目次

1　はじめに

近年，マルチテナント型のクラウドにコンテナ技術が利用されている．マルチテナント型のクラウドの例として， Amazon Web Services や Google Cloud Platform がある．こういったクラウドを管理する場合に，isolation が重要である．ここで，isolation と throttle との違いを明確にしておく．ディスクへの書き込みを 15% に制限しているが， 30% 必要としているプロセスと， 85% に制限しているが 50% しか必要としていないプロセスを実行したとする．throttle であれば，前者はディスクへの書き込みを 15% ，後者は 50% の割合で使う．一方， isolation では，余っている分を自由に利用できるので，後者が 50% しか利用していないため，前者は必要としている 30% 分を使うことができる．

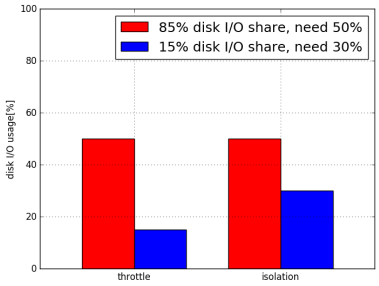

図 1.1　throttle と isolation との違い

コンテナ技術の一つとして， Linux コンテナ (LXC) がある．LXC は linux カーネル機能の一つである cgroup を利用して，コンテナ毎の資源管理を行っている．また，それによってコンテナ間の isolation を保っている．しかし， cgroup による isolation は不完全である． 2 個のコンテナを用意する．片方のコンテナでは cgroup によって disk I/O 使用率を 85% に制限し，Flexible IO(FIO) write benchmark を実行する．もう一方のコンテナでは，disk I/O 使用率を 15% に制限し，FIO write benchmark を 100 命令に一回 fsync しながら実行する．この状況では，disk I/O 使用率を 85% に制限している方は，本来は 85% 使えるはずが 40% しか使えなかった．

コンテナ環境における isolation の不完全性は， DDoS 攻撃への脆弱性となる可能性がある．本論文では，実際にコンテナ環境において利用される， MySQL と varmail に対して攻撃が可能であることを示し，その解析を行う．MySQL への攻撃が可能であるか検証するためコンテナを 2 つ用意する．片方のコンテナで， disk I/O 使用率を 85% に制限して MySQL benchmark を実行する．もう一方のコンテナで， disk I/O 使用率を 15% に制限して，ファイルのメタデータを頻繁に更新するスクリプトと FIO read benchmark を実行する． この時， MySQL は，本来 disk I/O を 85% 使えるはずが，50% しか使えていなかった．よって， MySQL に対して攻撃が可能だと言える．同様に， varmail への攻撃が可能であるか検証するためにコンテナを 2 つ用意する．片方のコンテナで disk I/O 使用率を 85% に制限して varmail benchmark を実行する．もう一方のコンテナで， disk I/O 使用率を 15% に制限して，ファイルのメタデータを頻繁に更新するスクリプトと FIO read benchmark を実行する．この時， varmail は，本来 disk I/O を 85% 使えるはずが， 50% しか使えていなかった．よって， varmail に対して攻撃が可能だと言える．

cgroup による isolation が不完全な原因は，journal であった．LXC ではコンテナ間で file system を共有し

ているため，複数のコンテナの disk I/O が，一つの journal でシリアライズされる．攻撃側は，頻繁に fsync を呼び，メタデータを更新しようとしている．攻撃側の頻繁な更新リクエストにより， journal に負荷がかかる．これによって， disk I/O の遅延時間が増加するようになるが，一つの journal で管理していることによって，すべてのコンテナでこの影響を受ける．したがって，攻撃対象の遅延時間を増やし，パフォーマンスを下げることが出来たのだと考える．また，攻撃側の，メタデータの更新リクエストの頻度を上げるほど，攻撃対象の遅延時間は増加する傾向にあった．

本論文の構成を以下に示す． 第 2 章では，LXC の資源管理の仕組みについて説明する． 第 3 章では，journaling file system の仕組みについて説明する． 第 4 章では，現在のコンテナ環境において， DDoS 攻撃が可能であることを示す． 第**??**章では， DDoS 攻撃が可能である原因を解析する． 第**??**章では，本研究に関連する研究について紹介する． 第**??**章では，まとめと今後の課題について述べる．

2 Linux コンテナにおける資源管理

近年，仮想化技術としてコンテナが注目されている．コンテナを実現する技術の一つに Linux コンテナ (LXC) がある．本章では， LXC の資源管理の仕組みについて説明をする．

LXC は，一つのマシン上にコンテナという隔離空間を作り出す．LXC は， Linux カーネル機能の一つである cgroup を使い，コンテナへの資源の割り当てを管理している．

cgroup は， OS が管理する資源を一元的に管理できる Linux カーネル機能である．cgroup は，プロセス，ファイルシステム， CPU ，メモリ， block I/O デバイスなどの各種デバイスといった多くのものを管理できる． cgroup はサブシステムによって，これら資源を管理している． cgroup は資源のグループ化を行い，グループごとに資源利用の優先度を決めたり，利用できる資源を制限したりしている．さらに，グループを隔離することで，他のグループから中が見えないようにしている．

表 2.1 cgroup のサブシステムとその機能

サブシステム名	機能
blkio	ブロックデバイスへの入出力アクセスの制限を設定する
cpu	CPU コアの時間配分の割合を設定する
cpuacct	タスクが消費する CPU 時間をレポートする
cpuset	使用可能な CPU コア数を設定する
devices	デバイスへのアクセスを制御する
freezer	タスクの一時停止と再開を制御する
hugetlb	cgroup からの hugetlb の使用
memory	タスクによって使用されるメモリの制限を設定する
net_cls	プロセスが発信するパケットに識別子を付与し，制御する
net_prio	タスクのネットワークの優先度を動的に設定する
pids	起動するプロセス数を制限する

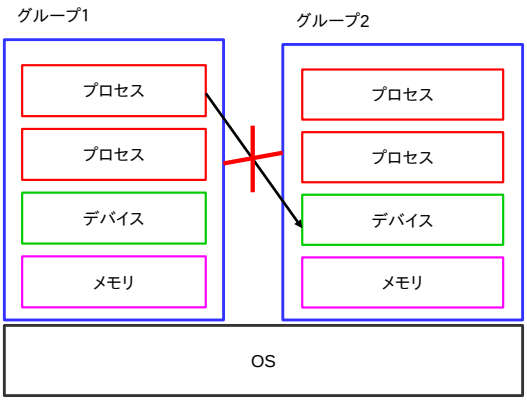

図 2.1　cgroup によるグループ化

3 journaling file system

3.1 journaling の意味

3.2 journaling の手順

4　Linux コンテナに対する DDoS 攻撃

5　コンテナにおける資源管理の脆弱性解析

6　関連研究

7　まとめ