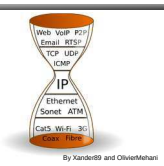
Osaka University

インターネットプロトコルスタックの進化過程の評価を目的としたLinuxカーネルの分析

大阪大学 基礎工学部 情報科学科
村田研究室 宮川裕考

研究背景

- **インターネットのプロトコルスタック**
  - IPを中心とするモノリシックな機能結合により構成
- **インターネット利用形態の多様化**
  - スマートフォンの普及、サービスの多様化
  - 柔軟かつ迅速なネットワーク構築への社会的要求
- **ネットワーク仮想化技術への期待と課題**
  - ネットワークの機能をコンポーネントとして抽出し、仮想化技術を用いてそれらの機能を組み合わせて実行する NFV (Network Function Virtualization) の標準化が進められている
  - コンポーネント化の労力が未知
    - 仮想化したいネットワーク機能が他のプロトコルや他のネットワーク機能にどの程度依存しているか
    - ネットワーク機能の独立性がどの程度担保されているか



By Xander89 and OlivierMehani

Linuxカーネルのプロトコル実装を題材とし、プロトコル間／プロトコル内のネットワーク機能がどのように結合し変化しているのかを明らかにする

研究のアプローチ

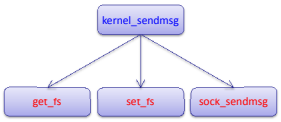
- **アプローチ**
  - 多数の通信プロトコルを実装した Linux カーネルを対象とし、関数の呼び出し関係を分析
  - 開発進行にともなう通信関連の関数呼び出し関係の変遷を分析
- **研究の位置づけ**
  - Linux カーネル全体の関数呼び出し関係を調べた研究 [4]
    - ネットワーク分析手法を適用し、いくつかの統計的指標の推移を調査
    - ソフトウェア品質管理の観点からの機能故障に対する堅牢性を調べている
  - 本報告では、 Linux カーネルをネットワーク機能間の利用関係に着目して分析する

[4] Y. Gao, Z. Zheng, and F. Qin, "Analysis of linux kernel as a complex network," Chaos, Solitons & Fractals, vol. 69, pp. 246 - 252, Nov. 2014.

Linuxカーネルの分析

- **分析対象**
  - Linux カーネル 3.0 (2011) – 3.16 (2014)
- **分析手順**
  - Linux カーネルからコールグラフを生成
    - コールグラフ：関数をノード、関数呼び出しをリンクとした有向グラフ
    - 通信に関連した関数群が保持されるディレクトリ 'net' を対象として作成
  - コールグラフを構成するノードをネットワーク機能にもとづき分類
    - ネットワーク機能は複数の関数により構成
  - ネットワーク機能間の結びつきをバージョンごとに分析

```
int kernel_sendmsg(struct socket *sock, struct
msghdr *msg, struct kvec *vec, size_t num,
size_t size) {
        mm_segment_t oldfs = get_fs();
        int result;
        set_fs(KERNEL_DS);
        // some codes
        result = sock_sendmsg(sock, msg, size);
        set_fs(oldfs);
        return result;}
```

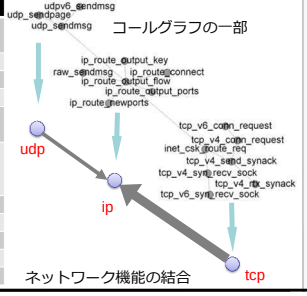


ネットワーク機能にもとづくノードの分類

コールグラフに含まれるノードをそれが担うネットワーク機能により分類する

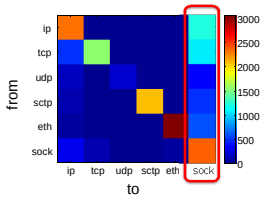
- ネットワーク機能は関数名に含まれる特定文字列にもとづき分類

グループ	対応する正規表現
ip	ip(¥verbi¥!w*(4$\|6))? inet(4$|6)?
tcp	tcp(v?(4$|6))?
udp	udp(4$\|5v?6$|5lite)?
sctp	sctp(v6|probe)?
socket	sock skb?
ethernet	.*80211 arp eth(er$|tool)?
icmp	icmp(v(4$|6))?
netfilter	nf
nlmsg	(ge)?nlmsg
router	rt(6$\|$nl$\|$nh$\|$netlink$\|$msg$\|$m)?
xfrm	xfrm

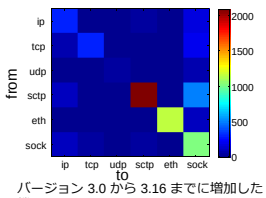


ネットワーク機能の接続関係

- **1 つのバージョンにおける接続関係（左図）**
  - ethernet ・IP 内へのリンクが多い
  - socket へのリンクが多い
- **バージョン進行による接続関係の変化（右図）**
  - 開発進行により新しいプロトコル (SCTP) へのリンクが急増
    - SCTP: 比較的新しいトランスポート層プロトコル
  - 機能内のリンク（対角線上）の増加が顕著



バージョン 3.16 における機能間のリンク数

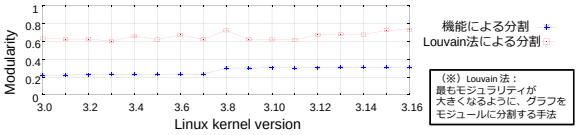


バージョン 3.0 から 3.16 までに増加した機能間のリンク数

ネットワーク機能の独立性の分析

- **ネットワーク機能の独立性を評価**
  - 評価指標：モジュラリティ
    - コールグラフをいくつかのノード集合 (モジュール) に分割
    - 分割 P に対するモジュラリティ$Q(P)$は以下の式により定義される
      - $Q(P) = \frac{1}{2m}\sum_{i=1}^{n}\sum_{j=1}^{n}\left[A_{i,j} - \frac{k_i k_j}{2m}\right]\delta(C_i, C_j)$
  - $Q(P)$が大きいと**機能内**のつながりが強く、小さいと**機能間**のつながりが強い
- **ネットワーク機能に基づく分割Pを与えた場合のモジュラリティは小さく、ネットワーク機能が独立していないことが分かる**

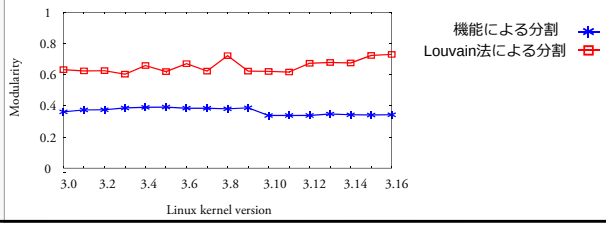
分割 Pにおいて、ノードi とノードj が同じモジュールに属する時1, そうでない時0となる２値変数



（※）Louvain 法：
最もモジュラリティが大きくなるように、グラフをモジュールに分割する手法

IP内のネットワーク機能の独立性

- **プロトコル内におけるネットワーク機能の独立性を評価**
  - 評価指標: モジュラリティ
- **ネットワーク機能に基づく分割Pを与えた場合のモジュラリティは小さく、プロトコル内においてもネットワーク機能は独立していない**



まとめと今後の課題

- **まとめ**
  - ネットワーク機能仮想化による柔軟なネットワーク構築
  - Linux カーネルのコールグラフを用いて、ネットワーク機能間の関係を分析した
  - プロトコル間とプロトコル内の双方においてネットワーク機能の独立性が損なわれていることを明らかにした
    - →ネットワーク機能をコンポーネント化するにあたりコストが増大する
- **今後の課題**
  - ネットワーク機能の使用頻度にもとづいた分析
  - インターネットプロトコルの進化過程の分析
    - ネットワーク仮想化技術を効果的に導入するための方策の提言
    - プロトコルスタック構成法の提案