制御システムの可用性を考慮したプロセスの複製によるライブフォレンジック手法の提案

時松 勇介[1]　山内 利宏[1,a)]

概要：制御システムが攻撃を受けた際にはデジタルフォレンジックの適用が必要だが，制御システムは可用性を最優先するため，情報システムにおける既存手法の適用が難しい．本稿では制御システムの可用性を考慮したプロセスの複製によるライブフォレンジック手法を提案する．提案手法では，物理メモリの完全メモリダンプを取得するのではなく，プロセスを複製して複製プロセスからメモリイメージのスナップショットを取得する．複製時のメモリコピーを最小限にすることでシステムの可用性を保ち，メモリイメージを取得することでファイルシステムに痕跡を残さない攻撃に対処できる．また，制御システムへの適用可能性を評価し，考察した結果を報告する．

キーワード：制御システム，ライブフォレンジック，ファイルレスマルウェア，オペレーティングシステム

1. はじめに

従来，電力やガスなどの重要インフラに用いられる制御システムは，独自技術を利用し，閉じたネットワーク環境で運用されてきたため，サイバー攻撃による脅威とはほとんど無縁であると考えられてきた．しかし，近年では，システムの構築や運用にかかるコストの削減を目的として，汎用技術の導入や標準プロトコルの採用が進展している [1][2]．また，競争力強化のため，情報システムとの連携が進み，ネットワークを介して制御システムに接続する事例も出てきている [1][2]．このような状況に伴い，制御システムを標的とするサイバー攻撃が表面化しており，制御システムに対して適切なセキュリティ対策を施すことが求められている [1][2]．

制御システムがサイバー攻撃による被害を受けた場合，被害拡大防止や再発防止のため，デジタルフォレンジックの適用による早急な原因究明に取り組む必要がある．ここで，デジタルフォレンジックとは，システム上で事件や事故が発生した際に，不正者や犯罪者の特定，事故原因の究明，および責任の究明のため，システム内に残る電磁的証拠を調査する手段や技術を指す [3]．

ただし，制御システムと情報システムは，運用上の違いから情報セキュリティに対する考え方が異なっており，情報システムが機密性を重視するのに対して，制御システムは可用性を重視する．既存のデジタルフォレンジック手法は，情報システムの特徴にあわせて設計されていることから，制御システムへの適用が困難となる可能性がある．また，従来，攻撃の痕跡はファイルシステムに残されていたことから，既存のデジタルフォレンジック手法の多くは，HDD から電磁的証拠を取得する．ただし，近年，ファイルシステムに攻撃の痕跡を残さない攻撃手法が増加しており [4]，制御システムにおいてもこの攻撃に対処できるデジタルフォレンジック手法が必要である．

そこで，本稿では，制御システムの可用性を考慮したプロセスの複製によるライブフォレンジック手法（以降，提案手法と呼ぶ）を提案する．ライブフォレンジックとは，デジタルフォレンジックの一種であり，稼動状態の計算機から電磁的証拠を取得し，取得した電磁的証拠を解析する手法を指す．提案手法では，制御システムがサイバー攻撃による被害を受けた場合，その攻撃の痕跡は動作中のプロセスに残されている可能性が高いと考え，動作中のプロセスに対してライブフォレンジックを適用する．ただし，制御処理を行うプロセスからの電磁的証拠の取得は，プロセスが本来行うべき制御処理に大幅な遅延を発生させ，制御システムの可用性を損なう恐れがある．このため，提案手法では，対象となるプロセスを複製し，複製プロセスに対象プロセスの複製時点での状態を保持させる．プロセス複製時のメモリコピーを最小限に抑え，複製プロセスから電磁的証拠を取得することにより，対象プロセスが行う制御処理の遅延を抑え，制御システムの可用性を維持できる．

[1] 岡山大学 大学院自然科学研究科
[a)] yamauchi@cs.okayama-u.ac.jp

表 1　制御システムと情報システムの運用上の違い

	制御システム	情報システム
求められる処理内容	リアルタイムかつ周期的な制御処理	非リアルタイムで高速な処理
システムの更新間隔	10–20 年	3–5 年
稼働時間	24 時間 365 日	業務時間内
セキュリティの対象	モノ（設備，製品），サービス（連続稼動）	情報
攻撃の被害	国民生活に影響大	情報漏洩，金銭的損失
セキュリティ優先順位	可用性 > 完全性 > 機密性	機密性 > 完全性 > 可用性

また，電磁的証拠としてプロセスのメモリイメージのスナップショットを取得することにより，ファイルシステムに攻撃の痕跡を残さない攻撃手法に対処できる．

2. 制御システムにおけるデジタルフォレンジック

2.1 制御システム

制御システムは，運用上の違いから，情報システムとは情報セキュリティに対する考え方が異なる．文献 [2][5] を参考に，制御システムと情報システムの運用上の違いを**表 1** に示す．なお，表 1 では，制御システムとして，電力やガスなどの重要インフラ制御システムを想定している．情報システムについては，一般的な業務を行う個人向けシステムなどの特徴を示している．

制御システムは，重要インフラを支えるシステムであり，予期せぬシステムの停止や誤作動は国民生活に大きな影響を及ぼすことが予想される．このことから，制御システムでは，24 時間 365 日の安定稼動が求められる．また，制御システムの処理はリアルタイムかつ周期的であり，これを継続的に実行する必要がある．制御処理の大幅な遅延は，制御システムの制御不全を引き起こす恐れがあることから，制御処理を大幅に遅延させるような処理の追加は許容されない．

以上のような特徴を持つことから，制御システムでは，情報システムとは異なり，可用性が最も重視される．

2.2 デジタルフォレンジック

デジタルフォレンジックとは，システム上で事件や事故が発生した際に，不正者や犯罪者の特定，事故原因の究明，および責任の究明のため，システム内に残る電磁的証拠を調査する手段や技術を指す [3]．デジタルフォレンジック研究会 [6] では，インシデントレスポンスや法的紛争・訴訟に際し，電磁的記録の改ざん・毀損等についての分析・情報収集等を行う一連の科学的調査手法・技術として定義されている．基本的なデジタルフォレンジックの工程は，電磁的証拠の収集，保全，解析，および報告の 4 段階から構成される．また，稼動状態の計算機から電磁的証拠を収集し，収集した電磁的証拠を解析する手法を特にライブフォレンジックという．

2.3 関連研究

文献 [7] では，dd コマンドを用いて動作中の計算機から HDD のイメージファイルを取得する手法がシステムに与える負荷を測定することにより，制御システムへの適用可能性を評価している．また，文献 [8] では，文献 [7] の手法の実行による制御処理の遅延時間を測定し，制御システムへの適用可能性を評価している．

文献 [9] では，ホットデジタルフォレンジックという方式を提案している．ホットイメージバックアップ技術を用いた定期的な電磁的証拠の収集と，制御システムから切り離された環境でのホットイメージバックアップの差分解析により，インシデントの取りこぼしとシステムの性能低下を防ぐことを目的としている．

文献 [10] では，平常時には制御システムの状態を監視するモニターモードを動作させ，異常を検出した場合には電磁的証拠を収集するフォレンジックモードに切り替える方式を提案している．また，低負荷な電磁的証拠の収集ツールの実現を課題として挙げており，これの対処としてプロセスやメモリに関する電磁的証拠の収集を検討している．

文献 [11] では，制御システムの制御機器から電磁的証拠を収集する方法として，システム上に電磁的証拠の収集を目的とした装置を配置する方式を提案している．また，制御システムにおいては，制御処理の実行による電磁的証拠の上書きが起こりやすいことから，迅速なインシデントレスポンスが求められることを述べており，これの支援のために制御システムにおけるデジタルフォレンジックの実施モデルを提案している．

文献 [7]–[11] では，制御システムに適用可能なデジタルフォレンジック手法について検討しているものの，ファイルシステムに攻撃の痕跡を残さない攻撃手法については考慮されておらず，制御システム向けのライブフォレンジック手法の検討は十分とはいえない．

2.4 制御システムにおけるデジタルフォレンジックの課題

制御システムにおけるデジタルフォレンジックの課題を以下に示す．

(課題 1) ファイルシステムに攻撃の痕跡を残さない攻撃手法に対処不可

制御システムがファイルシステムに攻撃の痕跡を残さない攻撃手法による被害を受けた場合，HDD から電磁的証拠を取得する手法 [7][8][9] では対処できない．また，文献 [10][11] についても，ファイルシステムに攻撃の痕跡を残さない手法への対処は考慮されていない．さらに，制御システムにおいては可用性が重視さ

れることから，可用性を損なわないデジタルフォレンジック手法が必要である．

(課題 2) 制御システムの可用性の低下

情報システムにおける既存手法は，制御システムでの利用は考慮していないことから，制御システムの可用性を損なう恐れがある．また，文献 [8] の評価では，dd コマンドを用いた HDD のイメージファイルの作成によって発生する最大遅延時間が非常に大きくなっており，制御システムの可用性を低下させる恐れがある．さらに，(課題 1) に対処するためには，制御システムに対してメモリフォレンジックを適用する必要がある．メモリフォレンジックとは，デジタルフォレンジックの一種であり，メモリから電磁的証拠を取得して解析する手法である．主要なメモリフォレンジックツールに Volatility[12] があり，メモリダンプを解析することにより，メモリ上の様々な情報を収集できる．ただし，制御システムにおいては，物理メモリの完全メモリダンプの取得は，メモリ上に存在するプロセスが本来行うべき制御処理に大幅な遅延を発生させ，システムの可用性を低下させる恐れがある．

3. 制御システムの可用性を考慮したプロセスの複製によるライブフォレンジック手法

3.1 要件

本稿では，近年，ファイルシステムに攻撃の痕跡を残さない攻撃手法が増加していることに着目し，制御システムの可用性を低下させずに，ファイルシステムに攻撃の痕跡を残さない攻撃手法に対処できるライブフォレンジック手法を実現する．要件を以下に示す．

(要件 1) メモリ上の電磁的証拠の取得と解析

ファイルシステムに攻撃の痕跡を残さない攻撃手法による被害を受けた場合，その攻撃の痕跡はメモリ上に残されていることが多い．ファイルシステムに攻撃の痕跡を残さない攻撃手法に対処するためには，メモリから電磁的証拠を取得し，解析する必要がある．

(要件 2) ある時点におけるメモリの状態の保存

メモリから電磁的証拠を取得している間にメモリの状態が変化した場合，得られた電磁的証拠に整合性がとれず，電磁的証拠の解析に支障をきたす可能性がある．このため，ある時点のメモリの状態を保存し，得られる電磁的証拠の整合性を保つ必要がある．

(要件 3) 低負荷な電磁的証拠の取得

電磁的証拠を取得する際に高い処理負荷が発生した場合，制御システムが本来行うべき制御処理に大幅な遅延が発生し，制御不全に陥る可能性がある．制御システムの可用性を維持するため，電磁的証拠を取得する際に発生する処理負荷を小さくする必要がある．

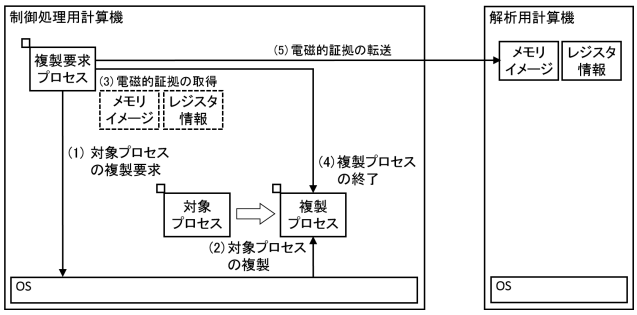


図 1 提案手法の全体像

3.2 設計方針

制御システムがサイバー攻撃による被害を受けた場合，その攻撃の痕跡は動作中のプロセスに残されている可能性が高いと考えられる．そこで，提案手法では，動作中のプロセスに対してライブフォレンジックを適用する．

まず，(要件 1) を満たすために，電磁的証拠としてプロセスのメモリイメージのスナップショットを取得する．これにより，メモリ上に残された攻撃の痕跡を解析でき，ファイルシステムに攻撃の痕跡を残さない攻撃手法に対処できる．

また，(要件 2) を満たしつつ，(要件 3) を実現するため，ライブフォレンジックの適用対象となるプロセスを複製する．複製プロセスに対象プロセスの複製時点での状態を保持させ，複製プロセスから電磁的証拠を取得する．これにより，プロセスから電磁的証拠を取得する際，プロセスが本来行うべき制御処理を停止する必要が無くなるため，制御システムの可用性の低下を抑えることができる．

さらに，(要件 3) への対処として，プロセスを複製する際のメモリコピーを削減することにより，発生する処理負荷や処理時間を抑える．

また，制御システムの可用性を維持するため，デジタルフォレンジックの適用によって制御システムに追加される処理は最小限にする必要がある．そこで，提案手法では，制御システム上で取得した電磁的証拠を制御システムから切り離された環境に転送し，以降の工程は制御システムから切り離された環境で実施する．これにより，制御システムに追加される処理が電磁的証拠の取得と転送に絞られるため，デジタルフォレンジックの適用による制御システムへの影響を抑えることができる．

3.3 基本方式

提案手法の全体像を**図 1** に示し，以下で処理の流れを説明する．

(1) ライブフォレンジックの適用対象プロセスの複製を OS に要求

(2) 対象プロセスの複製を生成

複製プロセスは複製時点での状態を保持

(3) 複製要求プロセスは，複製プロセスから電磁的証拠を取得
(4) 電磁的証拠の取得後，複製プロセスを終了
(5) 複製プロセスから取得した電磁的証拠を制御システムから切り離された環境に転送

提案手法は，対象プロセスに関する電磁的証拠を複製プロセスから取得するため，複製プロセスは，複製時点での対象プロセスの状態を保持する必要がある．複製プロセスが処理を開始した場合には，複製プロセスの状態が変化して電磁的証拠としての有効性を失うことから，複製プロセスは，生成された後，処理を開始せずに複製時点での状態を保持する．

また，提案手法では，取得した電磁的証拠を制御システムから切り離された環境に転送する．このため，以降の工程については，制御システムの可用性への影響を考慮する必要はない．

3.4 取得する電磁的証拠

ファイルシステムに攻撃の痕跡を残さない攻撃手法に対処するためには，メモリから電磁的証拠を取得し，メモリ上に残された攻撃の痕跡を解析する必要がある．そこで，提案手法では，プロセスのメモリイメージのスナップショットを取得する．プロセスのメモリ上に攻撃の痕跡が残っていた場合，得られたスナップショットを解析することにより，ファイルシステムに攻撃の痕跡を残さない攻撃手法に対処できる．

また，提案手法では，メモリイメージのスナップショットに加え，レジスタ情報を取得する．レジスタ情報は電磁的証拠取得時点でのプロセッサの内部状態を表しており，有用な情報が得られる可能性が高い．例えば，マルウェアにより，レジスタが書き換えられていた場合には，レジスタ情報を解析することで検出できる場合がある．

3.5 期待される効果

提案手法を適用することにより，期待される効果を以下に示す．

(効果 1) ファイルシステムに攻撃の痕跡を残さない攻撃手法に対処可能

ファイルシステムに攻撃の痕跡を残さない攻撃手法による被害を受けた場合には，メモリ上に攻撃の痕跡が残されていることが多い．提案手法では，電磁的証拠としてプロセスのメモリイメージのスナップショットを取得することにより，メモリ上に残された攻撃の痕跡を解析できる．このため，制御システムに限らず，ファイルシステムに攻撃の痕跡を残さない攻撃手法に対処できる．

(効果 2) 制御システムの可用性を維持

提案手法では，ライブフォレンジックの適用対象となるプロセスを複製し，複製プロセスから電磁的証拠を取得する．また，プロセスを複製する際のメモリコピーを削減することにより，処理負荷や処理時間を抑える．これにより，プロセスが本来行うべき制御処理に与える影響を抑え，制御システムの可用性の維持を可能にする．

4. 実現方式

4.1 実現における課題

3.2 節に述べた設計を実現するために，以下に示す課題に対処する必要がある．

(課題 1) メモリコピーを最小限とするプロセスの複製

提案手法では，プロセスを複製し，複製プロセスから電磁的証拠を取得する．制御システムの可用性を維持するために，プロセスを複製する際の処理負荷や処理時間は，最小限に抑える必要がある．このため，メモリコピーを最小限とするプロセスの複製を実現する必要がある．

(課題 2) プロセスのメモリイメージのスナップショットの取得

提案手法では，ファイルシステムに攻撃の痕跡を残さない攻撃手法に対処するため，プロセスのメモリイメージのスナップショットを取得する．このため，プロセスのメモリイメージのスナップショットの取得を実現する必要がある．

(課題 3) レジスタ情報の取得

レジスタ情報はライブフォレンジックの適用時点におけるプロセッサの内部状態を表しており，有用な情報を得られる可能性が高いことから，提案手法では，電磁的証拠としてレジスタ情報を取得する．このため，レジスタ情報の取得を実現する必要がある．

(課題 4) 制御システムの可用性を考慮した電磁的証拠の転送

提案手法では，制御システムへの影響を抑えるため，取得した電磁的証拠を制御システムから切り離された環境に転送する．このため，制御システムの可用性を考慮した電磁的証拠の転送を実現する必要がある．

本稿では，(課題 1) と (課題 2) について対処し，(課題 3) と (課題 4) については今後の課題とする．なお，取得した電磁的証拠については，暫定的対処として，USB 経由で外付け HDD に保存する．また，マルチコア環境では，プロセスを複製している間にも対象プロセスが動作する可能性があり，システムに予期せぬ不具合を引き起こすことが考えられる．このため，本稿では，シングルコア環境で提案手法を実現する．

4.2 メモリコピーを最小限とするプロセスの複製

メモリコピーを最小限とするプロセスの複製を実現する

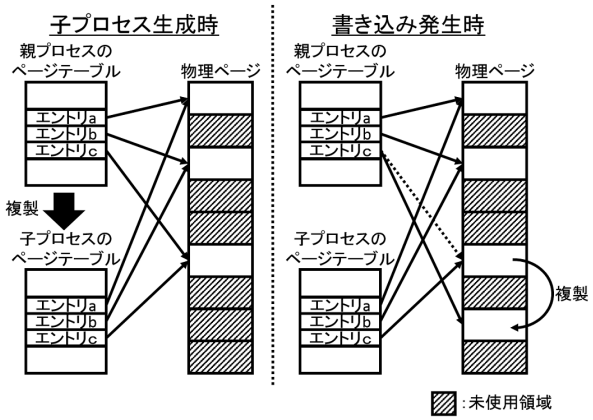


図 2　コピーオンライト機構の動作

表 2　指定したプロセスの複製を生成するシステムコール

形式	`pid_t create_proc_snap(pid_t pid)`
引数	`pid_t pid`：複製元プロセスのプロセス ID
戻り値	成功：複製プロセスのプロセス ID 失敗：−1
機能	呼出元の子プロセスとして，引数に指定されたプロセス ID に対応するプロセスの複製を生成する．生成された子プロセスは，ランキューに登録されずに生成時点での状態を保持する．

表 3　評価環境

CPU	Intel(R) Core(TM) i5-4460，3.20GHz
メモリ	8GB
OS	Ubuntu 14.04.3 LTS
カーネル	Linux 3.13.0-67，64bit

ため，fork() システムコールの実装を改変し，指定したプロセスの複製を生成するシステムコールを新たに作成する．

fork() システムコールは，呼び出し元プロセスの複製を生成するシステムコールである．内部処理として，呼び出し元プロセスの `task_struct` 構造体のポインタを用いて呼び出し元プロセスの各データ構造を複製する．これにより，呼び出し元プロセスの複製の生成を実現している．また，fork() システムコールは，コピーオンライト機構を利用しており，メモリコピーを最小限に抑えつつ呼び出し元プロセスの複製を生成できる．プロセスの複製におけるコピーオンライト機構の動作を**図 2** に示す．コピーオンライト機構では，子プロセス生成時に親プロセスの物理ページを複製するのではなく，親子間で物理ページを共有する．親子間で共有されている物理ページに対して書き込み要求が発生した時点で初めて物理ページを複製することにより，メモリコピーを最小限に抑えた効率的なプロセスの複製を実現している．図 2 では，例として，親プロセスでページテーブルのエントリ c に対応する物理ページへの書き込み要求が発生した場合の動作を示している．該当する物理ページが共有されていることから，親プロセスのページテーブルのエントリ c に対応する物理ページが複製される．これにより，親子間での物理ページの共有が解かれ，物理ページへの書き込みが実行される．

メモリコピーを最小限とするプロセスの複製を実現するために，新たに作成するシステムコールのインタフェースを**表 2** に示す．プロセスの複製は，ライブフォレンジックの適用対象となるプロセスとは別のプロセスから実行されるため，新たに作成するシステムコールでは，指定したプロセスの複製の生成を実現する必要がある．このため，新たに作成するシステムコールでは，引数として複製元プロセスのプロセス ID を指定する．内部処理として，引数として与えられたプロセス ID に対応するプロセスの `task_struct` 構造体のポインタを取得する．これを用いて複製元プロセスの各データ構造を複製することにより，指定したプロセスの複製の生成を実現する．また，複製プロセスは，複製された時点でのプロセスの状態を保持する必要がある．このため，複製プロセスは，ランキューに登録せず，生成された時点の状態を保持させる．

4.3　プロセスのメモリイメージのスナップショットの取得

プロセスのメモリイメージのスナップショットの取得を実現する方法として ptrace() システムコールによるメモリ読み出しを用いる方法がある．ptrace() システムコールは，プロセスの実行の追跡や，コアイメージやレジスタの調査を行う手段を提供するシステムコールである．第 1 引数として PTRACE_PEEKTEXT または PTRACE_PEEKDATA を指定することにより，プロセスから 1 ワードのメモリを読み出すことができる．ただし，ptrace() システムコールを用いるメモリ読み出しでは一度に読み出すことのできるメモリサイズが小さく，読み出すメモリサイズが大きい場合にはシステムコール呼び出しによるオーバヘッドが問題となる．

このため，提案手法では，`/proc/<pid>/mem` インタフェースを用いてプロセスのメモリイメージのスナップショットの取得を実現する．`/proc/<pid>/mem` インタフェースからは，read() システムコールを用いることにより，プロセスのメモリを任意のサイズで読み出すことができる．これにより，システムコール呼び出しによるオーバヘッドを削減できるため，プロセスのメモリイメージのスナップショットの取得にかかる処理時間を短縮できる．なお，ptrace() システムコールを用いる方式と `/proc/<pid>/mem` インタフェースに対して read() システムコールを用いる方式について，5 章で評価を実施し，両者の方式によるメモリ読み出し時間を比較している．

5.　評価

5.1　評価項目と評価環境

評価項目を以下に示す．また，評価環境を**表 3** に示す．

表 4　プログラムサイズ（単位：B）

テキスト部	978,305
データ部	36,536
BSS 部	23,512
合計	1,038,353

(評価 1)　複製プロセスから得られるメモリのスナップショットの完全性の検証
複製元プロセスと複製プロセスからそれぞれメモリのスナップショットを取得して比較することにより，両者のメモリのスナップショットが同一であることを確認する．

(評価 2)　プロセスの複製と終了にかかる処理時間の測定
ライブフォレンジックの適用対象となるプロセスの複製と終了にかかる処理時間を測定し，プロセスの複製によるシステムへの影響を評価する．

(評価 3)　メモリ読み出し処理時間の測定
提案手法におけるプロセスのメモリイメージのスナップショットの取得について，ptrace() システムコールを用いる方式と/proc/<pid>/mem インタフェースに対して read() システムコールを用いる方式のメモリ読み出し処理時間をそれぞれ測定し，両者の方式を比較する．

(評価 4)　コピーオンライト機構による複製元プロセスへの影響の評価
提案手法では，コピーオンライト機構を利用しているため，頻繁にメモリ領域を更新するプロセスに対して提案手法を適用した場合には，更新時に遅延が発生する．そこで，頻繁にメモリ領域を更新するプロセスに提案手法を適用した場合の遅延時間を測定し，その影響について考察する．

5.2　複製プロセスから得られるメモリのスナップショットの完全性の検証

複製元プロセスと複製プロセスからそれぞれメモリのスナップショットを取得し，両者から得られるメモリのスナップショットが同一であることを確認する．本実験では，UNIX で用いられるシェルの 1 つである bash を対象プロセスとし，複製元プロセスと複製プロセスから得られるメモリのスナップショットを比較する．実験に使用する bash のプログラムサイズを**表 4** に示す．

複製元プロセスと複製プロセスから得られたメモリのスナップショットの一部を**図 3** に示す．複製元プロセスと複製プロセスから得られたメモリのスナップショットを diff コマンドおよび cmp コマンドを用いて比較し，両者が同一であることを確認した．以上より，複製元プロセスと複製プロセスから得られるメモリのスナップショットが同一であることを確認した．

表 5　プロセスの複製と終了にかかる処理時間（単位：μs）

メモリサイズ	複製	終了	合計
28,176KB	42.24	155.04	197.28

5.3　プロセスの複製と終了にかかる処理時間の測定

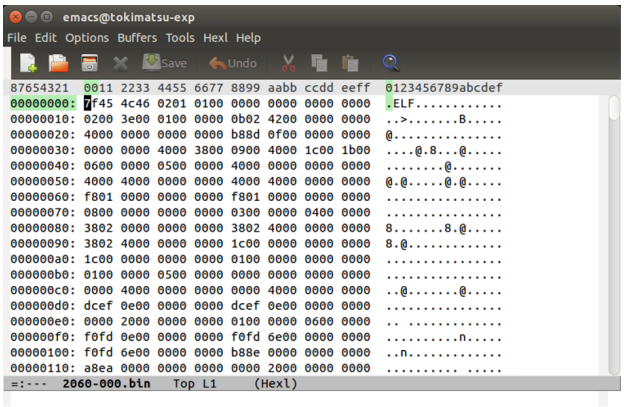
提案手法において，プロセスの複製と終了にかかる処理時間を測定した．本実験では，bash を対象プロセスとし，プロセスの複製と複製プロセスの終了を含む提案手法の一連の処理を 1000 回実行する．プロセスの複製では，提案手法のために新たに実現した create_proc_snap() システムコールの処理時間を測定する．複製プロセスの終了では，プロセスを終了させるために第 1 引数に SIGKILL を指定した kill() システムコールを実行し，その処理時間を測定する．対象プロセスの複製と複製プロセスの終了について，それぞれ 1 回当たりの平均処理時間を算出し，制御システムへの影響について評価した．

測定結果を**表 5** に示す．プロセスの複製にかかる処理時間は 42.24μs であり，コピーオンライト機構による高速なプロセスの複製が実現できていることが確認できる．また，プロセスの複製と解放にかかる処理時間は 197.28μs であることから，プロセスの複製と解放によるシステムへの影響は小さいと推測できる．ただし，プロセスを複製することによるメモリ使用量の増加もシステムに影響を与える要因であるため，今後，これについて評価する必要がある．

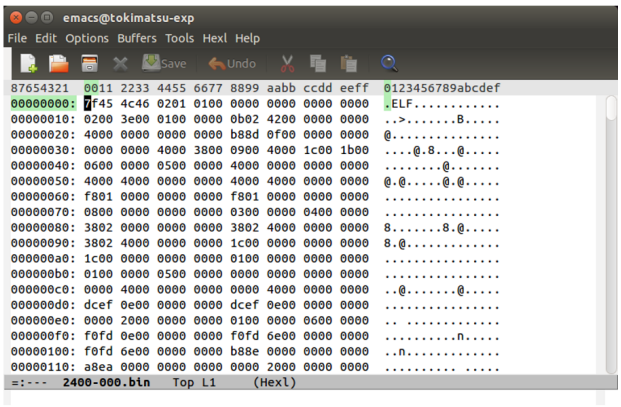
5.4　メモリ読み出し処理時間の測定

プロセスのメモリイメージのスナップショットの取得において，ptrace() システムコールを用いる方式と/proc/<pid>/mem インタフェースに対して read() システムコールを用いる方式のメモリ読み出し処理時間を比較した．本実験では，bash を対象プロセスとし，bash のメモリのテキスト部から指定したサイズのメモリ読み出しを 1000 回実行する．ptrace() システムコールを用いる方式では，第 1 引数に PTRACE_PEEKTEXT を指定した ptrace() システムコールによる指定サイズのメモリ読み出しに要する処理時間を測定する．/proc/<pid>/mem インタフェースに対して read() システムコールを用いる方式では，open() システムコールによる/proc/<pid>/mem インタフェースのオープン，lseek() システムコールによるファイルオフセットの設定，read() システムコールによる指定サイズのメモリ読み出し，および close() システムコールによる/proc/<pid>/mem インタフェースのクローズに要する処理時間を測定する．両者の方式において，1 回当たりの平均処理時間を算出し，その差を比較した．

測定結果を**表 6** に示す．また，両者の方式におけるシステムコール呼び出し回数を**表 7** に示す．表 6 に示す処理時間から，/proc/<pid>/mem インタフェースに対して read() システムコールを用いる方式の方が，より高速なメ



(a) 複製元プロセス



(b) 複製プロセス

図 3　プロセスから得られるメモリのスナップショット（一部）

表 6　メモリ読み出し処理時間（単位：μs）

読み出しサイズ	read() 方式	ptrace() 方式	read() に対する ptrace() の処理時間比
1KB	5.07	28.31	5.58
64KB	12.47	1,705.59	136.76
256KB	35.43	6,845.02	193.21

表 7　システムコール呼び出し回数

読み出しサイズ	read() 方式				ptrace() 方式
	open()	lseek()	read()	close()	ptrace()
1KB	1	1	1	1	128
64KB	1	1	1	1	8,192
256KB	1	1	1	1	32,768

モリ読み出しを実現できていることを確認できる．

また，表 6 からは，読み出すメモリサイズの増加に伴い，`/proc/<pid>/mem` インタフェースに read() システムコールを用いる方式に対して ptrace() システムコールを用いる方式の処理時間比が大きくなっていることを確認できる．この要因として，表 7 の結果から分かるように両者の方式におけるシステムコール呼び出し回数の差が挙げられる．また，別の要因として，read() システムコールによるメモリ読み出しでは，指定した領域のメモリを共有することにより，メモリコピーを最小限に抑え，効率的なメモリ読み出しを実現しているのではないかと予測している．この要因については，今後調査する必要がある．

5.5　コピーオンライト機構による複製元プロセスへの影響の評価

提案手法はコピーオンライト機構を用いてプロセスを複製していることから，頻繁にメモリ領域を更新するプロセスに対して提案手法を適用した場合，その領域の最初のメモリ内容更新時に性能低下が発生する．そこで，指定したサイズのメモリ領域を定期的に更新するプロセスに対して提案手法を適用し，提案手法の適用中に指定したサイズのメモリ領域の更新が行われた場合の処理時間を測定した．提案手法の適用の有無によるメモリ領域の更新処理時間の差を遅延時間とし，提案手法の適用による性能低下を評価した．評価に使用したプログラムのサイズを**表 8** に示す．評価では，確保および更新するメモリ領域のサイズが 16KB，64KB，256KB，1,024KB の場合について，それぞれ 500 回測定を行い，最小遅延時間，最大遅延時間，および平均遅延時間を算出した．

表 8　プログラムサイズ（単位：B）

テキスト部	2,528
データ部	616
BSS 部	16
合計	3,160

測定結果を**表 9** に示す．また，メモリコピーによって発生する遅延時間を**図 4** に示す．更新するメモリ領域のサイズが 16KB，64KB，および 256KB の場合を見ると，平均遅延時間および最大遅延時間ともに 0.1ms 以下に収まっている．文献 [13] では，制御システムの通信における発信から応答までの時間は 0.25ms〜10ms とされていることから，これらの場合の遅延時間は十分許容範囲に含まれているといえる．また，更新するメモリ領域のサイズが 1,024KB の場合の遅延時間は 0.25ms を超過しているが，制御システムにおける 1 回の通信処理の間に 1,024KB ものメモリ更新処理が発生するとは考えにくい．このため，コピーオンライト機構によるメモリコピーは，制御システムの可用性を大きく低下させるほどの遅延を発生させることは無いと考えられる．なお，表 9 の 1 ページ（4KB）当たりの遅延時間を見ると，最大遅延時間が 0.0015ms〜0.0035ms であることから，更新するメモリ領域のサイズが 284KB〜664KB 以下の場合であれば遅延時間は 0.25ms 以下に収まり，許容範囲に含まれると推測できる．これについては，制御システムにおける 1 回の通信処理の間に更新されるメモリ領域のサイズを調査し，改めて考察する必要がある．

表 9　メモリ更新処理時間（単位：ms）

更新	提案手法非適用時の処理時間	提案手法適用時の処理時間			遅延時間（括弧：1 ページ当たりの遅延時間）		
サイズ	平均	最小	平均	最大	最小	平均	最大
16KB	0.008	0.014	0.015	0.022	0.006（0.0015）	0.008（0.0020）	0.014（0.0035）
64KB	0.031	0.049	0.053	0.062	0.018（0.0011）	0.022（0.0014）	0.031（0.0019）
256KB	0.124	0.191	0.204	0.220	0.067（0.0010）	0.080（0.0013）	0.096（0.0015）
1,024KB	0.495	0.758	0.792	0.887	0.263（0.0010）	0.297（0.0012）	0.392（0.0015）

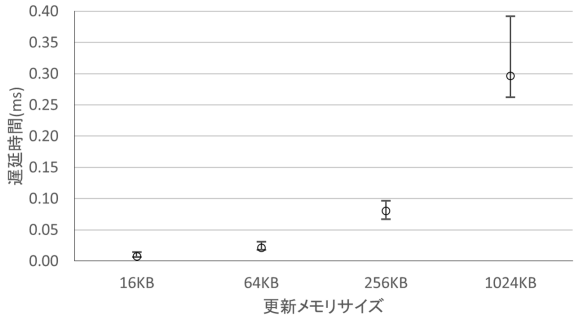


図 4　メモリコピーによって発生する遅延時間

6. おわりに

制御システムの可用性を考慮したプロセスの複製によるライブフォレンジック手法を提案し，実現方式と評価について述べた．

提案手法では，動作中のプロセスに対してライブフォレンジックを適用する．対象となるプロセスが本来行うべき制御処理への影響を抑えるため，対象プロセスを複製し，複製プロセスから電磁的証拠を取得する．また，コピーオンライト機構を用いることで，プロセスの複製時に発生するメモリコピーを最小限に抑え，制御システムの可用性の低下を抑える．さらに，電磁的証拠としてプロセスのメモリイメージのスナップショットを取得することで，ファイルシステムに攻撃の痕跡を残さない攻撃手法に対処できる．

評価では，まず，複製元プロセスから得られるメモリのスナップショットと複製プロセスから得られるメモリのスナップショットが同一であることを確認した．次に，プロセスの複製と解放にかかる処理時間を測定し，メモリコピーを最小限とするプロセスの複製を実現できていることを確認した．さらに，プロセスのメモリイメージのスナップショットの取得について，ptrace() システムコールを用いる方式と read() システムコールを用いる方式の処理時間を比較し，read() システムコールを用いる方式の方が高速であることを確認した．最後に，コピーオンライト機構による複製元プロセスへの影響を評価し，コピーオンライト機構を用いることによって発生するメモリコピーは，制御システムの可用性に大きな影響を与えないことを示した．

今後の課題として，レジスタ情報の取得，電磁的証拠の転送の実現，およびマルチコア環境への対応が挙げられる．また，提案手法の有用性を評価するため，既存のメモリフォレンジックツールと提案手法を比較する必要がある．

謝辞　本研究の一部は，科学研究費補助金基盤研究（B）（課題番号：16H02829）による．

参考文献

[1] IPA：2010 年度 制御システムの情報セキュリティ動向に関する調査，入手先 ⟨http://www.ipa.go.jp/files/000014121.pdf⟩，（参照 2015-12-21）．

[2] IPA：重要インフラの制御システムセキュリティと IT サービス継続に関する調査報告書，入手先 ⟨https://www.ipa.go.jp/files/000013981.pdf⟩，（参照 2015-12-22）．

[3] 上原哲太郎：ディジタルフォレンジック-電磁的証拠の収集と分析の技術-，会誌「情報処理」，Vol.48，No.8，pp.889-898．

[4] McAfee Labs：McAfee 脅威レポート：2015 年第 3 四半期，入手先 ⟨http://www.mcafee.com/jp/resources/reports/rp-quarterly-threats-nov-2015.pdf⟩．（参照 2016-01-21）．

[5] 大和田徹，内山宏樹，伯田恵輔，森田伸義，萱島信：制御・車載システムにおけるセキュリティ機能要件に関する考察，2015 年 暗号と情報セキュリティシンポジウム (SCIS2015) 予稿集，CD-ROM，（2015）．

[6] デジタルフォレンジック研究会：デジタルフォレンジックとは，入手先 ⟨https://digitalforensic.jp/⟩，（参照 2016-8-2）．

[7] Ahmed, I., Obermeier, S., Naedele, M., and Richard III, G. G.：SCADA Systems: Challenges for Forensic Investigators，the IEEE Computer Society，（2012）．

[8] 田村研輔，松浦幹太：制御システムにおけるライブフォレンジックの適用可能性に関する実験的評価，2015 年 暗号と情報セキュリティシンポジウム (SCIS2015) 予稿集，CD-ROM，（2015）．

[9] 越智貴夫，小島孝夫，外川政夫，板倉征男：ホットデジタルフォレンジックによるインシデント検知方法の提案，情報処理学会研究報告，Vol.2008-CSEC-040，No.21，pp.267-272，（2008）．

[10] Taveras, P.：SCADA live forensics: real time data acquisition process to detect, prevent or evaluate critical situations，1st Annual International Interdisciplinary Conference，（2013）．

[11] Eden, P.，Blyth, A.，Burnap, P.，Jones, K.，and Stoddart, K.：A Forensic Taxonomy of SCADA Systems and Approach to Incident Response，3rd International Symposium for ICS & SCADA Cyber Security Research，（2015）．

[12] Google Code Archive：The Volatility Framework，入手先 ⟨https://code.google.com/archive/p/volatility/⟩，(accessed 2016-8-8)．

[13] Galloway, B.，and Hancke, G. P.：Introduction to Industrial Control Networks，IEEE COMMUNICATIONS SURVEYS & TUTORIALS，Vol.15，No.2，（2013）．