eBPF とフィルタドライバを利用した OS レベル仮想パッチによる脆弱性ランタイム保護

島田 凌 [1,2]　池上 祐太 [1,a)]　濱村 将人 [1]

概要：ソフトウェアの脆弱性は，サーバやクライアントシステムに加え，ルーター，VPN 機器，FW．WAF，UTM，IPS/IDS といったネットワーク機器やセキュリティ対策機器にも深刻な脅威である．特に，VPN 機能を持つ機器はインターネットに公開され，認証情報や認証サーバへのアクセス経路を持つため，攻撃者の主要な標的となっている．これらの機器の前段にセキュリティ対策機器を導入する対策があるが，VPN 通信への悪影響や導入機器自身が新たな攻撃対象のリスクとなることから，有効な対策とはなっていない．このため，ネットワーク機器やセキュリティ対策機器に対する脆弱性対策は，サーバやクライアントシステムと並んで課題となっている．本研究では，Linux と Windows の双方に対して，eBPF やフィルタドライバの仕組みを利用し，OS レベルで仮想パッチを適用する手法を提案する．eBPF やフィルタドライバを利用することで，システムを停止・再起動させることなく，仮想パッチを適用できる．本稿では，シグネチャ形式でのカオスパッチにて実現しているが，eBPF やフィルタドライバではシグネチャ以外の OS の振舞いも監視できるため，より高度な防御機構も実現できる．さらに，エンドポイントで動作する仕組みであるため，サーバやクライアントのシステムだけでなく，ネットワーク機器にも適用でき，ゼロデイ脆弱性や N デイ脆弱性への応急対応に有効である． 評価では，Linux や Windows のサーバやクライアントシステムに提案手法を適用し，脆弱性を狙う悪性通信を検知・遮断することができた．また，OpenVPN や SoftEther VPN を動作させた VPN サーバに対しても提案手法を適用し，VPN 通信に影響させることなく，悪性通信を検知・遮断することができた．現在は，暗号通信に関連する API をフックすることで，暗号通信にも対応できるよう検討している．

キーワード：eBPF，WFP，フィルタドライバ，脆弱性，仮想パッチ，VPN，IPS

Leveraging eBPF and Filter Driver for OS-Level Virtual Patching in Runtime Vulnerability Protection

Ryo Shimada[1,2]　Yuta Ikegami[1,a)]　Masato Hamamura[1]

Abstract: Software vulnerabilities pose a serious threat not only to servers and client systems but also to network and security devices such as routers, VPN appliances, firewalls (FW), web application firewalls (WAF), unified threat management (UTM) systems, and intrusion prevention/detection systems (IPS/IDS). In particular, devices equipped with VPN functionality are frequently exposed to the Internet and maintain access paths to authentication credentials and authentication servers, thereby becoming primary targets for attackers. Although one possible countermeasure is to deploy additional security appliances in front of these devices, such an approach has proven ineffective due to potential adverse effects on VPN traffic as well as the risk that the newly introduced devices themselves may become targets of attack. Consequently, addressing vulnerabilities in network and security appliances has emerged as a critical issue on par with securing servers and client systems. In this study, we propose a method for applying virtual patches at the operating system (OS) level to both Linux and Windows platforms by leveraging the mechanisms of eBPF and filter drivers. By utilizing eBPF and filter drivers, virtual patches can be deployed without requiring system shutdown or reboot. In this paper, we demonstrate our approach through a signature-based “chaos patching” mechanism; however, because eBPF and filter drivers can monitor not only signatures but also broader OS behavior, more sophisticated defensive mechanisms can be realized. Furthermore, since the proposed method operates at the endpoint level, it can be applied not only to server and client systems but also to network appliances, providing an effective means of emergency response against zero-day and n-day vulnerabilities. In our evaluation, the proposed method was applied to Linux and Windows servers and client systems, where it successfully detected and blocked malicious communications exploiting software vulnerabilities. The method was further applied to VPN servers running OpenVPN and SoftEther VPN, where it was able to detect and block malicious traffic without adversely affecting VPN communications. Currently, we are investigating an extension of this approach by hooking APIs related to cryptographic communication, thereby enabling support for encrypted traffic.

Keywords: eBPF, WFP, Filter Driver, Vulnerability, Virtual Patching, VPN, IPS

1. はじめに

サイバー攻撃において，リモートから初期侵入や情報窃取を実施する際に，ソフトウェアの脆弱性を悪用するケースは多い．リモートから悪用できる脆弱性は，サーバやクライアントシステムに加え，ルーター，VPN 機器，FW，WAF，UTM，IPS/IDS といったネットワーク機器やセキュリティ対策機器に対しても深刻な脅威である．特に VPN 機

1 Powder Keg Technologies 株式会社
Powder Keg Technologies, Inc.
2 筑波大学大学院
Graduate School of University of Tsukuba
a) yuta.ikegami@powderkegtech.com

能を持つ機器はインターネットに公開され，認証情報の保持や認証サーバへのアクセス経路を持つため，攻撃者の主要な標的となっている．これらの対策としては，脆弱性パッチを適用することが最も有効な対策であるが，緊急性の高いゼロデイ脆弱性や N デイ脆弱性については，パッチのリリースから適用されるまでの期間は，何らかの対策をしなければならない．この場合においては，多くの組織が対象機器を停止させるといった対策を取るインシデントハンドリングのルールになっているケースが多くあり，パッチが適用されるまでシステムを利用できなくなる等の業務影響も非常に大きくなる．システム停止の課題に対応するために，これらの機器の前段にセキュリティ対策機器を導入する方法があるが，VPN 通信への悪影響や導入機器自身が新たな攻撃対象のリスクとなることから，有効な対策とはなっていない．このため，ネットワーク機器やセキュリティ対策機器に対する脆弱性対策は課題となっている．

本研究では，Linux と Windows の双方に対して，eBPF やフィルタドライバの仕組みを利用し，OS レベルで最小限の仮想パッチを適用する手法を提案する．具体的には，eBPF やフィルタドライバを利用し，脆弱性が悪用されるソフトウェアに通信が到達する前に，NIC ドライバや TCP/IP スタックの処理時点で通信を検査・遮断する．eBPF やフィルタドライバを利用することで，システムを停止・再起動させることなく，仮想パッチを適用できる．提案手法はエンドポイントに導入する仕組みであるため，サーバやクライアントのシステムだけでなく，ネットワーク機器にも適用できる可能性があり，ゼロデイ脆弱性や N デイ脆弱性への応急対応に有効である．本稿で実現した仮想パッチはシグネチャ形式であるが，eBPF やフィルタドライバでは OS の振舞いも監視できるため，より高度な防御機構も実現できる．評価では，Linux や Windows のサーバやクライアントシステムに提案手法を適用し，リモートから脆弱性を悪用する通信を検知・遮断することができた．また，OpenVPN や SoftEther VPN を動作させた VPN システムに対して提案手法を適用し，VPN 通信に影響させることなく，悪性通信を検知・遮断することができた．

本研究の貢献は以下の２点である．

- eBPF とフィルタドライバを用いて，Linux と Windows の双方における仮想パッチによる脆弱性ランタイム保護技術を提案した．
- VPN サーバ自身に提案手法を導入し，VPN 通信を阻害することなく，対象とする脆弱性を悪用する通信を検知・遮断できることを実証した．

2. リモートからの脆弱性攻撃と課題

サイバー攻撃における初期侵入の主要な経路として，リモートからの脆弱性攻撃が広く利用されている．公開系システムや VPN 機器はインターネットに直接接続されているため攻撃対象となりやすく，特にパッチ未適用の状態の機器は攻撃者も定常的に探索しており，攻撃対象となる可能性が高い[1]．警察庁の調査によれば，ランサムウェア攻撃における侵入経路の約 47%が VPN 機器を経由しており，その大半が既知の脆弱性を突いたものである[2]．近年では，Fortinet，Ivanti，および Palo Alto Networks 製品に存在したゼロデイ脆弱性を狙った大規模攻撃が報告され，VPN が「企業ネットワークへの玄関口」として利用されている実態が明らかになった[3]．これらの脆弱性は，IT ネットワーク側の公開系システムや VPN 機器が侵入起点となり，制御システム（OT）への横展開に利用されるケースも確認されている[4]．これらの実態より，公開系システムや VPN 機器は攻撃者にとって最も効率的な初期侵入手段であることを示しており，リモートから悪用できる脆弱性への早急な対策が急務となっている．

我々が独自に調査した主要な VPN 機器とそのベース OS を表 1 に示す．本調査より，多くの機器が Linux や FreeBSD といった汎用 OS を基盤として採用していることが分かった．これらの機器は通常，独自 CLI や GUI を通じて操作する仕様となっており，利用者が直接 Linux や FreeBSD のコマンドを実行することはできない．しかし，基盤となる汎用 OS に深刻な脆弱性が発見された場合，それらの脆弱性の影響がこれらの機器にも発生する可能性がある[5,6,7]．

表 1　主要な VPN 機器とベース OS

Table 1　Major VPN devices and base OS.

メーカ	機種	ベース OS
Cisco	・ ASA ・ FirePower	Linux
Palo Alto Networks	・ PA ・ GlobalProtect VPN	Linux
Fortinet	・ FortiGate	Linux
Check Point	・ Quantum Security Gateway ・ Remote Access VPN	Linux
Juniper	・ SRX ・ MAG	FreeBSD/Linux
SonicWall	・ NSA ・ TZ	Linux
Barrracuda	・ CloudGen Firewall	Linux
Ivanti	・ Ivanti Connect Secure	Linux
YAMAHA	・ RTX	独自 OS
NEC	・ IX	Linux
アライドテレシス	・ CentreCOM AR	Linux
F5	・ BIG-IP APM	Linux
Zyxel	・ ZyWALL	独自 OS
WatchGuard	・ Firebox	Linux
Array Networks	・ AG	FreeBSD
OpenVPN	・ OpenVPN	FreeBSD/Linux/Windows
SoftEther	・ SoftEther VPN ・ Packetix VPN	FreeBSD/Linux/Windows

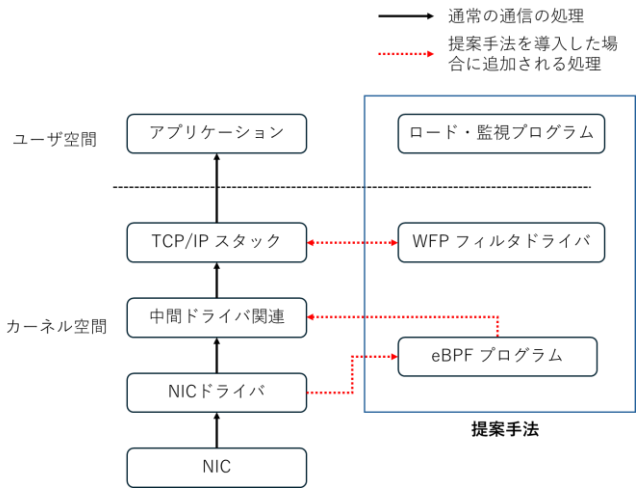

図 1　提案手法の概要図

Figure 1　Overview diagram of proposed method.

したがって，ネットワーク機器のセキュリティ確保には，製品固有の脆弱性対策に加え，ベース OS 由来の脆弱性への迅速な対応が不可欠である．

これらの脆弱性への対応として，最も効果的な手法は脆弱性パッチを機器に適用することである．しかし，ゼロデイ脆弱性やNデイ脆弱性として悪用されるケースが多いため，脆弱性が公表された時点で，パッチのリリースがない場合は，対象機器を停止またはネットワークから遮断するといった対応を取る組織が多い．これにより，パッチのリリースから適用されるまでの期間は，システム停止等が発生するため，業務影響が大きい．このため，パッチのリリースから適用される期間の間においても，脆弱性を保護する対策が課題である．

3. 提案手法

3.1 提案手法の概要

図 1 に提案手法の概要図を示す．提案手法は，Linux と Windows の双方の OS に対応できる仕組みとなっており，Linux では eBPF，Windows では WFP フィルタドライバを用いる．eBPF と WFP フィルタドライバは，それぞれカーネル空間で動作するプログラムであるため，それらを起動･監視するためのロード･監視プログラムも必要となる．eBPF と WFP フィルタドライバには，対象となる脆弱性を検出するためのシグネチャを保持しており，インバウンド通信をすべて補足し，シグネチャとマッチするか検査する．シグネチャとマッチした通信は提案システムによりドロップされ，通信元の IP アドレスを一定期間ブロックする．この仕組みにより，メーカから正規のパッチが適用されるまでの間，提案手法により仮想パッチを適用することで，対象システムを脆弱性攻撃から保護することができる．また，提案手法はシステムの停止や再起動をすることなく導入できる．これにより，業務影響も最小限に抑制できる．

3.2 eBPF を利用した仮想パッチ

eBPF（extended Berkeley Packet Filter）とは，Linux カーネルに対して安全かつ柔軟に機能拡張を可能にする技術である．eBPF の登場以前においては，カーネルモジュールでの適用が Linux カーネルへの機能追加の主流であったが，その開発にはカーネル内部の詳細な知識を要し，さらに実装上の不具合がシステム全体のクラッシュにつながる危険性があった．これに対し，eBPF は事前に「eBPF 検証器」による安全性チェックを行うため，ユーザ空間で作成したプログラムを安全にカーネル空間にロードでき，信頼性を維持したまま機能を拡張できる．eBPF は，Linux Kernel 3.18 以降で導入され，ネットワークのロードバランサやセキュリティ対策機能等，幅広い用途に応用されている．

提案手法では，eBPF の機能の一つである XDP（eXpress Data Path）を利用し，NIC ドライバでの処理直後に通信処理をフックする機能を用いる．XDP を利用することで，NIC ドライバ以降に通信を渡す前に，悪性通信であるか否かを判別することができるため，アプリケーションを代表とするユーザ空間はもちろん，TCP/IP スタックのようなカーネル処理での脆弱性も保護することができる．

提案手法は，事前に脆弱性のシグネチャを作成しておき，ロード・監視プログラムの起動時に，BPF マップにシグネチャを展開する．BPF マップとは，eBPF プログラムとユーザ側プログラムの共有メモリであり，双方向でデータのやり取りが可能となる．eBPF プログラムは，NIC ドライバで処理された通信パケットのヘッダとペイロードを解析し，ペイロード部分にシグネチャと一致するデータが存在するか確認する．ペイロードにシグネチャと一致するデータ存在する場合，perf_event により，ロード･監視プログラムに通知し，悪性通信の検出を通知する．また，悪性通信の送信元 IP アドレスを一定時間ブロックする機能も実装しており（現在の実装では 10 分間ブロックする），同じ送信元からの別の脆弱性を悪用する攻撃が実施されたとしても，防御することが可能となる．

3.3 Aho-Corasick 法

eBPF で作成されたプログラムは，カーネル空間で安全に実行されることを目的として設計されているため，無限ループや過度に複雑な制御構造を禁止する厳格な検証機構が実装されている（eBPF 検証機）．このため，提案手法で用いるシグネチャ探索のような文字列探索やパターン照合のように繰り返し処理を伴うアルゴリズムを実装する際には，ループ回数に上限を設けるか，定数回に制限する必要がある．そこで，提案手法ではこの制約配下でも効率的にシグネチャ探索を実施するため，Aho-Corasick 法という文字列探索アルゴリズムを用いている．

Aho-Corasick 法は複数の文字列パターンを同時に処理できる有限状態オートマトン（Trie 木）を構築し，入力テキストを一度走査するだけで複数の文字列を同時に照合できるというアルゴリズムであり，効率的にシグネチャ探索が可能となる．提案手法では，ロード・監視プログラムの起動時に，Aho-Corasick 法をシグネチャの文字列に適用し，Trie 木構造を作成する．この Trie 木構造を BPF マップに保存し，eBPF プログラムからも参照できるようにする．これにより，eBPF の制約が存在する eBPF プログラムにおいても，高速かつ効率的にシグネチャ探索することができる．現在の実装では最大 1,150 バイトのデータ長まで eBPF プログラムで確認することができ，ほとんどの脆弱性を悪用する通信の特徴はカバーできると考える．

3.4 WFP フィルタドライバを利用した仮想パッチ

Windows については，WFP フィルタドライバにより，脆弱性を悪用する通信を検出・ブロックする．WFP は，Windows Vista 以降に導入されたパケット/ストリームフィルタリング API である．TCP/IP スタックの各段階（IP 層/TCP/UDP 層 / ソケット層 / アプリ層） にフック処理を挿入することができ，WFP フィルタドライバで通信を捕捉・処理することができる．いくつかの EPP（Endpoint Protection Platform）や EDR（Endpoint Detection and Response）製品において，マルウェアの通信先を検知・フィルタする目的で WFP フィルタドライバが利用されている．

提案手法では，TCP/IP スタックの TCP/UDP 層で処理をフックし，通信パケットのペイロード情報を解析し，脆弱性を悪用する通信か否かを判別している．WFP では eBPF のようにループの制限等は存在しないため，Aho-Corasick 法は利用せず，RtlCompareMemory 関数でシグネチャを探索している．提案手法は，商用の IDS/IPS のようにあらゆる悪性通信を保護するといった目的ではなく，対象システムの特定の脆弱性のみを保護するといった目的であるため，シグネチャの数も小さいことから，RtlCompareMemory 関数でも問題ないと考える．シグネチャの数を数十以上に増やす場合は，WFP フィルタドライバにおいても，Aho-Corasick 法の実装は必要になると考える．

4. 評価

4.1 評価概要と実験環境

本評価では，提案手法により脆弱性を悪用する通信が正しく検知・ブロックされるか，VPN サーバ上でも VPN 通信を阻害することなく動作できるか，提案手法の導入によるオーバヘッドの測定の 3 つの評価を実施した．

提案手法を導入するサーバ環境は，VMware ESXi 上に構築した仮想マシンで実施し，公開系サーバとしてインターネット経由で VPN 接続等ができるように設定し，評価用クライアント環境のグローバル IP アドレスのみ受け付け

```
/* 1) CVE-2017-0143 (MS17-010, EternalBlue) */
static const unsigned char P_CVE_2017_0143_1[] = {
    0x32, 0x00, 0x00, 0x00,
    0x00, 0x18, 0x07, 0xc0
};
static const unsigned char P_CVE_2017_0143_2[] = {
    0x33, 0x00, 0x00, 0x00,
    0x00, 0x18, 0x07, 0xc0
};

/* 2) CVE-2021-44228 (Log4Shell) */
static const unsigned char P_CVE_2021_44228_1[] =
    "%24%7bjndi%3aldap";

static const unsigned char P_CVE_2021_44228_2[] =
    "%24%7B%24%7B%3A%3A-j%7D%24%7B%3A%3A-n%7D%24%7B%3A%3A-d%7D"
    "%24%7B%3A%3A-i%7D%3A%24%7B%3A%3A-1%7D%24%7B%3A%3A-d%7D"
    "%24%7B%3A%3A-a%7D%24%7B%3A%3A-p";

static const unsigned char P_CVE_2021_44228_3[] =
    "${jndi:ldaps:";

/* 3) CVE-2017-12615 (Tomcat) */
static const unsigned char P_CVE_2017_12615[] =
    "PUT /test2.jsp/ HTTP/1.1\r\n";

/* 4) CVE-2022-22947 (Spring) */
static const unsigned char P_CVE_2022_22947[] =
    "{new String(T(org.springframework.util.StreamUtils).copyToByteArray("
    "T(java.lang.Runtime).getRuntime().exec(new String[]{\\\"id\\\"})"
    ".getInputStream()))}";
```

図 2　作成したシグネチャ
Figure 2　Created signature.

るようネットワーク設定をした上で評価した．サーバ環境の Linux については，Ubuntu Server 24.04（Linux Kernel 6.8.0-64-generic），4vCPU，メモリ 8GB であり，Windows については，Windows 11 Pro（23H2） 8vCPU，メモリ 16GB で実施した．攻撃評価に用いるクライアント環境は，Kali Linux 2025.01（Linux Kernel 6.12.13-amd64），2vCPU，メモリ 8GB で実施した．

4.2 脆弱性の検出と防御

本評価で用いた脆弱性は，次の 4 種類である．

- CVE-2017-0143 (Windows SMB の脆弱性)
- CVE-2021-44228 (Log4j の脆弱性)
- CVE-2017-12615 (Tomcat の脆弱性)
- CVE-2022-22947 (Spring の脆弱性)

いずれの脆弱性においても，リモートから脆弱性を悪用することが可能な脆弱性である．それぞれの脆弱性に対応したシグネチャを図 2 に示す．提案手法では，これらのシグネチャを eBPF および WFP フィルタドライバのプログラムのヘッダファイルに記載し，プログラムロード時に読み込まれるようになっている．

脆弱性を悪用する攻撃検証は，クライアントマシンに対象の脆弱性通信をキャプチャした pcap ファイル[21]を利用し，scapy を利用して pcap の通信データを提案手法が導入されているサーバに対して送信することで検証した．

表 2　VPN 方式と利用ポート

Table 2　VPN method and ports used.

VPN ソフト	SSL-VPN	L2TP/IPsec
OpenVPN	・ 443/tcp	-
SoftEther VPN	・ 443/tcp	・ 500/udp ・ 4500/udp ・ 1701/udp

検証では，Linux と Windows 共に，対象の脆弱性を悪用するパケットを検知・遮断することができ，一定時間クライアントマシンからの疎通がブロックされることが確認できた．シグネチャに含まれない通信パケットについては，ブロックすることなく許可されていることも確認できた．

4.3 VPN 通信配下での動作

本評価では，表 2 のとおり，2 種類の VPN 方式および 2 種類の VPN ソフトを用い，それぞれの環境下で提案手法が問題なく動作するか確認した. VPN 方式は, SSL-VPN と L2TP/IPsec を用い，VPN 通信が確立された上で提案手法を導入し，通信が問題なく確立維持されているか確認した．VPN ソフトは，SSL-VPN の検証では，OpenVPN と SoftEther VPN を利用した．L2TP/IPsec の検証では，SoftEther VPN を利用した．

Linux および Windows 環境のそれぞれにおいて，SSL-VPN および L2TP/IPsec ともに VPN 通信を阻害することなく, 提案手法を導入することができた．また, 4.2 項で用いた脆弱性を悪用する通信を発生させても，問題なく対象通信を止めることができた．これより，提案手法は VPN 通信に悪影響を与えることなく，有効に対象システムに導入し，仮想パッチを施すことが確認できた．

5. 関連研究

5.1 eBPF を利用した悪性通信の防御

文献[8] は，eBPF を用いて Snort のシグネチャを eBPF のプログラムに適用し，eBPF にて IDS/IPS を構築するという研究内容であり，脆弱性を悪用する通信の検出も対象としている．しかし，検出機構やシグネチャはユーザ空間プログラムで実装しており, eBPF プログラムは簡易的に通信内容を確認するのみであり，メイン処理はユーザ空間となっている．このため, eBPF プログラム側にて悪性通信の判別は実施していない．また，本研究は，Snort のスループットの向上で eBPF を活用するということが主な目的であり，eBPF プログラムに適用した Snort のシグネチャが脆弱性を悪用した通信に対して，有効に検知・ブロックできるかといった評価が十分に実施されていない．

文献[9-17]は，eBPF を用いてポートスキャン，ブルートフォース攻撃, および DDoS 等の悪性通信を保護する研究内容である. eBPF にて通信トラフィックの測定や通信のヘッダを解析し，対象とする不正通信を防御している．DDoS 攻撃には eBPF プログラムをロードバランサとして別システムに通信を転送して通信をドロップすることで, DDoS 攻撃に対応している．多くの手法が DDoS 攻撃に特化しており, パケットのペイロード部分までの検査はしておらず, 脆弱性を悪用する通信の判別や防御までには至っていない．

5.2 WFP や NDIS フィルタドライバを利用した悪性通信の防御

文献[18-20] は，WFP や NDIS のフィルタドライバで，悪性通信を防御したり，通信ログをキャプチャしたりというものである．これらの研究は，マルウェアのアウトバウンド通信を特定して保護することや通信ログの保全を目的としており, 脆弱性を悪用する通信を検知する機能はない．このため，本稿のようなインバウンドの脆弱性を悪用する通信については対象外となっている．

6. 考察と課題

6.1 暗号通信への対応

提案手法は，NIC ドライバや TCP/IP スタック周辺での通信処理をフックすることで悪性通信を判別しているため, 暗号通信には対応できていない．脆弱性を悪用する通信においても暗号通信（https）の割合は増えてきており，暗号通信の中身もデコードし，ペイロードを確認する処理が必要となる．これを実現するには，暗号ライブラリ内のデコード に利用している処理をフックし, 通信内容を確認するといった方法が考えられる．

6.2 TCP/IP スタックの脆弱性への対応

TCP/IP スタックの脆弱性については，これまで複数の危険性の脆弱性が報告されている [22-29]. 提案手法の eBPF の方式においては，NIC ドライバ直後の処理をフックしているため，TCP/IP スタックの脆弱性にも対応することができる．しかし，TCP/IP スタックの脆弱性は，単純なシグネチャでは検知することが難しく，通信の処理フローやプロトコルヘッダも関連付けて確認する必要がある脆弱性も存在する．現在の提案手法は，パケット内のペイロード部分のみしかチェックしていないため，処理フローで発生する脆弱性やプロトコルヘッダも関連した脆弱性は検知できない可能性が高い．また，Windows においての WFP フィルタドライバは，TCP/UDP 層の処理後にフックしているため，IP 層や TCP/UDP 層の処理で発生する TCP/IP スタックの脆弱性には対応できない．これに対応するには，データリンク層で処理をフックするか，NDIS フィルタドライバで対応する必要がある．

6.3 レガシーOS や商用ネットワーク機器等への適用

提案手法は, eBPF を利用しているため, Linux Kernel 3.18 より前のレガシーOS には対応できない．レガシーOS に対応するには, カーネルモジュール等で同様の手法を構築する必要がある．Windows においても，Vista より前の

OS は WFP フィルタドライバは適用できないため，別の手法を用いる必要がある．

商用のネットワーク機器やセキュリティ機器等については，表 1 のとおり，Linux をベース OS にした機器が多いため，Linux Kernel 3.18 以降のバージョンを利用している場合は，eBPF の手法を適用できる可能性がある．しかし，ユーザは機器メーカが提供する独自 CLI や GUI の範囲内でのみしか操作できないため，ユーザ側が提案手法を適用できる可能性は低い．このため，機器を提供するメーカ側にて，提案手法を適用する仕組みの整備やメーカ側の対策として提案手法を活用することが考えられる（正式パッチのリリースおよび適用までの期間の応急処理としての仮想パッチの役割で提案手法にて保護する等）．

6.4 シグネチャの作成

提案手法は，事前に脆弱性のシグネチャを作成する必要があるため，シグネチャの作成の元となるエクスプロイトコードや悪性通信のキャプチャファイルを入手する必要がある．既知の脆弱性は，比較的入手しやすいが，ゼロデイ脆弱性や N デイ脆弱性は，メーカや一部のセキュリティベンダのみしかシグネチャの元となるデータを所有していない可能性が高く，それらのデータが提供されない場合は，提案手法を活用することは難しい．この課題に対応するには，公開された脆弱性情報のみからシグネチャを自動生成する技術やシグネチャ以外の検知方法を組み込む必要性がある．

6.5 オーバヘッドやリソース使用量の測定

本稿では，提案手法の導入によるオーバヘッドやリソース使用量の測定は実施できていない．関連研究では，eBPF を用いることで，既存の IDS/IPS よりも高速に通信を処理することができ，スループットが向上したという評価が多くあるため，我々の提案手法でも同様の効果が期待できる．

7. おわりに

本研究では，eBPF と WFP フィルタドライバを用いた OS レベルでの仮想パッチ手法を提案した．提案手法は，対象システムを停止・再起動させることなく導入できるため，業務影響を最小限に留めることができる．仮想パッチについても，既存の IDS/IPS のようにあらゆるシグネチャを照合するのではなく，対象システムに影響のある脆弱性のみを検知・防御できるシグネチャを仮想パッチにて適用するため，システム負荷も最小限に抑制できると考える．

提案手法は，Linux と Windows の双方で導入できるよう実現しており，既知の脆弱性を悪用する通信について，システムに影響を与えることなく脆弱性通信を検出できた．また，VPN サーバに対しても，VPN 通信を阻害することなく導入できた．今後は，6 章の「考察と課題」に記載した内容について取組み，さらなる機能向上と精度向上を図る．

謝辞 本発表は，ＮＥＤＯ（国立研究開発法人新エネルギー・産業技術総合開発機構）の委託事業「経済安全保障重要技術育成プログラム／先進的サイバー防御機能・分析能力強化」（JPNP24003）によるものである．

参考文献

[1] C. Deepika, K. Abirami..Systematic Literature Review on VPN Security: Adaptive Multi-Tunnelling as a Mitigation Strategy. Journal of Information Systems Engineering and Management, 2025, Vol. 10 No. 39s

[2] Trend Micro. ランサムウェアの侵入経路の約半数は VPN 機器から．トレンドマイクロ セキュリティブログ, 2024-11-01, https://www.trendmicro.com/ja_jp/jp-security/24/k/expertview-20241101-01.html.

[3] LAC. SSL-VPN 機器を狙う攻撃の動向. LAC WATCH, 2024-08-20, https://www.lac.co.jp/lacwatch/report/20240820_004076.html.

[4] Dragos. Why adversaries target VPN appliances: The pathway from IT to OT cyber attack. Dragos Blog, 2024-09-04, https://www.dragos.com/blog/why-adversaries-target-vpn-appliances-the-pathway-from-it-to-ot-cyber-attack/.

[5] Cisco. (2022). *Cisco Bug Search Tool – Linux Kernel Vulnerability (CSCwm85644)*. Retrieved from https://bst.cisco.com/quickview/bug/CSCwm85644

[6] The Hacker News. (2019, December 10). *New unpatched vulnerability affects VPNs on Linux, Android, macOS, and iOS*. Retrieved from https://thehackernews.com/2019/12/linux-vpn-hacking.html

[7] Mixon-Baca, B., et al. (2024). *Vulnerabilities in VPNs*. In *Proceedings on Privacy Enhancing Technologies (PETS 2024)*. Retrieved from https://citizenlab.ca/2024/07/vulnerabilities-in-vpns-paper-presented-at-the-privacy-enhancing-technologies-symposium-2024/

[8] Wang, S.-Y., & Chang, J.-C. (2022). Design and implementation of an intrusion detection system by using extended BPF in the Linux kernel. Journal of Network and Computer Applications, 198, 103283.

[9] Hadi, H. J., Adnan, M., Cao, Y., Hussain, F. B., Ahmad, N., Alshara, M. A., & Javed, Y. (2024). iKern: Advanced Intrusion Detection and Prevention at the Kernel Level Using eBPF. Technologies, 12(8), 122.

[10] Zhang, J., Chen, P., & He, Z. (2024, June). Real-Time Intrusion Detection and Prevention with Neural Network in Kernel Using eBPF. In 2024 IEEE/IFIP International Conference on Dependable Systems and Networks (DSN).

[11] Miano, S., Risso, F., Vásquez Bernal, M., Bertrone, M., & Lu, Y. (2021). A Framework for eBPF-Based Network Functions in an Era of Microservices. IEEE Transactions on Network and Service Management, 18(1), 133–151

[12] Sadiq, A., Syed, H. J., Ansari, A. A., Ibrahim, A. O., Alohaly, M., & Elsadig, M. (2023). Detection of Denial of Service Attack in Cloud Based Kubernetes Using eBPF. Applied Sciences, 13(8), 4700.

[13] Bachl, M., Fabini, J., & Zseby, T. (2021). A flow-based IDS using Machine Learning in eBPF. arXiv:2102.09980.

[14] Tolay, M. (2025). eBPF-Based Real-Time DDoS Mitigation for IoT Edge Devices. arXiv:2508.00851.

[15] S. L. Narayanan, “X-IDS and R-XIDS: An eBPF-based solution for security attacks in IoT environments,” Master’s thesis, Purdue University Graduate School, USA, May. 2025. https://hammer.purdue.edu/articles/thesis/X-IDS_and_R-

XIDS_An_eBPF-based_solution_for_security_attacks_in_IoT_environments/28903928, （参照 2025-08-22）.
[16] Detection Frameworks and Latest Methodologies for eBPF-Based Backdoors. (2024). windshock blog.
[17] Smiru, R., et al. (2025). Securing the Cloud: Implementing eBPF for Efficient Packet Filtering and AI-Driven Traffic Classification. Journal of Emerging Technologies and Innovative Research (JETIR), 12(7).
[18] Chen, S.-S., Kuo, T.-Y., & Chen, Y.-W. (2013). Security Software Based on Windows NDIS Filter Drivers. In 2013 IEEE 37th International Computer Software and Applications Conference Workshops (COMPSACW) (pp. 260–264).
[19] He, C. (2007). Design and implementation of a personal firewall Based on NDIS Intermediate Drivers. In Proceedings of the 8th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD 2007).
[20] Guided tour inside WinDefender's network inspection driver. (2021, July). Quarkslab blog, R. Dumont.
[21] safest-place. ExploitPcapCollection. GitHub, n.d., https://github.com/safest-place/ExploitPcapCollection. （参照 2025-08-21）.
[22] Forescout Research Labs, "AMNESIA:33 – 33 vulnerabilities in open source TCP/IP stacks impact millions of IoT, OT and IT devices," Forescout, Dec. 2020. https://www.forescout.com/resources/amnesia33/, （参照 2025-08-22.
[23] JSOF Research, "Ripple20 – 19 zero-day vulnerabilities amplified by the supply chain," JSOF, Jun. 2020. https://www.jsof-tech.com/ripple20/, （参照 2025-08-22）.
[24] Cybersecurity and Infrastructure Security Agency (CISA), "Ripple20 vulnerabilities affecting Treck TCP/IP stack," CISA Alerts, Jun. 2020. https://www.cisa.gov/news-events/alerts/2020/06/16/ripple20-vulnerabilities-affecting-treck-ip-stacks, （参照 2025-08-22）.
[25] Cybersecurity and Infrastructure Security Agency (CISA), "NAME:WRECK – Vulnerabilities in TCP/IP stacks affecting DNS implementations," CISA Alerts, Apr. 2021. https://www.cisa.gov/news-events/alerts/2021/04/13/namewreck-dns-vulnerabilities, （参照 2025-08-22）.
[26] Armis Labs, "URGENT/11 – Critical vulnerabilities in VxWorks TCP/IP stack," Armis Security, Jul. 2019. https://www.armis.com/research/urgent11/, （参照 2025-08-22）.
[27] Forescout Research Labs and JFrog Security Research, "INFRA:HALT – 14 vulnerabilities in the NicheStack TCP/IP stack," Forescout, Aug. 2021. https://www.forescout.com/resources/infrahalt/, （参照 2025-08-22）.
[28] Siemens and Forescout Research Labs, "NUCLEUS:13 – Thirteen vulnerabilities in Nucleus RTOS TCP/IP stack," Siemens ProductCERT, Nov. 2021. https://cert-portal.siemens.com/productcert/pdf/ssa-044112.pdf, （参照 2025-08-22）.
[29] Microsoft Security Response Center, "Security Update Guide – CVE-2024-38063" (Update Guide for CVE-2024-38063), Microsoft. URL: https://msrc.microsoft.com/update-guide/vulnerability/CVE-2024-38063, （参照 2025-08-22）.