ハッキング競技 CTF を取り入れた情報教育のための教材作成支援システムの検討

松原 克弥　源 啓多

CTF (Capture The Flag) は，IT 技術に関する問題に対して適切な形で対処することで，その結果得られる得点で勝敗を決める競技である．CTF 競技では，計算機エンジニアリング分野に関する幅広い知識が求められ，実践的技術の習得も見込める．これらのイベント形式の学習は，参加者の技術習得に対する意欲が高く，アクティブラーニングによる実践的な教育効果が高いと推測される．しかし，実環境のシステム挙動やデバイス入出力への加工を必要とする CTF 問題作成は，出題分野に関する高度なハッキング能力が求められ，情報教育への導入障壁のひとつとなっている．本稿では，情報教育教材としての CTF 導入を目的として，CTF 問題作成支援のためのシステムを提案する．本システムでは，CTF で読み出す対象となる情報を様々なシステム媒体に対して透過的に仕込むことができ，かつ，容易には解き明かせないよう実装をブラックボックス化できる機構を実現する．また，CTF 参加者が自ら所有する PC 機材を使用できるように，特定の OS 環境に依存せずに実装できるように仮想化技術を活用して，OS よりも低レイヤで実装を行う．

CTF (Capture The Flag) is a game in which challengers compete that how fast find a flag obtained as a result by dealing with IT problems, such vulnerability and obfuscation, in a proper way. CTF challengers can deeply learn various subjects around computer engineering and acquire practical IT techniques. Such the event can sustain the participants' motivation of learning and then it can effectively encourage the active learning. However, CTF requires advanced skills of hacking system behavior and device I/Os for making contents. We think this may be one of the biggest barrier to introduce CTF into informatics education. In this paper, we propose a system for supporting making CTF contents as a material of informatics learning. The system implements mechanism for embedding flags transparently into various system media such display screen and network. Furturemore the processing code for embedding flags can be isolated from users. In addition, the virtualization technology allows CTF paticipants to bring their own PCs because the whole system would be implemented in a lower layer than OS.

1　はじめに

近年，ハッカソンやハンズオン，LT(Lightning Talk) 大会などの IT 技術の向上を目的とした教育系イベントが開催が国内外で増加している．また，情報セキュリティ分野では，CTF と呼ばれるコンテストが非常に注目されており，ホワイトハッカーと呼ばれる逸材の発掘やセキュリティに興味を持つ学生の IT 技術に関する研鑽に貢献している [4] [5] [6]．CTF は，IT 技術に関する問題に対して適切な形で対処することで，その結果得られる得点で勝敗を決める競技スタイルのイベントである．CTF における競技形式は 2 種類に分類でき，ファイルや画像，システム入出力データから指定された情報 (フラグ) を読み出す早さを競うクイズ形式と，各チームに与えられた脆弱性のあるシステムを攻撃から防御しつつ，他チームのシステムの脆弱性を突いて情報を読み出す攻防戦形式がある．CTF では，暗号や符号理論，信号処理，画像処理やネットワーク技術，プログラミング言語，データベース，ファイルシステムといった OS 技術などのソフトウェアエンジニアリング分野に関する幅広い知識が求められる．また，問題を解くことを通して，脆弱性解析やデバッグ技術，ログ解析やツールの活用などの実践的技術の習得が見込める．大学にお

A Study on Supporting Making CTF Materials for Informatics Learning.

Katsuya Matsubara, Keita Minamoto, 公立はこだて未来大学 システム情報科学部, School of Systems Information Science, Future University Hakodate.

表 1　Jeopardy の出題形式と CTFVisor との機能的適合性

出題形式	説明	CTFVisor との機能的適合性
Pwn	サーバ上で動くプログラムの脆弱性を攻撃して権限を奪取することで、サーバ上のフラグ (ファイル) にアクセスする	×
Reversing	プログラムバイナリを読み解いて動作を理解して，プログラムが持つ (生成する) フラグを見つける	×
Web	サーバ上で動作している Web サービスの脆弱性を見つけてフラグを得る	×
Crypto	暗号文を解読してフラグを読み解く	○
Network	ネットワークパケットのログなどを読み解いて、通信データ内に埋め込まれているフラグを見つける	○
Forensics	メモリダンプやディスクイメージ，USB などのデバイス入出力 RAW データを解析して，それらの中に埋め込まれたフラグを発見する	○
Stego	画像データや音声データに何らかの手法を用いて隠されているフラグ文字列を見つけ出す	○
Recon	Twitter，Facebook などの SNS や Internet Archive のデータを探索して，出題側が過去に発信したフラグを見つけだす	×
PPC	ネットワークを介して送られてくる競技プログラミングの課題を順に解いていく速さを競う	×

ける従来の講義・演習と比べて，これらのイベント形式の学習は，参加者の技術習得に対する意欲が高く，アクティブラーニングによる実践的な教育効果が高いと推測される．

これまでの CTF で用いられている題材は，画像や通信パケット等を直接加工した静的なデータや，Linux や Windows，Apache 等の特定バージョンのシステムに存在する既存の脆弱性を用いているものが多い．これら問題の作成には，高度な技術スキルを持つエンジニアが必要となる．例えば，ネットワークパケットに埋め込まれたフラグを見つける問題では，Ethernet パケットや TCP/IP，UDP/IP パケットの構造に対する深い知識を有したエンジニアが，Scapy [1] などの特殊なツールを使用してパケットデータを加工し，その通信を Wireshark [2] などのネットワークキャプチャ・ツールを用いてロギングすることで競技題材を作成する．

本研究は，データベースや OS 等のソフトウェアエンジニアリング教育において，当該分野の幅広い知識を活用する CTF を取り入れた実践的な学習の機会を提供することで，学生の当該教科に対する学習意欲の向上と卓越した高度 IT スキルを持つ人材の育成を目指している．本稿では，対象学生の興味と意欲を引きつける新たな CTF 教材の作成を支援するためのシステム CTFVisor の実現に関する検討について述べる．CTF 参加者が自ら所有する PC 機材を使用できるように，Windows や macOS，Linux 等の特定の OS 環境に依存せずに実装できるように仮想化技術を活用して，OS よりも低レイヤで実装を行う．CTF 教材作成者は，特殊なツールに関する知識を持たなくても，本システムの設定，もしくは，簡易な処理関数を記述することで意図する CTF コンテンツを作成できる．

CTFVisor で作成する CTF 教材では，仮想化技術を活用することで，通信データからファイル，デバイス I/O 等の幅広い媒体を対象として OS 透過的に CTF 問題を仕込むことができる．システム基盤で問題作成の支援機能を提供することで，各対象媒体への高度な技術スキルを持たない者でも出題が可能とな

り，大学等の情報教育における CTF 採用が広がることが期待できる．既存 CTF への参加経験が豊富な学生に対しても，仮想化技術による出題というこれまでにない形式により，新たな技術の習得や参加へのモチベーション向上に貢献することが期待できる．

以降，第 2 章では，CTF 競技における出題形式についてまとめ，本提案システムが対象とする CTF 問題形式について検討する．第 3 章では，CTF 形式の教材作成支援システム CTFVisor の機能と実現方式について述べる．第 4 章では，現在の実装状況について示す．最後に，第 5 章でまとめと今後の課題について述べる．

2 CTF の出題形式

CTF の出題形式は，Attach&Defence と呼ばれるサーバ攻略対戦形式と Jeopardy と呼ばれるクイズ形式の 2 種類のタイプに分類できる．前者は，脆弱性やアクセス制御などのセキュリティに特化した内容の CTF だが，後者は，暗号や符号理論，信号処理，画像処理やネットワーク技術，プログラミング言語，データベース，ファイルシステムといった OS 技術などの多岐にわたる分野の出題が可能であり，ソフトウェアエンジニアリング分野全般の実践的学習に活用することができる．そのため，本提案の CTFVisor では，Jeopardy (クイズ形式) の CTF 作問を支援対象とする．さらに，Jeopardy における問題は，形式に応じて Pwn, Reversing, Web, Crypto, Network, Forensics, Stego, Recon, PPC などのジャンルに分類できる (表参照)．Jeopardy の各形式について，CTFVisor で実現可能な機能との適合性について検討する．本提案の CTFVisor は，仮想化技術を使って，実際の PC やデバイスの動作に対して加工を行う機能を実現する．そのため，実動作させない Reversing や，プログラムの既存脆弱性を利用する形式の Pwn や Web といった形式の CTF 問題は，CTFVisor との機能的適合性が低い．また，インターネットなどの外部にあるサーバやデータを利用する Recon や PPC においても，CTF 参加者の持ち込み PC で動作する CTFVisor から作問支援することは難しい．Crypto では，ディスクや画面等のデバイスを介して暗号データを提示するような出題形式において，CTFVisor の機能の活用が考えられる．Network 形式では，CTFVisor が持つネットワークパケットの監視・加工機能により，従来の CTF 作問で用いている Scapy のような特殊なツールと同様の加工処理を実現できる．さらに，従来の CTF ではネットワークトラフィックのログデータによる出題しかできないが，CTFVisor では，CTF 参加者の持ち込み PC 上において，OS 環境への特殊なドライバ等のインストールなしに加工処理されたパケットを実際に送出するような出題が可能である．同様に，Forensics 形式においても，画面出力や音声出力を CTFVisor で加工することにより，PC での実動作を伴う出題が可能になる．

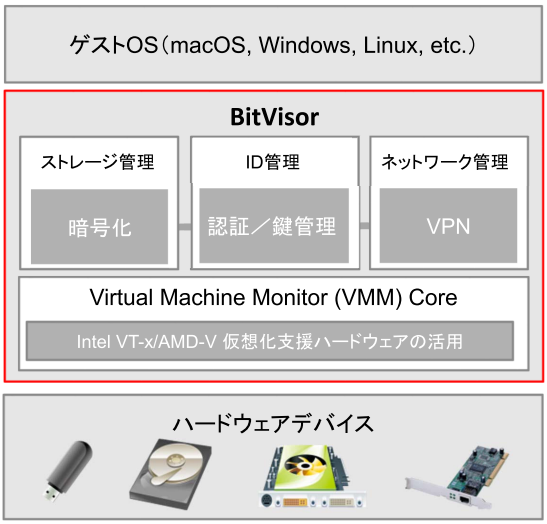

図 1　BitVisor のアーキテクチャ

3 CTF 作問支援システムの検討

本提案では，仮想化技術を用いて CTF 問題作成を支援する機能の実現を行う．提案手法の実現と実装には，軽量仮想マシンモニタ・ソフトウェア BitVisor [3] のデバイス入出力に対する透過的介在機能を利用する．

3.1 BitVisor

BitVisor は，ハードウェア上で直接動作する仮想マシンモニタとして設計・実装されている (図 1 参照)．

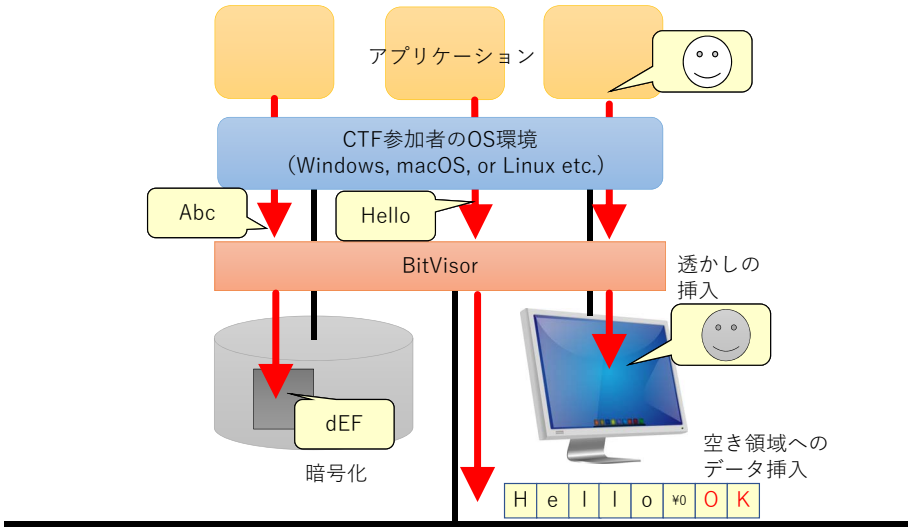

図 2　BitVisor を用いた CTF 問題向けデータ加工処理

PC の情報漏えい対策として研究開発された BitVisor は，OS 環境からのディスクやネットワーク，USB などへの入出力を監視して，必要に応じて暗号化やアクセス制御を行う機能を持つ．本提案の CTF 教材支援では，これらセキュリティ機能を実装するための基盤技術として実現されているデバイス入出力への透過的介在機能を用いる．BitVisor では，仮想的なデバイスを介して OS 環境を動作させるのではなく，実際のハードウェア上で動作する OS 環境に対して，監視対象のデバイスへの入出力のみを捕捉する．デバイスのインタフェース仕様が BitVisor 導入後も変わらないため，OS 環境に特別なデバイスドライバ等を追加インストールする必要がない．また，監視対象ではないデバイスは，OS から直接アクセスすることができるため，GPU などの性能を求められるデバイスに対する性能への影響を最小限にすることが可能となる．また，ハードウェアプラットフォーム全体の仮想化を必要としないため，BitVisor が専有するメモリ容量を 128MiB 程度に抑えることができる．BitVisor 上では，Linux や FreeBSD などのオープンソース OS だけでなく，Windows や macOS などの商用 OS を BitVisor 上で動作させることが可能となっている．

3.2　CTF 作問支援システム CTFVisor

第 2 章で述べたとおり，本提案では，Jeopardy の Crypto，Network，Forensics の各形式の CTF を対象として，作問を支援する機能を BitVisor に実装する．図 2 は，CTFVisor による CTF 問題に応じた入出力データの処理例を示している．Crypto 形式の CTF に対しては，アプリケーションからディスクへの書き込みを捉えて暗号化する．CTF 挑戦者は，様々なデータをディスクへ書き込むことで，CTFVisor による暗号化のアルゴリズムや鍵を推測し，ターゲットとなる暗号化済フラグデータを解読する．Network 形式 CTF に対しては，アプリケーションからのネットワーク送信のパケット内に CTFVisor がデータを挿入する．CTF 挑戦者は，様々なデータを送信して，その通信トラフィックをキャプチャし，CTFVisor が埋め込んでいるフラグを見つける．Forensics では，OS やアプリケーションの画面描画に対して，フラグデータを埋め込む透かし処理 (ステガノグラフィ) を施してから画面表示する．CTF 挑戦者は，画面描画とキャプチャを繰り返すことで，CTFVisor によって埋め込まれた画像片を見つけ出し，フラグを取り出す．

CTFVisor では，ネットワーク，ディスク，USB，画面のそれぞれに対する I/O 加工機能を実現する．ネットワーク I/O 加工機能では，OS デバイスドライ

バから Ethernet デバイスハードウェアへの送信指示を監視して，デバイスによる送信直前で送信データを加工するためのコールバック関数を呼び出す．CTF 作問者は，コールバック関数を記述して登録することで，フラグの埋め込み等のネットワークパケットに対する加工処理を実装できる．ディスク，USB デバイス，画面に対する I/O 加工機能も，ネットワーク I/O 加工機能と同様の流れで処理を実装する．現実装では，BitVisor の開発言語である C 言語を用いてコールバック関数を記述する．

4 実装

現在，CTFVisor のプロトタイプ実装を進めている．Network 形式の出題例として，OS やアプリケーションから出力されるネットワークパケットを監視して，Ethernet ヘッダや TCP/IP, UDP/IP ヘッダの空き領域へ指定したフラグを埋め込む機能を実装している．また，Forensics 形式の出題例として，スクリーン描画されるビットマップデータを表示直前に加工して，フラグを埋め込むような透かし処理の実装を検討している．画面描画への透かし処理では，任意のタイミングで発生する OS による加工前画面データの再描画により透かしデータが消失してしまうことにより，透かしデータの表示時間を調整できないことで，問題の難易度が意図せずあがってしまう課題がある．

5 まとめと今後の課題

本稿では，情報教育教材としての CTF 導入を目的として，CTF 問題作成支援のためのシステム CTFVisor を提案した．CTFVisor を用いることで，これまで特殊なツールを使って作成していた CTF 問題を CTF 参加者の PC 自体に組み込むことができ，対象となるフラグ情報を様々なシステム媒体に対して透過的に仕込むことができる．CTF 問題作成は，CTFVisor の設定ファイルや，CTFVisor 内にコールバック関数を登録するによりフラグ加工処理の記述が可能となる．従来の CTF で用いられているログ形式のデバイス I/O 情報による出題と比較して，CTF 参加者は，持ち込んだ PC 上で CTFVisor を動作させ，様々なデバイス I/O を通して作問者により組み込まれたフラグを見つけることができる．

今後，実装したシステムを用いた CTF 競技を開催して，CTF 参加者からのアンケート等を通してシステムが実現する CTF 教材の評価を進める．また，問題開示から回答までの時間や正答率から，仮想化技術を用いた問題作成が難易度や学習理解度に与える影響について考察する．さらに，CTF 参加者側の評価だけでなく，CTF 作問機能の評価方法についても検討する必要がある．高度な知識と特殊なツールを用いるを用いるこれまでの作問方法と比較して，本システム機能である CTF 作問支援機能の有効性について評価したい．

参 考 文 献

[1] Biondi, P., Valadon, G., and Lalet, P.: the Scapy network packet manipulation program, `http://www.secdev.org/projects/scapy/`.

[2] Combs, G.: the Wireshark network protocol analyzer, `https://www.wireshark.org`.

[3] Shinagawa, T., Eiraku, H., Tanimoto, K., Omote, K., Hasegawa, S., Horie, T., Hirano, M., Kourai, K., Oyama, Y., Kawai, E., Kono, K., Chiba, S., Shinjo, Y., and Kato, K.: BitVisor: A Thin Hypervisor for Enforcing I/O Device Security, *Proceedings of the 2009 ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments*, VEE '09, New York, NY, USA, ACM, 2009, pp. 121–130.

[4] 秋山卓巳, 前田優人, 矢倉大夢, 森越友祐, 竹迫良範: オンライン CTF の運営の裏側〜問題作成と攻撃対処〜, コンピュータセキュリティシンポジウム *2016* 論文集, Vol. 2016, No. 2, 情報処理学会, Oct. 2016, pp. 52–52.

[5] 赤木智史, 中矢誠, 富永浩之: ハッキング競技 CTF を取り入れた情報セキュリティ教育の導入イベントの実践報告, 情報教育シンポジウム *2014* 論文集, Vol. 2014, No. 2, 情報処理学会, Aug. 2014, pp. 169–172.

[6] 赤木智史, 中井智己, 中矢誠, 富永浩之: ハッキング競技 CTF を取り入れたセキュリティを意識させる情報リテラシ教育の大会イベント: 大会運営サーバの機能と初心者向けの問題設定, 電子情報通信学会技術研究報告, Vol. 114, No. 513(2015), pp. 39–44.