした Ansible サーバを構築した．次に，Swift のインストールに必要な設定を記述した Playbook などを作成や配置することで，使用済みパソコンにオブジェクトストレージとして利用するための環境が自動的に構築されるようにした．なお，Auth Server と Proxy Server はシステム全体で最低 1 台あればよく，今回は自動構築の対象をストレージノードのみとした．なお，Account Server と Container Server についてはゾーンごとに最低 1 台あればよい．

3. 運用と評価

自動構築システムにより構築処理を行った使用済みパソコン(CPU：Core 2 Duo E7400 2.8GHz, メモリ：2GB, NIC：1Gbps, HDD：160GB(1 ディスク))15 台を，既存のオブジェクトストレージシステムに組み込み実際に運用した．使用済みパソコンでは全台において Object Server のみを稼働させた．組み込み後のシステム全体の Object Server の構成を図 3 に示す．Swift 1～5 は汎用のサーバであり，使用済みパソコンに比べて高性能である．また，合計容量としては 72TB であるが，可用性を高めるためにゾーンを 3 つとしているため，保存可能な実効容量は 24TB である．

図 3　システム全体の Object Server の構成

各ゾーンにおける，無作為に選んだ 5 つのオブジェクトの作成と削除に要した時間（単位は秒）を表 1 に示す．Swift のログに記録される PUT と DELETE の処理に要した時間であり，ゾーン 1 は swift 2(CPU：Xeon E3-1220 V2 3.10GHz, メモリ：4GB, NIC：1Gbps)，ゾーン 2 は swift 4 (CPU：Xeon E3-1240 V2 3.40GHz, メモリ：8GB, NIC：10Gbps)，ゾーン 3 は特定の使用済みパソコン 1 台での結果である．表 1 より，作成と削除の書き込みの性能については，サーバと比べて使用済みパソコンでも遜色ないことが分かる．1 台につき複数の大容量ディスクを搭載しているサーバに対して，使用済みパソコンは 1 台につき 1 つの小容量ディスクのみを搭載しているという構成の違いなどが要因として考えられる．

表 1　オブジェクトの作成と削除に要した時間

	ゾーン 1		ゾーン 2		ゾーン 3	
	作成	削除	作成	削除	作成	削除
1	0.0701	0.0069	0.0792	0.1135	0.0108	0.0168
2	1.6096	0.0064	1.6600	0.0058	1.6299	0.0984
3	0.0622	0.1262	0.0932	0.2223	0.0175	0.0184
4	0.1739	0.0068	0.2371	0.0601	0.1199	0.0073
5	0.0478	0.0066	0.4143	0.0067	0.0110	0.0064
平均	0.3927	0.1529	0.4968	0.0817	0.3587	0.0295

今回は，使用済みパソコン以外に必要であったスイッチングハブ，LAN ケーブル，電源タップも既存のものを利用したため，導入費用は 0 円であった．仮にこれらを新規で購入したとしても数万円程度である．また，既存のディスクを換装する場合は更に費用が発生するが，4TB の 3.5 インチ HDD が 1 万円程度で購入可能であり，大容量が必要な場合でもストレージ製品やサーバに比べて安価に導入可能だと考えられる．

4. おわりに

使用済みパソコンを分散ストレージとして再利用するにあたり，自動構築システムを実装し，実環境への組み込みおよび運用により性能や導入費用の評価を行った．今後の課題として，読み込みに要する時間といった更なる性能の評価や，電力消費量や保守費などのランニングコストも算出した上での，ストレージ製品やサーバに対する比較や有効性の検証が挙げられる．

謝辞

本研究は JSPS 科研費 16K00477 の助成を受けている．

参考文献

[1] CHAPTER 26. KICKSTART INSTALLATIONS, https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/installation_guide/chap-kickstart-installations.
[2] Welcome to Swift's documentation!, https://docs.openstack.org/developer/swift/.
[3] Ansible is Simple IT Automation, https://www.ansible.com/.