(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第6880017号
(P6880017)
(45)発行日　令和3年6月2日(2021.6.2)　(24)登録日　令和3年5月7日(2021.5.7)

(51)Int.Cl.　F I
*G09C 1/00 (2006.01)*　G09C 1/00 650Z

請求項の数 57　(全 34 頁)

(21)出願番号　特願2018-523361(P2018-523361)
(86)(22)出願日　平成28年7月20日(2016.7.20)
(65)公表番号　特表2018-522291(P2018-522291A)
(43)公表日　平成30年8月9日(2018.8.9)
(86)国際出願番号　PCT/US2016/043117
(87)国際公開番号　WO2017/015357
(87)国際公開日　平成29年1月26日(2017.1.26)
審査請求日　令和1年7月22日(2019.7.22)
(31)優先権主張番号　14/804,713
(32)優先日　平成27年7月21日(2015.7.21)
(33)優先権主張国・地域又は機関
米国(US)

(73)特許権者　518023980
バッフル　インコーポレイテッド
アメリカ合衆国　カリフォルニア州　９５０５４　サンタ　クララ　フリーダム　サークル　３９４５　スイート　４５０
(74)代理人　100094569
弁理士　田中　伸一郎
(74)代理人　100088694
弁理士　弟子丸　健
(74)代理人　100103610
弁理士　▲吉▼田　和彦
(74)代理人　100067013
弁理士　大塚　文昭
(74)代理人　100086771
弁理士　西島　孝喜

最終頁に続く

(54)【発明の名称】信頼できないコンピュータ上でプライベートプログラムを実行するためのシステム及びプロセス

(57)【特許請求の範囲】
【請求項１】
各ステップが特定のコンピュータによって実行されるコンピュータプログラムの実行方法であって、
少なくとも１つの他のコンピュータに接続された信頼できるコンピュータ上に存在する実行可能なコンピュータプログラムを一連の演算に分割するステップと、
前記信頼できるコンピュータ上の前記一連の演算の各演算を難読化するステップと、
実行可能なコンピュータプログラムのリモート実行のために、前記難読化した演算を付属命令と共に、前記信頼できるコンピュータから前記少なくとも１つの他のコンピュータに送信するステップであって、
前記付属命令は、
前記少なくとも１つの他のコンピュータによって、前記難読化された演算のそれぞれの結果を計算し、
前記少なくとも１つの他のコンピュータから、前記難読化された演算のそれぞれの結果を、難読化された演算を計算するために、前記難読化された演算のそれぞれの結果を必要とする前記少なくとも１つの他のコンピュータの別のコンピュータに転送し、
前記少なくとも１つの他のコンピュータによって、前記実行可能なコンピュータプログラムの難読化された計算結果として、最終的な前記難読化された演算のそれぞれの結果を前記信頼できるコンピュータに転送する、ステップと、
を含み、

前記方法は、前記信頼できるコンピュータにおいて、前記実行可能なコンピュータプログラムの前記難読化された計算結果を受信するステップと、
前記実行可能なコンピュータプログラムの受信した前記難読化された計算結果を逆難読化するステップと、
を含む演算を前記信頼できるコンピュータに行わせるステップをさらに含むことを特徴とする方法。
【請求項２】
前記一連の演算は回路ゲート演算であり、各回路ゲート演算は、演算子と、第１のオペランドと、第２のオペランドとを有する、
請求項１に記載の方法。
【請求項３】
前記一連の演算の各演算を難読化する前記ステップは、
前記第１のオペランドを第１の乱数値で難読化するステップと、
前記第２のオペランドを第２の乱数値で難読化するステップと、
を含む、請求項２に記載の方法。
【請求項４】
前記一連の演算の各演算を付属命令と共に少なくとも１つの他のコンピュータに送信する前記ステップは、
前記難読化されたオペランドを、
前記難読化されたオペランドを用いて複数の演算の複数の結果を計算し、
前記複数の結果を第２のコンピュータに送信する、
ことを含む命令と共に第１のコンピュータに送信するステップと、
前記演算子、前記第１の乱数値及び前記第２の乱数値に基づいて前記複数の結果から１つの結果を選択し、
前記選択された結果を第３の乱数値で難読化し、
前記選択された結果を異なるコンピュータに送信する、
ことを含む命令を前記第２のコンピュータに送信するステップと、
を含む、請求項３に記載の方法。
【請求項５】
前記一連の演算はコンピュータ算術演算であり、各コンピュータ算術演算は、演算子と、第１のオペランドと、第２のオペランドとを有する、
請求項１に記載の方法。
【請求項６】
前記一連の演算の各演算を送信前に難読化する前記ステップは、
前記第１のオペランドを第１の乱数値で難読化するステップと、
前記第２のオペランドを第２の乱数値で難読化するステップと、
を含む、請求項５に記載の方法。
【請求項７】
前記一連の演算の各演算を付属命令と共に少なくとも１つの他のコンピュータに送信する前記ステップは、
前記演算子及び難読化されたオペランドを、
前記演算子と、前記難読化された第１のオペランドと、前記難読化された第２のオペランドとを用いて第１の演算結果を計算し、
第２のコンピュータから値を受け取り、
前記演算子と、前記第１の結果と、前記値とを用いて第２の演算結果を計算し、
前記第２の結果を異なるコンピュータに送信する、
ことを含む命令と共に第１のコンピュータに送信するステップを含む、
請求項６に記載の方法。
【請求項８】
前記第１のオペランドの前記難読化スキームは、前記第２のオペランドの前記難読化ス

キームと一致しない、
請求項６に記載の方法。
【請求項９】
第３の乱数値を用いて前記第１のオペランドの前記難読化スキームを前記第２のオペランドの前記難読化スキームに変換する遷移難読化関数が使用される、
請求項８に記載の方法。
【請求項１０】
前記遷移難読化関数を一連の演算に分割するステップと、
前記遷移難読化関数の各演算を、前記それぞれの演算の結果を計算して該結果を別のコンピュータに転送する付属命令と共に少なくとも１つの他のコンピュータに送信するステップと、
をさらに含む、請求項９に記載の方法。
【請求項１１】
前記遷移難読化関数の各演算を付属命令と共に少なくとも１つの他のコンピュータに送信する前記ステップは、
前記第１のオペランドが乗法難読化される場合、
前記難読化されたオペランドを、
第２のコンピュータから値を受け取り、
前記難読化された第１のオペランドと前記値との和を計算し、
前記和を第３のコンピュータに送信する、
ことを含む命令と共に第１のコンピュータに送信するステップと、
前記和と前記第１の乱数値との商を計算し、
前記商を異なるコンピュータに送信する、
ことを含む命令を前記第３のコンピュータに送信するステップと、
を含み、前記第１のオペランドが加法難読化される場合、
前記難読化されたオペランドを、
前記難読化された第１のオペランドと前記第３の乱数値との積を計算し、
前記積を第１のコンピュータに送信する、
ことを含む命令と共に第３のコンピュータに送信するステップと、
第２のコンピュータから値を受け取り、
前記積と前記値との間の差分を計算し、
前記差分を異なるコンピュータに送信する、
ことを含む命令を前記第１のコンピュータに送信するステップと、
を含む、請求項１０に記載の方法。
【請求項１２】
前記第２のコンピュータから受け取られる前記値は、前記第１の乱数値と前記第３の乱数値との積である、
請求項１１に記載の方法。
【請求項１３】
前記難読化は、コード難読化、データ難読化、又はこれらの両方を含む、
請求項１に記載の方法。
【請求項１４】
前記少なくとも１つの他のコンピュータはクラウドの一部である、
請求項１に記載の方法。
【請求項１５】
前記クラウドは信頼できない、
請求項１４に記載の方法。
【請求項１６】
前記クラウドは、複数の管理領域にわたる、
請求項１４に記載の方法。

【請求項１７】
前記クラウドは、複数の商業的に異なるインフラにわたる、
請求項１４に記載の方法。
【請求項１８】
前記少なくとも１つの他のコンピュータは、企業ネットワークの一部である、
請求項１に記載の方法。
【請求項１９】
前記少なくとも１つの他のコンピュータは、複数のコンピュータからランダムに選択される、
請求項１に記載の方法。
【請求項２０】
前記複数のコンピュータの各コンピュータは信頼できない、
請求項１９に記載の方法。
【請求項２１】
前記少なくとも１つの他のコンピュータは信頼できない、
請求項１に記載の方法。
【請求項２２】
前記少なくとも１つの他のコンピュータは、前記信頼できるコンピュータではない、
請求項１に記載の方法。
【請求項２３】
システムであって、少なくとも１つの他のコンピュータに通信可能に接続された少なくとも１つの信頼できるコンピュータを備え、前記信頼できるコンピュータは、実行時に前記システムに、
第１の信頼できるコンピュータ上に存在する実行可能なコンピュータプログラムを一連の演算に分割するステップと、
前記一連の演算の各演算を難読化するステップと、
前記実行可能なコンピュータプログラムのリモート実行のために、前記信頼できるコンピュータから、前記難読化した演算を付属命令と共に、前記少なくとも１つの他のコンピュータに送信するステップであって、
前記付属命令は、
前記少なくとも１つの他のコンピュータによって、前記難読化された演算のそれぞれの結果を計算し、
前記少なくとも１つの他のコンピュータから、前記難読化された演算のそれぞれの結果を、難読化された演算を計算するために、前記難読化された演算のそれぞれの結果を必要とする前記少なくとも１つの他のコンピュータの別のコンピュータに転送し、
前記少なくとも１つの他のコンピュータによって、前記実行可能なコンピュータプログラムの難読化された計算結果として、最終的な前記難読化された演算のそれぞれの結果を前記第１の信頼できるコンピュータに転送する、ステップと、
を含む動作を実行させるコンピュータ命令を記憶し、
前記命令は、さらに前記システムに備えられた前記第１の信頼できるコンピュータに、
前記実行可能なコンピュータプログラムの前記難読化された計算結果を受け取るステップと、
を含む動作を実行させる、
ことを特徴とするシステム。
【請求項２４】
前記一連の演算は回路ゲート演算であり、各回路ゲート演算は、演算子と、第１のオペランドと、第２のオペランドとを有する、
請求項２３に記載のシステム。
【請求項２５】
前記一連の演算の各演算を難読化する前記ステップは、

前記第１のオペランドを第１の乱数値で難読化するステップと、
前記第２のオペランドを第２の乱数値で難読化するステップと、
を含む、請求項２４に記載のシステム。
【請求項２６】
前記一連の演算の各演算を付属命令と共に少なくとも１つの他のコンピュータに送信する前記ステップは、
前記難読化されたオペランドを、
前記難読化されたオペランドを用いて複数の演算の複数の結果を計算し、
前記複数の結果を第２のコンピュータに送信する、
ことを含む命令と共に第１のコンピュータに送信するステップと、
前記演算子、前記第１の乱数値及び前記第２の乱数値に基づいて前記複数の結果から１つの結果を選択し、
前記選択された結果を第３の乱数値で難読化し、
前記選択された結果を異なるコンピュータに送信する、
ことを含む命令を前記第２のコンピュータに送信するステップと、
を含む、請求項２５に記載のシステム。
【請求項２７】
前記一連の演算はコンピュータ算術演算であり、各コンピュータ算術演算は、演算子と、第１のオペランドと、第２のオペランドとを有する、
請求項２３に記載のシステム。
【請求項２８】
前記一連の演算の各演算は、ハードウェアユニットによって実行される演算に対応する、
請求項２３に記載のシステム。
【請求項２９】
前記少なくとも１つの他のコンピュータ上の整数加算器をさらに備える、
請求項２８に記載のシステム。
【請求項３０】
前記少なくとも１つの他のコンピュータ上の整数乗算器をさらに備える、
請求項２８に記載のシステム。
【請求項３１】
前記少なくとも１つの他のコンピュータ上の整数比較器をさらに備える、
請求項２８に記載のシステム。
【請求項３２】
前記少なくとも１つの他のコンピュータ上の浮動小数点乗算器をさらに備える、
請求項２８に記載のシステム。
【請求項３３】
前記一連の演算の各演算を送信前に難読化する前記ステップは、
前記第１のオペランドを第１の乱数値で難読化するステップと、
前記第２のオペランドを第２の乱数値で難読化するステップと、
を含む、請求項２７に記載のシステム。
【請求項３４】
前記一連の演算の各演算を付属命令と共に少なくとも１つの他のコンピュータに送信する前記ステップは、
前記演算子及び難読化されたオペランドを、
前記演算子と、前記難読化された第１のオペランドと、前記難読化された第２のオペランドとを用いて第１の演算結果を計算し、
第２のコンピュータから値を受け取り、
前記演算子と、前記第１の結果と、前記値とを用いて第２の演算結果を計算し、
前記第２の結果を異なるコンピュータに送信する、

ことを含む命令と共に第１のコンピュータに送信するステップを含む、
請求項３３に記載のシステム。
【請求項３５】
前記第１のオペランドの前記難読化スキームは、前記第２のオペランドの前記難読化スキームと一致しない、
請求項３３に記載のシステム。
【請求項３６】
第３の乱数値を用いて前記第１のオペランドの前記難読化スキームを前記第２のオペランドの前記難読化スキームに変換する遷移難読化関数が使用される、
請求項３５に記載のシステム。
【請求項３７】
前記少なくとも１つの信頼できるコンピュータは、実行時に前記システムに、
前記遷移難読化関数を一連の演算に分割するステップと、
前記遷移難読化関数の各演算を、前記それぞれの演算の結果を計算して該結果を別のコンピュータに転送する付属命令と共に少なくとも１つの他のコンピュータに送信するステップと、
をさらに含む動作を実行させるコンピュータ命令を記憶する、
請求項３６に記載のシステム。
【請求項３８】
前記遷移難読化関数の各演算を付属命令と共に少なくとも１つの他のコンピュータに送信する前記ステップは、
前記第１のオペランドが乗法難読化される場合、
前記難読化されたオペランドを、
第２のコンピュータから値を受け取り、
前記難読化された第１のオペランドと前記値との和を計算し、
前記和を第３のコンピュータに送信する、
ことを含む命令と共に第１のコンピュータに送信するステップと、
前記和と前記第１の乱数値との商を計算し、
前記商を異なるコンピュータに送信する、
ことを含む命令を前記第３のコンピュータに送信するステップと、
を含み、前記第１のオペランドが加法難読化される場合、
前記難読化されたオペランドを、
前記難読化された第１のオペランドと前記第３の乱数値との積を計算し、
前記積を第１のコンピュータに送信する、
ことを含む命令と共に第３のコンピュータに送信するステップと、
第２のコンピュータから値を受け取り、
前記積と前記値との間の差分を計算し、
前記差分を異なるコンピュータに送信する、
ことを含む命令を前記第１のコンピュータに送信するステップと、
を含む、請求項３７に記載のシステム。
【請求項３９】
前記第２のコンピュータから受け取られる前記値は、前記第１の乱数値と前記第３の乱数値との積である、
請求項３８に記載のシステム。
【請求項４０】
前記難読化は、コード難読化、データ難読化、又はこれらの両方を含む、
請求項２３に記載のシステム。
【請求項４１】
前記少なくとも１つの他のコンピュータはクラウドの一部である、
請求項２３に記載のシステム。

【請求項４２】
前記クラウドは信頼できない、
請求項４１に記載のシステム。
【請求項４３】
前記クラウドは、複数の管理領域にわたる、
請求項４１に記載のシステム。
【請求項４４】
前記クラウドは、複数の商業的に異なるインフラにわたる、
請求項４１に記載のシステム。
【請求項４５】
前記少なくとも１つの他のコンピュータは、企業ネットワークの一部である、
請求項２３に記載のシステム。
【請求項４６】
前記少なくとも１つの他のコンピュータは、複数のコンピュータからランダムに選択される、
請求項２３に記載のシステム。
【請求項４７】
前記複数のコンピュータの各コンピュータは信頼できない、
請求項４６に記載のシステム。
【請求項４８】
前記少なくとも１つの他のコンピュータは信頼できない、
請求項２３に記載のシステム。
【請求項４９】
前記少なくとも１つの他のコンピュータは、前記信頼できるコンピュータではない、
請求項２３に記載のシステム。
【請求項５０】
前記信頼できるコンピュータは、１又は２以上の仮想機械を含む、
請求項１に記載の方法。
【請求項５１】
前記少なくとも１つの他のコンピュータは、１又は２以上の仮想機械を含む、
請求項１に記載の方法。
【請求項５２】
前記別のコンピュータは、１又は２以上の仮想機械を含む、
請求項１に記載の方法。
【請求項５３】
前記信頼できるコンピュータは、１又は２以上の仮想機械を含む、
請求項２３に記載のシステム。
【請求項５４】
前記少なくとも１つの他のコンピュータは、１又は２以上の仮想機械を含む、
請求項２３に記載のシステム。
【請求項５５】
前記別のコンピュータは、１又は２以上の仮想機械を含む、
請求項２３に記載のシステム。
【請求項５６】
前記一連の演算の各演算を付属命令と共に前記少なくとも１つの他のコンピュータに送信する前記ステップは、各演算を計算のために複数のコンピュータに送信するステップを含む、
請求項１に記載の方法。
【請求項５７】
前記一連の演算の各演算を付属命令と共に前記少なくとも１つの他のコンピュータに送

信する前記ステップは、各演算を計算のために複数のコンピュータに送信するステップを含む、
請求項２３に記載のシステム。
【発明の詳細な説明】
【技術分野】
【０００１】
〔関連出願との相互参照〕
本出願は、２０１５年７月２１日に出願された米国特許出願第１４／８０４，７１３号に対する優先権を主張するものであり、この文献の開示は全体が引用により本明細書に組み入れられる。
【０００２】
開示する発明は、コンピュータセキュリティの分野に関する。
【背景技術】
【０００３】
最近では、特にユーザによる大規模コンピュータネットワークへの自由なアクセスを可能にするクラウドネットワーク及び拡張型企業ネットワークの急増に伴い、リモートコンピュータプログラムを実行する際のセキュリティが継続的に論争されている。複数のコンピュータ上でプログラムを実行する能力と速度の組み合わせは有利ではあるが、詮索の目に曝されることもある。リモートコンピュータの一部又は全部が、悪意あるユーザ又は攻撃者の支配を受けて重要な機密情報を危険に曝してしまう可能性もある。
【発明の概要】
【発明が解決しようとする課題】
【０００４】
従って、信頼できないコンピュータ上又は複数のコンピュータ上で、（単複の）信頼できないコンピュータに完全にアクセスできる攻撃者から実行プログラム及びデータを秘密にしておくようにプログラムを実行する方法及びシステムに対するニーズが存在する。本発明は、これらの及びその他の重要なニーズに対処するものである。
【課題を解決するための手段】
【０００５】
本発明は、コンピュータプログラムの実行方法であって、少なくとも１つの他のコンピュータに接続された信頼できるコンピュータ上に存在するコンピュータプログラムを一連の演算に分割するステップと、一連の演算の各演算を、それぞれの演算の結果を計算して結果を別のコンピュータに転送する付属命令と共に少なくとも１つの他のコンピュータに送信するステップと、信頼できるコンピュータにおいて、コンピュータプログラムの計算結果を受け取るステップとを含む方法を提供する。
【０００６】
本発明は、コンピュータプログラムを実行するシステムも提供する。このシステムは、少なくとも１つの他のコンピュータに通信可能に接続された少なくとも１つの信頼できるコンピュータを含むことができる。少なくとも１つの信頼できるコンピュータのうちの第１の信頼できるコンピュータは、実行時にシステムにプログラム実行プロセスを含む動作を実行させるコンピュータ命令を記憶することができる。このプロセスは、第１の信頼できるコンピュータ上のコンピュータプログラムを一連の演算に分割するステップと、一連の演算の各演算を、それぞれの演算の結果を計算して結果を別のコンピュータに転送する付属命令と共に少なくとも１つの他のコンピュータに送信するステップと、信頼できるコンピュータにおいて、コンピュータプログラムの計算結果を受け取るステップとを含むことができる。
【０００７】
この概要及び以下の詳細な説明は例示的かつ説明的なものにすぎず、添付の特許請求の範囲に定める本発明を限定するものではない。当業者には、本明細書に示す本発明の詳細な説明に照らして本発明の他の態様も明らかになるであろう。

【０００８】
この概要及び以下の詳細な説明は、添付図面と共に読むことによってさらに理解される。図面には、本発明を説明する目的で本発明の例示的な実施形態を示すが、本発明は、開示する具体的な方法、構成及び装置に限定されるものではない。また、図面は必ずしも縮尺通りではない。
【図面の簡単な説明】
【０００９】
【図１】ネットワークコンピュータにわたってプログラムをシュレッディングするための本発明の実施形態を示す図である。
【図２】分割デバイスドライバ動作のための本発明の実施形態を示す図である。
【図３】コード及びデータのシュレッドの対称鍵暗号化及び暗号解読動作のための本発明の実施形態を示す図である。
【図４】シュレッドを難読化するための本発明の実施形態を示す図である。
【図５】回路ゲート演算をシュレッディングするための本発明の実施形態を示す図である。
【図６】別の回路ゲート演算をシュレッディングするための本発明の実施形態を示す図である。
【図７】さらなる難読化レイヤを用いて回路ゲート演算をシュレッディングするための本発明の実施形態を示す図である。
【図８】コンピュータネットワーク上のゲート演算の任意の回路をシュレッディングするための本発明の実施形態を示す図である。
【図９】さらなる暗号化を用いて演算をシュレッディングするための本発明の実施形態を示す図である。
【図１０】通常のデバイスドライバ動作を示す図である。
【図１１】分割デバイスドライバ動作のための本発明の実施形態を示す図である。
【図１２】数学演算をシュレッディングするための本発明の実施形態を示す図である。
【図１３】別の数学演算をシュレッディングするための本発明の実施形態を示す図である。
【図１４】遷移暗号関数をシュレッディングするための本発明の実施形態を示す図である。
【図１５】別の遷移暗号関数をシュレッディングするための本発明の実施形態を示す図である。
【図１６】暗号化された比較演算をシュレッディングするための本発明の実施形態を示す図である。
【図１７】別の暗号化された比較演算をシュレッディングするための本発明の実施形態を示す図である。
【図１８】数学的難読化遷移関数をシュレッディングするための本発明の実施形態を示す図である。
【図１９】別の数学的難読化遷移関数をシュレッディングするための本発明の実施形態を示す図である。
【発明を実施するための形態】
【００１０】
本発明は、本開示の一部を成す添付図及び実施例に関連して行う以下の詳細な説明を参照することによってさらに容易に理解することができる。本発明は、本明細書において説明及び／又は図示する具体的な装置、方法、用途、条件又はパラメータに限定されるものではなく、本明細書で使用する用語は、特定の実施形態をほんの一例として説明するためのものであり、特許請求する発明を限定するためのものではないと理解されたい。また、文脈において別途明確に示していない限り、添付の特許請求の範囲を含め、本明細書で使用する「１つの（英文不定冠詞）」及び「その（英文定冠詞）」という単数形は複数形を含み、特定の数値への言及は、少なくともその特定の値を含む。本明細書で使用する「複

数」という用語は、１つよりも多くを意味する。値の範囲を示していない場合、別の実施形態は、１つの特定の値から、及び／又は他の特定の値までを含む。同様に、「約（ａｂｏｕｔ）」という先行詞を使用することによって値を近似値として示す場合には、特定の値が別の実施形態を形成すると理解されるであろう。全ての範囲は包含的であって組み合わせることができる。

【００１１】

明確にするために本明細書において個別の実施形態の文脈で説明している本発明のいくつかの特徴は、単一の実施形態において組み合わせて提供することもできると理解されたい。これとは逆に、明確にするために単一の実施形態の文脈で説明している本発明の様々な特徴は、個別に又はいずれかの部分的組み合わせの形で提供することもできる。さらに、範囲で示す値についての言及は、その範囲内のありとあらゆる値を含む。

【００１２】

本明細書で開示する解決策は、少なくとも１つの他のコンピュータにネットワーク接続された信頼できるコンピュータ上でコンピュータプログラムを実行する方法と、この方法を実行できるシステムとを含む。この方法は、コンピュータプログラムを一連の演算に分割するステップと、一連の演算の各演算を、演算を計算した後に演算結果を別のコンピュータに転送するように演算を受け取ったそれぞれのコンピュータに命じる付属命令と共に少なくとも１つの他のコンピュータに送信するステップと、信頼できるコンピュータにおいて、コンピュータプログラムの出力を受け取るステップとを含む。暗号化及び難読化を用いて方法のセキュリティを高めることもできる。

【００１３】

信頼できるコンピュータとは、攻撃者によって危殆化されていないと分かっているコンピュータのことであり、計算リソース又はその他のリソースを提供するために少なくとも１つの他のコンピュータを必要とすることができる。非限定的な例では、この少なくとも１つの他のコンピュータを、信頼できるコンピュータに通信可能に接続されているが信頼できるコンピュータを含まない、信頼できる又は信頼できない、企業ネットワークの一部、クラウドの一部、モバイル装置、一般に複数のコンピュータの一部、或いは他のいずれかの１つ又は一連のコンピュータとすることができる。さらなる非限定的な例では、少なくとも１つの他のコンピュータを複数のコンピュータからランダムに選択することができる。「クラウド」に関して言えば、クラウドは、信頼できる場合も、又は信頼できない場合もあり、複数の管理領域にわたることもあり、複数の商業的に異なるインフラにわたることもあり、或いはいずれかの組み合わせ又は当業で周知の他の状況の場合もある。コンピュータは、（電気的又は光学的な）有線接続、無線接続、及び／又はコンピュータによるデータの通信を可能にするいずれかのタイプの接続を用いて共にネットワーク化することができる。コンピュータは、１つの又は一群の仮想マシンを含むこともできる。

【００１４】

コンピュータプログラムを一連の演算に分割し、一連の演算の各演算を付属命令と共に少なくとも１つの他のコンピュータに送信することを「シュレッディング（ｓｈｒｅｄｄｉｎｇ）」と呼ぶことができ、各演算は「シュレッド（ｓｈｒｅｄ）」と呼ばれる。シュレッディングプロセスでは、プログラムが一群のシュレッドに変換され、各シュレッドが少なくとも１つのコンピュータ上でプログラムの一部を実行して他のシュレッドと通信し、シュレッドの集合体が完全なプログラムを実行するようになる。ある実施形態では、シュレッディングプロセスが、コンピュータのランダム選択を用いてネットワークコンピュータ上にシュレッドを配置し、各コンピュータが、そのシュレッド内に指定される部分的計算を実行した後に残りの計算を次のコンピュータに転送する。各コンピュータは、シュレッドをどこから受け取ったか、シュレッド内で何を計算することになっているか、及びそのシュレッドの結果をどこに転送すべきかしか分からない。ネットワークコンピュータの数が十分に多ければ、潜在的な攻撃者は、全てのコンピュータを同時にモニタしてシュレッディング計算をつなぎ合わせることができない。

【００１５】

図１に、シュレッディングプロセスの実施形態を用いて信頼できるコンピュータ及びクラウドコンピュータ上で実行されるプログラム例を示す。プログラム１００は、入力、関数「ｆ」、関数「ｇ」及び出力という４つの部分を含む。信頼できるコンピュータ上で実行されるシュレッダプログラムは、このプログラムのために実行できる２値を分析してこれらの４つの部分を発見し、これに応じてやはり２進実行可能形式の４つのシュレッドを生成する。入力演算及び出力演算は、信頼できるコンピュータ１（ＴＣ１）１１０及び信頼できるコンピュータ４（ＴＣ４）１４０上でそれぞれ実行される。ＴＣ１　１１０及びＴＣ４　１４０は、必要な入力装置及び出力装置に接続されているという理由でシュレッダによって選択される。関数「ｆ」及び「ｇ」は計算コストが高く、従ってクラウドコンピュータ２（ＣＣ２）１２０及びクラウドコンピュータ３（ＣＣ３）１３０上でそれぞれ実行される。ＣＣ２　１２０及びＣＣ３　１３０は、利用可能な機械のプールからシュレッダによってランダムに選択される。ＴＣ１　１１０は、入力を受け取り、この入力を変数「ｘ」に割り当てた後に、これをＣＣ２　１２０に転送する。ＣＣ２　１２０は、変数「ｘ」を受け取って関数「ｆ（ｘ）」を計算し、結果を変数「ｙ」に割り当てた後に、これをＣＣ３　１３０に転送する。ＣＣ３　１３０は、変数「ｙ」を受け取って関数「ｇ（ｘ）」を計算し、結果を変数「ｚ」に割り当てた後に、これをＴＣ４　１４０に転送する。ＴＣ４　１４０は、変数「ｚ」を受け取って出力する。

【００１６】

図１で分かるように、一連の演算の一部は、信頼できるコンピュータからの入力及び／又は出力（Ｉ／Ｏ）相互作用を必要とし得る。Ｉ／Ｏ装置又は信頼できるＩ／Ｏが存在しないことを考慮すると（セキュリティのために、Ｉ／Ｏ装置からの平文データを見られるのは信頼できるコンピュータのみとすべきである）、シュレッドを実行するコンピュータ上には特別な「分割」デバイスドライバを実装することができる。分割デバイスドライバは、実際には各コンピュータ上に１つずつの２つのドライバであり、各ドライバが通常のデバイスドライバの半分の作業を実行する。これらのドライバの両半分はユーザプロセスとして実行されるので、シュレッドを実行するコンピュータ上のオペレーティングシステムも、信頼できるコンピュータ上のオペレーティングシステムも変更する必要がない。以下は、分割デバイスドライバをどのように実装して使用できるかを示す実施形態例である。

【００１７】

ある実施形態では、特定のコンピュータ上における演算の、すなわちシュレッドの結果の計算中に、この演算が、第１の信頼できるコンピュータに接続された装置からの入力を必要とする場合、特定のコンピュータは、特定のコンピュータ上のプログラムドライバにおいて入力要求を生成し、この入力要求をプログラムドライバからネットワークドライバに受け渡し、ネットワークドライバを介してこの入力要求を第１の信頼できるコンピュータに送信し、ネットワークドライバにおいて第１の信頼できるコンピュータから装置の応答を受け取り、この応答を、計算において使用されるようにネットワークドライバからプログラムドライバに受け渡すことができる。受け取られる応答は、セキュリティを高めるために暗号化することができる。

【００１８】

ある実施形態では、特定のコンピュータ上における演算の、すなわちシュレッドの結果の計算中に、この演算が、第１の信頼できるコンピュータに接続された装置からの入力を必要とする場合、第１の信頼できるコンピュータは、第１の信頼できるコンピュータ上のネットワークドライバにおいて、装置に入力を求める要求を特定のコンピュータから受け取り、この要求をネットワークドライバから装置の入力ドライバに受け渡し、装置からの応答を入力ドライバからネットワークドライバに受け渡し、ネットワークドライバを介して応答を特定のコンピュータに送信することができる。第１の信頼できるコンピュータは、セキュリティを高めるために、装置からの応答を入力ドライバからネットワークドライバに受け渡す前に応答を暗号化することができる。

【００１９】

ある実施形態では、特定のコンピュータ上における演算の結果の計算中に、この演算が、第１の信頼できるコンピュータに接続された装置からの出力を必要とする場合、特定のコンピュータは、特定のコンピュータ上のプログラムドライバにおいて出力要求を生成し、この出力要求をプログラムドライバからネットワークドライバに受け渡し、ネットワークドライバを介して出力要求を第１の信頼できるコンピュータに送信し、ネットワークドライバにおいて第１の信頼できるコンピュータから装置の状態を受け取り、この状態を、計算において使用されるようにネットワークドライバからプログラムドライバに受け渡すことができる。出力要求は、暗号化データを含むことができる。

【００２０】

ある実施形態では、特定のコンピュータ上における演算の結果の計算中に、この演算が、第１の信頼できるコンピュータに接続された装置からの出力を必要とする場合、第１の信頼できるコンピュータは、第１の信頼できるコンピュータ上のネットワークドライバにおいて、この装置への出力要求を特定のコンピュータから受け取り、この要求をネットワークドライバから装置の出力ドライバに受け渡し、装置からの状態を出力ドライバからネットワークドライバに受け渡し、ネットワークドライバを介して状態を特定のコンピュータに送信することができる。要求は、暗号化データを含むことができ、第１の信頼できるコンピュータは、要求をネットワークドライバから出力ドライバに受け渡す前に、データを暗号解読する必要があり得る。

【００２１】

ある実施形態では、特定のコンピュータ上における演算の結果の計算中に、この演算が、第１の信頼できるコンピュータに接続された装置からの入力を必要とする場合、この入力を収集する方法は、特定のコンピュータ上のプログラムドライバにおいて入力要求を生成するステップと、この入力要求をプログラムドライバから第１のネットワークドライバに受け渡すステップと、第１のネットワークドライバを介して入力要求を第１の信頼できるコンピュータに送信するステップと、第１の信頼できるコンピュータ上の第２のネットワークドライバにおいて入力要求を特定のコンピュータから受け取るステップと、入力要求を第２のネットワークドライバから装置の入力ドライバに受け渡すステップと、装置からの応答を入力ドライバから第２のネットワークドライバに受け渡すステップと、第２のネットワークドライバを介してこの応答を特定のコンピュータに送信するステップと、第１のネットワークドライバにおいて第１の信頼できるコンピュータから装置の応答を受け取るステップと、この応答を、計算において使用されるように第１のネットワークドライバからプログラムドライバに受け渡すステップとを含むことができる。この方法は、セキュリティを高めるために、装置からの応答を入力ドライバから第２のネットワークドライバに受け渡す前に応答を暗号化するステップをさらに含むことができる。従って、第１の信頼できるコンピュータからの受け取られる応答は暗号化することができる。

【００２２】

ある実施形態では、特定のコンピュータ上における演算の結果の計算中に、この演算が、第１の信頼できるコンピュータに接続された装置からの出力を必要とする場合、この出力を実行する方法は、特定のコンピュータ上のプログラムドライバにおいて出力要求を生成するステップと、この出力要求をプログラムドライバから第１のネットワークドライバに受け渡すステップと、第１のネットワークドライバを介して出力要求を第１の信頼できるコンピュータに送信するステップと、第１の信頼できるコンピュータ上の第２のネットワークドライバにおいて出力要求を特定のコンピュータから受け取るステップと、出力要求を第２のネットワークドライバから装置の出力ドライバに受け渡すステップと、装置からの状態を出力ドライバから第２のネットワークドライバに受け渡すステップと、第２のネットワークドライバを介してこの状態を特定のコンピュータに送信するステップと、第１のネットワークドライバにおいて第１の信頼できるコンピュータから状態を受け取るステップと、この状態を、計算において使用されるように第１のネットワークドライバからプログラムドライバに受け渡すステップとを含むことができる。セキュリティを高めるために、出力要求は暗号化データを含むことができ、出力要求が第２のネットワークドライ

バから出力ドライバに受け渡される前にこのデータを暗号解読する必要がある。

【００２３】

セキュリティを高めるために、暗号化は複数のレベルで使用することができる。信頼できるコンピュータと少なくとも１つの他のコンピュータとの間のネットワーク接続は、全て暗号化することができる。このようなネットワーク暗号化は、例えばトランスポートレイヤセキュリティ（ＴＬＳ）又はその他のいずれかの好適な暗号化スキームを用いて実装することができる。

【００２４】

図２に、明示してはいないがネットワーク通信にＴＬＳ暗号化を伴うことができる分割デバイスドライバの実装及び動作の高水準な実施形態を示す。クラウドコンピュータ１（ＣＣ１）２００上の分割デバイスドライバは、プログラムと相互作用するドライバと、ネットワークと相互作用するドライバという２つの部分を含む。同様に、信頼できるコンピュータ１（ＴＣ１）２１０上の分割デバイスドライバは、ネットワークと相互作用するドライバであるネットワークインターフェイス１　２１２と、入力装置と相互作用するドライバである入力デバイスドライバ１　２１４という２つの部分を含む。ＣＣ１　２００は、分割デバイスドライバスキームを活用して「ｘ＝ｉｎ（）」入力演算を実行する。この入力演算では、プログラムと相互作用するドライバが入力要求２０２を生成する。この入力要求２０２は、ネットワークと相互作用するドライバに受け渡され、ＴＬＳ暗号化されてＴＣ１　２１０に送信される。ネットワークインターフェイス１　２１２は、この要求を受け取って入力デバイスドライバ１　２１４に受け渡す。入力装置である入力装置１　２２０からの応答が受け取られると、この応答が入力デバイスドライバ１　２１４によってネットワークインターフェイス１　２１２に受け渡される。ネットワークインターフェイス１　２１２は、ＴＬＳを用いてこの応答を暗号化し、ＣＣ１　２００に応答２２２を送信する。ＣＣ１　２００上のネットワークドライバは、ＴＬＳ暗号解読された応答２２２をプログラムインターフェイスドライバに転送し、プログラムインターフェイスドライバは、待機中のプログラムにこれをさらに転送する。プログラムは、受け取った入力を変数「ｘ」に割り当てて実行を継続する。図２には、クラウドコンピュータ４（ＣＣ４）２３０と信頼できるコンピュータ４（ＴＣ４）２４０との間で分割デバイスドライバを用いてどのように出力を実行できるかも示す。分割デバイスドライバの出力動作は、コンピュータが装置からの入力を受け取る代わりに状態メッセージ２５２を受け取ることを除いて入力動作と同様である。

【００２５】

一連の演算の各演算、すなわち各シュレッドは暗号化することができる。各シュレッドは、高度暗号化標準（ＡＥＳ）ガロアカウンタモード（ＧＣＭ）又は他のいずれかの好適なスキームなどの対称鍵暗号化スキームを用いて暗号化することができる。対称鍵暗号化は、各シュレッドが異なる鍵を用いて暗号化されるように、少なくとも１つの他のコンピュータの（単複の）共有鍵を使用する。このようなスキームは、シュレッドの対象であるコンピュータ上でしかシュレッドの暗号解読を可能にせず、残りのプログラムを他のあらゆるコンピュータから隠す。

【００２６】

一連の演算の各演算の各データ値は、暗号化することができる。データ値の暗号化は、ＡＥＳ　ＧＣＭなどの対称鍵スキーム、ＲＳＡなどの公開鍵スキーム、又は他のいずれかの好適なスキームを用いて行うことができる。このような暗号化は、信頼できるコンピュータ及び少なくとも１つの他のコンピュータの両方において実行することができる。対称鍵スキームを使用する場合、使用される秘密鍵は、データの送信側とデータの受信側との間のリンクの共有鍵である。公開鍵スキームを使用する場合、送信側は受信側の公開鍵を用いて暗号化し、受信側は受信側の秘密鍵を用いて暗号解読する。このようなスキームは、少なくとも１つの他のコンピュータが計算に必要なデータを暗号解読できるようにするが、他の全てのデータをこのコンピュータから隠す。

【００２７】

図３に、図１のプログラム例におけるコード及びデータのシュレッドの対称鍵暗号化及び暗号解読動作の実施形態を示す。図３においても、図２と同様に、通信パケットのＴＬＳ暗号化及び暗号解読は明示していない。プログラムの計算に関与する４つのコンピュータである信頼できるコンピュータ１（ＴＣ１）３１０、クラウドコンピュータ２（ＣＣ２）３２０、クラウドコンピュータ３（ＣＣ３）３３０及び信頼できるコンピュータ４（ＴＣ４）３４０は、信頼できるコンピュータ０（ＴＣ０）３００に知られている秘密鍵Ｋ１、Ｋ２、Ｋ３及びＫ４をそれぞれ有する。シュレッディングされたプログラムコードは、シュレッディング動作中にＴＣ０　３００上で共有鍵Ｋ１、Ｋ２、Ｋ３及びＫ４を用いて暗号化される。例えば、ＴＣ０　３００上の「ｉｎ’＝Ｅｎｃ（Ｋ１，ｉｎ）」という文は、鍵Ｋ１を用いて「ｉｎ」のコードを暗号化して暗号化コード「ｉｎ’」を生成することを意味する。ＴＣ１　３１０上の対応する暗号解読演算は「ｉｎ＝Ｄｅｃ（Ｋ１，ｉｎ’）」であり、この暗号解読によって「ｉｎ」演算のコードが生成される。ＴＣ１　３１０は、鍵Ｋ１しか知らないので、暗号化プログラムの他の部分を暗号解読することはできず、鍵Ｋ１を用いて他の何らかのコード部分を暗号解読しようと試みても有効なコードは生成されない。この暗号化及び暗号解読プロセスは、シュレッドを送信した各コンピュータの鍵を計算に用いて各コンピュータが計算対象のシュレッドにしかアクセスできないようにして、全てのシュレッドについて繰り返される。コンピュータ間の通信リンクは、データの送信側及び受信側しか知らない共有鍵も有する。この場合、通信リンクの共有鍵は、Ｋ１２、Ｋ２３及びＫ３４である。例えば、ＴＣ１　３１０は、共有鍵Ｋ１２を用いて入力「ｘ」を暗号文「ｘ’」に暗号化し、ＣＣ２　３２０において鍵Ｋ１２を用いてｘ’を暗号解読して「ｘ」を得る。このプロセスを全てのコンピュータにわたって使用して、攻撃者がネットワークを介して平文を傍受できないことを確実にする。

【００２８】

さらなるセキュリティのためには、データ難読化、コード難読化、又はこれらの両方を含むことができる難読化を使用すべきである。難読化は、各シュレッド内のコード及び／又はデータを、他のコンピュータにおける攻撃者からオリジナルのコード及びデータを隠すように修正するものである。コード難読化は、変換の難読化を用いて、シュレッド実行中にシュレッドのコード、並びに命令及びデータを完全に視認できる攻撃者からオリジナルプログラムのロジックを隠す。データ難読化は、攻撃者が逆難読化されたデータ値を復元するのが困難になるように、シュレッディングされたコードが難読化されたデータ値に対して実行されるようにデータを変換するものであり、ブラインディングはデータ難読化の一種である。

【００２９】

シュレッディングを伴う難読化では、一連の演算の各演算を他のそれぞれのコンピュータに送信する前に各演算を難読化し、受け取られたコンピュータプログラムの計算結果を逆難読化する。難読化方法は、シュレッディングレベルによって異なり、シュレッディングは、ゲートレベル、ハードウェアユニットレベル、命令レベル、暗号化スキームレベルという少なくとも４つの異なるレベル、及びプログラムをユニットに分割できるようにする他のいずれかのスキームに基づいて行うことができる。

【００３０】

図４に、プログラム「ｚ＝ａ＊ｘ＋ｙ」を４つのコンピュータにわたって難読化する例を示す。難読化の目的は、クラウドコンピュータ２（ＣＣ２）４１０及びクラウドコンピュータ３（ＣＣ３）４２０における攻撃者から、入力「ａ」、「ｘ」及び「ｙ」、並びに出力「ｚ」の値を隠すことである。信頼できるコンピュータ１（ＴＣ１）４００は、入力「ａ」、「ｘ」及び「ｙ」の値を難読化するためのワンタイムパッドとしてそれぞれ使用される３つの乱数値「ｒ」、「ｓ」及び「ｔ」を導入する。難読化に使用される演算は、変数の使用に依存する。例えば、変数「ａ」と「ｘ」は乗算されるので、難読化演算は、「ａ」及び「ｘ」にこれらのパッド値「ｒ」及び「ｓ」を乗算し、一方で変数「ｙ」は加数であるため、ワンタイムパッド「ｔ」を加算する。ＣＣ２　４１０及びＣＣ３　４２０は、難読化値「ａ’」、「ｘ’」及び「ｙ’」を用いて計算を行って、「ｂ’」及び「ｃ

’」をそれぞれ算出する。ＣＣ３　４２０は、「ｒ」及び「ｓ」による除算によって「ｂ’」の逆難読化を実行し、信頼できるコンピュータ４（ＴＣ４）４３０は、パッド「ｔ」を減算することによって値「ｃ’」の逆難読化を実行する。これらの手順は、関連する変数の実際の値を攻撃者が決して確認できないことを確実にする。

【００３１】

ゲートレベルのシュレッディングは、他の単一のコンピュータを観察する攻撃者が計算を理解したり、又は元々の計算の入力又は出力を復元したりすることが不可能なため、最高レベルのシュレッディングであり、完全な秘密性を提供する。一方で、ゲートレベルのシュレッディングは最も遅いレベルのシュレッディングでもあり、他のコンピュータ上でＩ／Ｏ演算を実行する能力を提供しない。このレベルでは、プログラムが、ＡＮＤゲート、ＯＲゲート、ＮＡＮＤゲート、ＮＯＲゲート及びＮＯＴゲートから成る回路に分割され、ＮＯＴゲートは、各ゲート演算が確実に２つのオペランドを有するように等しい入力のＮＡＮＤゲートに変換することができる。このように、プログラムが分割された一連の演算は、各々が演算子と、第１のオペランドと、第２のオペランドとを有する回路ゲート演算に分割される。

【００３２】

ある実施形態では、ゲート演算回路内の各ゲートを難読化し、他の２つのコンピュータにわたってシュレッディングする。各ゲート演算のプロセスでは、信頼できるコンピュータが、第１のオペランドを第１の乱数値で難読化し、第２のオペランドを第２の乱数値で難読化する。その後、これらの難読化されたオペランドを、第１のコンピュータが難読化されたオペランドを用いて複数の演算の複数の結果を計算して結果を第２のコンピュータに送信する旨の命令と共に第１のコンピュータに送信する。信頼できるコンピュータは、第２のコンピュータに、演算子と、第１の乱数値と、第２の乱数値とに基づいて複数の結果のうちの１つの結果を選択し、選択された結果を第３の乱数値で難読化し、これを信頼できるコンピュータ又は別のコンピュータとすることができる異なるコンピュータに送信する旨の命令を送信する。

【００３３】

図５に、他の２つのコンピュータ、すなわちクラウドコンピュータ１（ＣＣ１）５１０及びクラウドコンピュータ２（ＣＣ２）２１０にわたってゲート計算をシュレッディングする実施形態例を示す。この例では、ゲート演算「ｇ＝ａ＆ｂ」が、実行すべき現在のシュレッドである。図４と同様に、この実施形態もワンタイムパッドを使用するが、乱数値は「ｒ」、「ｓ」及び「ｕ」とする。信頼できるコンピュータ（ＴＣ）５００は、難読化されたプログラムを以下のように生成して「ｇ＝ａ＆ｂ」を実行する。最初に、ＴＣ５００は、「ａ」及び「ｂ」を難読化するための２つのランダムビット「ｒ」及び「ｓ」をそれぞれ選択する。次に、ＴＣ５００は、「ｃ＝ａ＾ｒ」及び「ｄ＝ｂ＾ｓ」を計算し、ここでの「＾」演算子はＸＯＲ演算を意味する。次に、ＴＣ５００は、「ｃ」及び「ｄ」をＣＣ１　５１０に送信する。ＣＣ１　５１０は、ｅ１＝ｃ＆ｄ、ｅ２＝ｃ＆！ｄ、ｅ３＝ｃ｜！ｄ及びｅ４＝ｃ｜ｄという４つの一時的な値を計算する。次に、ＣＣ１　５１０は、｛ｅ１，ｅ２，ｅ３，ｅ４｝をＣＣ２　５２０に送信する。ＣＣ２　５２０は、「ｒ」及び「ｓ」の値に基づいて、｛ｒ＝０，ｓ＝０｝の場合にはｆ＝ｅ１＾ｕ、｛ｒ＝０，ｓ＝１｝の場合にはｆ＝ｅ２＾ｕ、｛ｒ＝１，ｓ＝０｝の場合にはｆ＝！ｅ３＾ｕ、｛ｒ＝０，ｓ＝０｝の場合にはｆ＝！ｅ４＾ｕというプログラムのうちの１つを実行し、ランダムビット「ｕ」を用いて結果を難読化する。次に、ＣＣ２　５２０は、ｆをＴＣ５００に送信する。ＴＣ５００は、「ｇ＝ｆ＾ｕ」を計算することによって結果を逆難読化する。

【００３４】

図５の実施形態例では、ＴＣ　５００が３つのＸＯＲ演算を計算してデータを難読化し、ＣＣ１　５１０及びＣＣ２　５２０が計算を実行する。ＣＣ１　５１０は４つの一般的な演算を実行し、ＣＣ２　５２０は正しい演算を選択するので、ＣＣ１　５１０及びＣＣ２　５２０は、いずれも入力データ、出力データ又は計算が分からない。ＣＣ２　５２０では、ＡＮＤの計算又は計算の結果が明白なように思えるが、注意深く検討すると、計算

もデータも隠されていることが分かる。ＣＣ２　５２０におけるプログラムは、｛ｅ１，ｅ２，ｅ３，ｅ４｝の値のうちの１つを単純に通過させるもの又はその否定であり、シュレッダプログラムのみが知っている「ｒ」、「ｓ」及び「ｕ」の乱数値に応じてシュレッダプログラムによって生成される。さらに、図６に示すように、ＣＣ２　５２０は、ＣＣ１　５１０におけるｅ４＝ｃ＆ｄ、ｅ３＝ｃ＆！ｄ、ｅ２＝ｃ｜！ｄ及びｅ１＝ｃ｜ｄという変数名の置換に基づいて「ａ　ＯＲ　ｂ」を計算するので、自機がＡＮＤ演算を計算しているか、それともＯＲ演算を計算しているかを知ることはできない。「ｇ＝ａ｜ｂ」を計算する図６は、ＣＣ２　６２０において行われる計算以外の全ての面で図５と同一である。

【００３５】

同様に、ＣＣ２　５２０においてさらなるＮＯＴ演算を追加することによって、ＮＡＮＤ演算又はＮＯＲ演算を計算することもできる。上述したように、単一ビットのＮＯＴ演算は、等しい入力のＮＡＮＤ演算を使用することができる。入力を隠すためのランダムビット「ｒ」と、出力を隠すための別のランダムビット「ｕ」とを用いた同様の方法を使用して、単一ビットを記憶するシーケンス回路を難読化することができる。従って、ＴＣ５００上で（入力及び出力を隠すために使用される全てのランダムなビットの値から成る）ワンタイムパッドを導入し、２つのシュレッドを生成してワンタイムパッドの下で回路を実行することによって、あらゆる回路を難読化することができる。このスキームは、攻撃者がこれらの２つのシュレッドを同時に確認できない限りセキュアである。

【００３６】

図５及び図６に示すワンタイムパッド方法は、（１）ＣＣ２（５２０、６２０）において確認できる値｛ｅ１，ｅ２，ｅ３，ｅ４｝が、ＣＣ１（５１０、６１０）における入力変数「ｃ」及び「ｄ」の４つの組み合わせの特異的パターンを有し、（２）ＡＮＤ演算の値｛ｅ１，ｅ２，ｅ３，ｅ４｝のパターンが、ＯＲ演算の値｛ｅ１，ｅ２，ｅ３，ｅ４｝のパターンと異なる、という理由から、回路を複数回使用する場合にはセキュアでない。ＣＣ２（５２０、６２０）において４つの値｛ｅ１，ｅ２，ｅ３，ｅ４｝を収集して分析する攻撃者は、演算を特定することができる。これに対抗するために、さらなる４つのランダムビット「ｔ１」、「ｔ２」、「ｔ３」及び「ｔ４」を導入して｛ｅ１，ｅ２，ｅ３，ｅ４｝の値を難読化することができる。図７に、これらの新たなビットを用いた計算を示す。

【００３７】

図７は、さらなる難読化のために｛ｔ１，ｔ２，ｔ３，ｔ４｝を使用する以外、図５と同一である。この例では、信頼できるコンピュータ（ＴＣ）７００が、「ｃ」及び「ｄ」を難読化してこれらをクラウドコンピュータ１（ＣＣ１）７１０に送信し、クラウドコンピュータ１（ＣＣ１）７１０が、図５及び図６と同様に｛ｅ１，ｅ２，ｅ３，ｅ４｝を計算する。しかしながら、今回はｅ１をｔ１でＸＯＲ演算し、ｅ２をｔ２でＸＯＲ演算し、ｅ３をｔ３でＸＯＲ演算し、ｅ４をｔ４でＸＯＲ演算して｛ｅ１，ｅ２，ｅ３，ｅ４｝の値を難読化する。一般に、クラウドコンピュータ２（ＣＣ２）７２０、又は第２のシュレッドを処理するコンピュータは、以下の表に示すように、計算される演算と、ランダムビット「ｒ」、「ｓ」、「ｕ」、「ｔ１」、「ｔ２」、「ｔ３」及び「ｔ４」の値とに依存して特定の値「ｅ」を通過させるもの又はその否定のいずれかである。

表１：難読化された回路ゲート演算の選択

| {r, s} | AND | NAND | OR | NOR |
|---|---|---|---|---|
| {r = 0, s = 0} | $f = e_1 \wedge (t_1 \wedge u)$ | $f = !e_1 \wedge (t_1 \wedge u)$ | $f = e_4 \wedge (t_4 \wedge u)$ | $f = !e_4 \wedge (t_4 \wedge u)$ |
| {r = 0, s = 1} | $f = e_2 \wedge (t_2 \wedge u)$ | $f = !e_2 \wedge (t_2 \wedge u)$ | $f = e_3 \wedge (t_3 \wedge u)$ | $f = !e_3 \wedge (t_3 \wedge u)$ |
| {r = 1, s = 0} | $f = !e_3 \wedge (t_3 \wedge u)$ | $f = e_3 \wedge (t_3 \wedge u)$ | $f = !e_2 \wedge (t_2 \wedge u)$ | $f = e_2 \wedge (t_2 \wedge u)$ |
| {r = 1, s = 1} | $f = !e_4 \wedge (t_4 \wedge u)$ | $f = e_4 \wedge (t_4 \wedge u)$ | $f = !e_1 \wedge (t_1 \wedge u)$ | $f = e_1 \wedge (t_1 \wedge u)$ |

図７では、演算が「ｇ＝ａ＆ｂ」であるため、ＣＣ２　７２０は、ＡＮＤ列の下の計算のうちの１つを選択する。その後、ＣＣ２　７２０は「ｆ」をＴＣ７００に送信し、ＴＣ７００が最終結果を求めて「ｆ」を逆難読化する。

【００３８】

４つの演算（ＡＮＤ、ＮＡＮＤ、ＯＲ、ＮＯＲ）の各々について｛ｅ１，ｅ２，ｅ３，ｅ４｝の４つの値を分析すると、４つの演算の各々について（｛ｔ１，ｔ２，ｔ３，ｔ４｝の１６個の値のための）全く同じ１６個のパターンが見られることが分かる。従って、ＣＣ２　７２０における攻撃者は、これらのパターンを用いて演算を特定することができない。しかしながら、ＣＣ２　７２０においてプログラムを見た攻撃者は、「ｒ」の値と「ｓ」の値とが等しいかどうかを判断することができるので、若干の情報漏洩が存在する。この２つの入力ビットのＡＮＤ演算、ＮＡＮＤ演算、ＯＲ演算又はＮＯＲ演算を難読化するスキームは、７つのビット（ｒ，ｓ，ｔ１，ｔ２，ｔ３，ｔ４，ｕ）を含む乱数鍵を用いて２つのシュレッドを生成する。第１のシュレッドは常に同じブール演算を実行し、第２のシュレッドは正しい演算を選択し、これらの２つのシュレッド間の中間値の通信が難読化される。このいずれかの任意の回路を難読化して計算するプロセスを図８で確認することができる。図８の各矩形はコンピュータを表す。「入力を送信」と表記するコンピュータ８００から「計算１」と表記する第１のコンピュータ８１０に初期入力が送信され、第１のコンピュータ８１０において第１のシュレッドが演算を計算して、「選択１」と表記する第２のコンピュータ８２０にこの演算を送信し、第２のコンピュータ８２０において第２のシュレッドが正しい演算を選択する。このプロセスは、回路の結果が計算されて「出力を受信」と表記するコンピュータ８５０に送信されるまで（例えば、それぞれ「計算２」及び「選択２」と表記するコンピュータ８３０及び８４０において、その後にコンピュータ１　８１０又は別の選択されたコンピュータに戻って）継続する。

【００３９】

ハードウェアユニットレベルのシュレッディングでは、ゲートレベルのシュレッディングよりもプライバシーは低下するが、さらに高速で実行される。このレベルのシュレッディングは、特定の機能を実行するハードウェアユニットの演算を含む回路にプログラムを分割する。例としては、汎用コンピュータにおける標準的なハードウェアユニットである整数加算、整数乗算、整数比較及び浮動小数点乗算が挙げられる。このように、これらの一連の演算は、各々が演算子と、第１のオペランドと、第２のオペランドとを有する数学演算である。他のコンピュータにおける攻撃者は、行われる演算のタイプは分かるが、正確な演算を推測することはできない。例えば、攻撃者は、整数加算が行われていることは分かるが、オペランド又は結果の平文値は分からない。最終的に、ハードウェアユニットレベルのシュレッディングは、ゲートレベルのシュレッディングと同様に、他のコンピュータ上でＩ／Ｏ演算を実行する能力を提供しない。

【００４０】

命令レベルのシュレッディングは、プログラムを命令に分割して各シュレッドが命令のサブセットを実行するようにする。命令の例としては、ｘ８６機械語命令及びＪａｖａ（登録商標）仮想マシン（ＪＶＭ）バイトコードが挙げられる。これらの命令も、ハードウェアユニットレベルのシュレッディングと同様に数学演算を伴う。しかしながら、命令レベルのシュレッディングは、ハードウェアユニットレベルのシュレッディングとは異なり、上述した分割デバイスドライバを用いてＩ／Ｏ演算を実行するプログラムに対応する。

【００４１】
ある実施形態では、プログラムを分割した一連の演算の各数学演算を難読化し、２つのコンピュータにわたってシュレッディングする。各数学演算のプロセスでは、信頼できるコンピュータが、第１のオペランドを第１の乱数値で難読化し、第２のオペランドを第２の乱数値で難読化する。その後、これらの難読化されたオペランドを、第１のコンピュータが演算子と第１の難読化されたオペランドと第２の難読化されたオペランドとを用いて第１の演算結果を計算する旨の命令と共に第１のコンピュータに送信する。第１のコンピュータは、第２のコンピュータからの値も受け取り、演算子と第１の結果とこの値とを用いて第２の演算結果を計算し、信頼できるコンピュータ又は別のコンピュータとすることができる異なるコンピュータに第２の結果を送信する。第２のコンピュータは、オペランドを難読化するために使用する乱数値と最終結果とを知っており、この知識を用いて、第１のコンピュータに送信する値を計算する。さらなる演算を伴ういくつかの実施形態では、受け取られる値が、第３の乱数値から第１の乱数値と第２の乱数値との和を差し引いたものである。後述する図１２に、１つのこのような実施形態を示す。乗算演算を伴ういくつかの実施形態では、受け取られる値が、第３の乱数値を第１の乱数値と第２の乱数値との積で除算したものである。後述する図１３に、１つのこのような実施形態を示す。いくつかの実施形態は、演算を乗法難読化スキームから加法難読化スキームに、及びこの逆に遷移させることを必要とすることもできる。後述する図１８に、乗法難読化スキームから加法難読化スキームに遷移する実施形態を示し、後述する図１９に、加法難読化スキームから乗法難読化スキームに遷移する実施形態を示す。

【００４２】
難読化に必要な乱数値は、以下の方法によって生成することができる。プログラムの実行サイクルに、サイクル「ｉ」で実行される命令によって定められる変数が関数「Ｋｅｙ（ｉ）」によってブラインディングされるように付番する。関数「Ｋｅｙ（ｉ）」は、浮動小数点計算中の丸め誤差を制限するために、「大きすぎる又は小さすぎる」ことのない乱数浮動小数点値を生成する。この乱数浮動小数点値は、計算中のあらゆるゼロ除算を防ぐように非ゼロでもある。「Ｋｅｙ」関数は、ノンス「ｉ」のための乱数値を生成するＳａｌｓａ２０などの高速ストリーム暗号を用いて実装される。このストリーム暗号には、加法ブラインド値を生成する１つのシードと、乗法ブラインド値を生成する異なるシードという２つの秘密シードが使用される。プログラム内の各命令は、加算又は乗算ではあるがこれらの両方ではないので、これらの２つのシードの一方を用いて実行結果をブラインディングする。一方のスキームにおいてブラインディングされた命令の結果を他方のスキームの演算で使用する場合には、変換演算を用いてスキームを変更する。

【００４３】
図１２及び図１３に示す例では、ＴＣ（１２００、１２３０、１３００及び１３３０）及びＣＣ３（１２２０及び１３２０）が両シードを知っており、ＣＣ２（１２１０及び１３１０）はいずれのシードも知らない。図１８及び図１９に示す例では、ＴＣ（１８００、１８４０、１９００、及び１９４０）及びＣＣ３（１８２０及び１９２０）が両シードを知っており、ＣＣ２（１８１０及び１９１０）はいずれのシードも知らず、ＣＣ４（１８３０及び１９３０）は乗法シードのみを知っている。

【００４４】
暗号化スキームレベルで行われるシュレッディングは、データに対して行われる演算に依存する準同型暗号スキームを用いて、一連の演算の各演算の各データ値を暗号化する。加算演算は、Ｐａｉｌｌｉｅｒなどの加法準同型暗号（ＡＨＥ）スキームを用いて暗号化され、乗算演算は、Ｅｌ　Ｇａｍａｌなどの乗法準同型暗号（ＭＨＥ）スキームを用いて暗号化される。一方のスキームで暗号化されたデータを、互換性のない演算を用いて操作する必要がある場合には、遷移暗号関数を用いてＡＨＥデータ値をＭＨＥデータ値に変換し、ＭＨＥデータ値をＡＨＥデータ値に変換することができる。これらの遷移関数は、遷移暗号関数を一連の演算に分割し、遷移暗号関数の各演算を、それぞれの演算の結果を計算して別のコンピュータに転送する付属命令と共に少なくとも１つの他のコンピュータに

送信することによってシュレッディングしてセキュリティを高めることができる。後述する図１４には、ＰａｉｌｌｉｅｒからＥｌ　Ｇａｍａｌへのシュレッディングされた遷移暗号関数の実施形態を示し、後述する図１５には、Ｅｌ　ＧａｍａｌからＰａｉｌｌｉｅｒへのシュレッディングされた遷移暗号関数の実施形態を示す。２つの暗号化された整数を比較することも可能である。後述する図１６には、Ｐａｉｌｌｉｅｒ暗号を用いてシュレッディングされた比較関数の実施形態を示し、後述する図１７には、Ｅｌ　Ｇａｍａｌを用いてシュレッディングされた比較関数の実施形態を示す。

【００４５】

図９に、Ｅｌ　Ｇａｍａｌ及びＰａｉｌｌｉｅｒ暗号スキームを用いて「ｚ＝ａ＊ｘ＋ｂ」を実行する実施形態例を示す。信頼できるコンピュータ１（ＴＣ１）９００は、Ｅｌ　Ｇａｍａｌを用いて「ａ」及び「ｘ」の値を暗号化し、Ｐａｉｌｌｉｅｒを用いて「ｂ」の値を暗号化する。「ＥＧ（）」関数は、Ｅｌ　Ｇａｍａｌでの暗号化を意味し、「ＥＰ（）」関数は、Ｐａｉｌｌｉｅｒでの暗号化を意味する。クラウドコンピュータ２（ＣＣ２）９１０は、ＭＨＥ暗号化値に基づく乗算「ａ＊ｘ」を計算してＭＨＥ暗号化値「ｙ’」を生成する。次に、ＣＣ２　９１０は、クラウドコンピュータ３（ＣＣ３）９２０との間でシュレッディングされた遷移関数「ＧＰ１（）」、「ＧＰ２（）」及び「ＧＰ３（）」を用いてＥｌ　Ｇａｍａｌ暗号化値「ｙ’」をＰａｉｌｌｉｅｒ暗号化値「ｙ’’」に変換し、ＡＨＥ暗号化値を乗算することによって「ｂ＋ｙ」を加算する。その後、信頼できるコンピュータ４（ＴＣ４）９３０が「ｚ」のＰａｉｌｌｉｅｒ暗号化値を受け取り、暗号解読Ｐａｉｌｌｉｅｒ関数「ＤＰ（）」を用いてこの値を暗号解読して解を得る。シュレッディングされた遷移関数「ＧＰ１（）」、「ＧＰ２（）」及び「ＧＰ３（）」については、以下で図１５に関してさらに完全に説明する。

【００４６】

上述したように、分割デバイスドライバは、シュレッド内の入力及び出力演算が他のコンピュータ上で実行されることを可能にする。さらなるセキュリティレイヤとしては、Ｉ／Ｏ装置からの平文データが信頼できるコンピュータにしか分からないように、シュレッドがＩ／Ｏ装置からの暗号化データにしか作用できないようにすることが好ましい。図１０に、単一のコンピュータ上で動作する通常のデバイスドライバ相互作用の動作例を示し、対照的に図１１には、信頼できるコンピュータとクラウドコンピュータとの間の分割デバイスドライバ相互作用の詳細な動作例を示す。

【００４７】

図１０に、単一のコンピュータ上で動作する通常のデバイスドライバ相互作用の動作例を示す。図１０に示すアプリケーション例であるアプリケーション１０００は、キーボードであるキーボードハードウェア１０２０に入力を要求し、この入力をコンソール又は画面であるコンソールハードウェア１０４０に出力する。このプロセスは、図１０に表記される８つのステップを必要とする。ステップ１において、アプリケーションがコンピュータのカーネル内のカーネルキーボードデバイスドライバ（ＫＫ）１０１０に文字を要求する。ステップ２において、ＫＫ１０１０がキーボードハードウェア１０２０に文字を要求する。ステップ３において、キーボードハードウェア１０２０が入力を受け取り、ＫＫ１０１０に文字で応答する。ステップ４において、ＫＫ１０１０がこの文字をアプリケーションに受け渡して入力動作を完了する。ステップ５において、アプリケーションが出力動作を開始し、コンピュータのカーネル内のカーネルコンソールデバイスドライバ（ＫＣ）１０３０に文字を送信する。ステップ６において、ＫＣ１０３０がこの文字をコンソールハードウェア１０４０に送信する。ステップ７において、コンソールハードウェア１０４０がこの文字を印刷してＫＣ１０３０に状態を送信する。ステップ８において、ＫＣ１０３０がこの状態をアプリケーション１０００に受け渡す。この段階で出力動作は完了し、アプリケーション１０００は、例えば出力がコンソール画面に印刷されたかどうか、エラーであるかなどの出力の状態が分かる。

【００４８】

図１１には、図１０と同様の機能を実行する、信頼できるコンピュータ（ＴＣ）１１０

２とクラウドコンピュータ（ＣＣ）１１０４との間の分割デバイスドライバの実施形態の動作を示しており、実行シュレッド１１００は、キーボードハードウェア１１６０に入力を要求した（ステップ１～１２）後に、この入力をコンソールハードウェア１１６２に出力する（ステップ１３～２４）。ＣＣ１１０４は、セキュリティを目的として信頼できるコンピュータから入力を得て（ここでは、ＴＣ１１０２である）信頼できるコンピュータに出力しなければならないので、デバイスドライバは、ＣＣ１１０４とＴＣ１１０２との間で分割される。

【００４９】

図１１の入力部分は、以下の通りである。ステップ１において、実行シュレッド１１００が、ユーザモードにおける分割キーボードデバイスドライバであるユーザモードキーボードデバイスドライバ（ＣＣＵＭＫ）１１１０に文字を要求する。ステップ２において、ＣＣＵＭＫ１１１０が、カーネルネットワークデバイスドライバ（ＣＣＫＮ）１１２０を用いて文字を要求する。ステップ３において、ＣＣＫＮ１１２０が、ＴＣ上のカーネルネットワークデバイスドライバ（ＴＣＫＮ）１１３０と通信する。ステップ４において、ＴＣＫＮ１１３０が、ユーザモードキーボードデバイスドライバ（ＴＣＵＭＫ）１１４０に文字を要求する。ステップ５において、ＴＣＵＭＫ１１４０が、カーネルキーボードデバイスドライバ（ＴＣＫＫ）１１５０に文字を要求する。ステップ６において、ＴＣＫＫ１１５０が、キーボードハードウェア１１６０に文字を要求する。ステップ７において、キーボードハードウェア１１６０が入力を受け取り、ＴＣＫＫ１１５０に文字で応答する。ステップ８において、ＴＣＫＫ１１５０が、この文字をＴＣＵＭＫ１１４０に送信する。ステップ９において、ＴＣＵＭＫ１１４０が、受け取った文字を暗号化してＴＣＫＮ１１３０に送信する。ステップ１０において、ＴＣＫＮ１１３０が、暗号化文字をＣＣＫＮ１１２０に送信する。ステップ１１において、ＣＣＫＮ１１２０が、暗号化文字をＣＣＵＭＫ１１１０に送信する。ステップ１２において、ＣＣＵＭＫ１１１０が、暗号化文字を実行シュレッド１１００に受け渡して入力動作を完了する。

【００５０】

図１１の出力部分は、以下の通りであり、実行シュレッド１１００が、ステップ１～１２のプロセスから暗号化文字を受け取ったと想定する。ステップ１３において、実行シュレッド１１００が出力動作を開始し、ユーザモードにおける分割コンソールデバイスドライバであるユーザモードコンソールデバイスドライバ（ＣＣＵＭＣ）１１１２に暗号化文字を送信する。ステップ１４において、ＣＣＵＭＣ１１１２が、ＣＣＫＮ１１２０を用いて暗号化文字を送信する。ステップ１５において、ＣＣＫＮ１１２０がＴＣＫＮ１１３０と通信する。ステップ１６において、ＴＣＫＮ１１３０が、ユーザモードコンソールデバイスドライバ（ＴＣＵＭＣ）１１４２に暗号化文字を送信する。ステップ１７において、ＴＣＵＭＣ１１４２が文字を暗号解読し、カーネルコンソールデバイスドライバ（ＴＣＫＣ）１１５２に平文文字を送信する。ステップ１８において、ＴＣＫＣ１１５２がコンソールハードウェア１１６２に平文文字を送信する。ステップ１９において、コンソールハードウェア１１６２が文字を印刷してＴＣＫＣ１１５２に状態を送信する。ステップ２０において、ＴＣＫＣ１１５２がＴＣＵＭＣ１１４２に状態を送信する。ステップ２１において、ＴＣＵＭＣ１１４２がＴＣＫＮ１１３０に状態を送信する。ステップ２２において、ＴＣＫＮ１１３０がＣＣＫＮ１１２０に状態を送信する。ステップ２３において、ＣＣＫＮ１１２０がＣＣＵＭＣ１１１２に状態を送信する。ステップ２４において、ＣＣＵＭＣ１１１２が実行シュレッド１１００に状態を送信する。この段階で出力動作は完了し、実行シュレッド１１００は、例えば出力がコンソール画面に印刷されたかどうか、エラーであるかなどの出力の状態が分かる。

【００５１】

図１０のアプリケーション１０００と図１１の実行シュレッド１１００との間の分割ドライバ以外のさらなる相違点は、実行シュレッド１１００が暗号化データに基づいて動作し、従ってここではＣＣ１１０４である攻撃者が観察しているコンピュータ上で実行できる点である。データの暗号化及び暗号解読のための鍵は、ＴＣＵＭＫ１１４０及びＴＣＵ

ＭＣ１１４２において鍵を使用するＴＣ　１１０２上でしか利用できない。

【００５２】

上述したように、コード難読化は、オリジナルプログラムのロジックを隠すために使用することができ、データ難読化と共に使用することができる。データ難読化については、図１２及び図１３を用いてさらに理解することができ、コード難読化は、以下に限定されるわけではないが、オペコード置換、関数マージング、制御フロー平坦化、及び不明瞭な述語を含むデコイコードを含むことができる。

【００５３】

図１２に、データ難読化を用いて和「ｇ＝ａ＋ｂ」を計算する実施形態を示す。信頼できるコンピュータ１（ＴＣ１）１２００は、「ａ」及び「ｂ」を難読化するための２つの乱数「ｒ」及び「ｓ」をそれぞれ選択する。ＴＣ１　１２００は、「ａ＋ｒ」及び「ｂ＋ｓ」を計算し、結果を変数「ｃ」及び「ｄ」にそれぞれ割り当てる。次に、ＴＣ１　１２００は、（ｃ、ｄ）をクラウドコンピュータ２（ＣＣ２）１２１０に送信する。ＣＣ２　１２１０は、和「ｅ＝ｃ＋ｄ」を計算し、クラウドコンピュータ３（ＣＣ３）１２２０にブラインディングを要求する。ＣＣ３　１２２０には、最終結果をブラインディングするために使用する乱数値「ｒ」及び「ｓ」、さらには第３の乱数値「ｕ」の知識が与えられる。ＣＣ３　１２２０は、「ｕ－ｒ－ｓ」の値を計算してこの値を変数「ｈ」に割り当て、これをＣＣ２　１２１０に送信する。ＣＣ２　１２１０は、ＣＣ３　１２２０から値「ｈ」を受け取り、「ｅ＋ｈ」を計算して結果を変数「ｆ」に割り当て、これを信頼できるコンピュータ４（ＴＣ４）１２３０に送信する。その後、ＴＣ４　１２３０は、「ｆ－ｕ」を計算することによって結果をアンブラインディングして動作を完了する。従って、ＴＣ　１２００及び１２３０が、入力データ（「ａ」及び「ｂ」）を難読化するための鍵として３つの乱数「ｒ」、「ｓ」及び「ｕ」を使用してデータ（「ｇ」）を出力する一方で、２つのクラウドコンピュータ１２１０及び１２２０が実際の和を計算する。入力データ又は出力データは、クラウドコンピュータに分からない。

【００５４】

図１３に、データ難読化を用いて積「ｇ＝ａ＊ｂ」を計算する実施形態を示す。信頼できるコンピュータ１（ＴＣ１）１３００は、「ａ」及び「ｂ」を難読化するための２つの乱数「ｒ」及び「ｓ」をそれぞれ選択する。ＴＣ１　１３００は、「ａ＊ｒ」及び「ｂ＊ｓ」を計算して結果を「ｃ」及び「ｄ」にそれぞれ割り当てる。次に、ＴＣ１　１３００は、（ｃ、ｄ）をクラウドコンピュータ２（ＣＣ２）１３１０に送信する。ＣＣ２　１３１０は、積「ｅ＝ｃ＊ｄ」を計算し、クラウドコンピュータ３（ＣＣ３）１３２０にブラインディングを要求する。ＣＣ３　１３２０には、最終結果をブラインディングするために使用する乱数値「ｒ」及び「ｓ」、さらには第３の乱数値「ｕ」の知識が与えられる。ＣＣ３　１３２０は、「ｕ／（ｒ＊ｓ）」の値を計算してこの値を変数「ｈ」に割り当て、これをＣＣ２　１３１０に送信する。ＣＣ２　１３１０は、ＣＣ３　１３２０から値「ｈ」を受け取り、「ｅ＊ｈ」を計算して結果を変数「ｆ」に割り当て、これを信頼できるコンピュータ４（ＴＣ４）１３３０に送信する。その後、ＴＣ４　１３３０は、「ｆ／ｕ」を計算することによって結果をアンブラインディングして動作を完了する。従って、ＴＣ　１３００及び１３３０が入力データ（「ａ」及び「ｂ」）を難読化するための鍵として３つの乱数「ｒ」、「ｓ」及び「ｕ」を使用してデータ（「ｇ」）を出力する一方で、２つのクラウドコンピュータ１３１０及び１３２０が実際の積を計算する。入力データ又は出力データは、クラウドコンピュータに分からない。

【００５５】

オペコード置換は、実際のオペコードの代わりにランダムなオペコードを用いてシュレッドの静的分解を阻止するものである。オペコードを含む一連の演算に分割されたプログラムでは、プログラムオペコードをランダムなオペコード順列にマッピングする置換マップを生成することができる。その後、この置換マップを用いて一連のオペコードをランダムなオペコード順列に変換し、これをリモートコンピュータに送信して、他のそれぞれのコンピュータによる逆難読化において使用することができる。逆難読化は、置換マップを

保持するリモートコンピュータにおいて置換マップの一部に対応するインデックスを他のコンピュータから受け取り、この他のコンピュータにリモートコンピュータから置換マップの一部を送信することによって行うことができる。対称的に、他のコンピュータは、置換マップの一部に対応するインデックスをリモートコンピュータに送信し、リモートコンピュータから置換マップの一部を受け取り、置換マップを用いてランダムなオペコード順列を元々の一連のオペコードに変換できることが分かる。

【００５６】

ある実施形態では、置換マップを用いたオペコード置換をＪａｖａプログラムと併用することができる。Ｊａｖａバイトコードは２５６個のオペコードを有し、これらのうちの５１個のオペコード（２０３～２５３の範囲）は未定義である。元々の２５６個のオペコードをランダムなオペコード順列にマッピングする置換マップを導入する。置換マップは、シュレッドを生成するコンピュータ及びリモートコンピュータには分かるが、置換されたオペコードを用いてシュレッドを実行する他のコンピュータには分からない。未使用のオペコード２５３は、難読化されたプログラムによってＧＥＴ　ＭＡＰ命令として使用される。ＧＥＴ　ＭＡＰ命令は、シュレッドを実行する他のコンピュータによってリモートコンピュータに送信された３２ビットインデックスをオペランドと見なす。リモートコンピュータは、この特定のシュレッドに使用すべき置換マップを含む２５６バイトの結果を戻す。ＧＥＴ　ＭＡＰ命令は、ユーザが指定したシュレッド内の回数だけでなくシュレッドの最初にも挿入される。セキュリティのためには、同じマップを使用する大きなコード部分の統計的分析を阻止するために数多くのＧＥＴ　ＭＡＰ命令が存在すべきである。性能最適化のためには、ループ内にわずかな数のＧＥＴ　ＭＡＰ命令が存在すべきである。

【００５７】

関数マージングは、無関係な関数を単一の関数に組み合わせる。無関係な関数は、それぞれがパラメータを有し、これらのパラメータのそれぞれの挙動を単一の関数内に保持する。このようなスキームを実装するために、この単一の関数は、無関係な関数のパラメータとさらなるパラメータとを全て取ってどの挙動を実行すべきかを選択する。無関係な関数の数が多い場合には、多少関連する関数のグループをマージして、グループ毎に単一のマージされた関数が存在するようにすることができる。

【００５８】

チェンクシフィケーション（ｃｈｅｎｘｉｆｉｃａｔｉｏｎ）としても知られている制御フロー平坦化は、関数を無限ループに変換する。ループから抜け出るには、元々の関数の挙動と同じ挙動を実行するｓｗｉｔｃｈ文を追加する。

【００５９】

デコイコードは、攻撃者が分析を必要とするコードの量を増やすために使用される。これは、コンピュータプログラムにデコイコードを挿入することによって行うことができる。デコイコードは、複数のわずかな変異を伴うコンピュータプログラムのオリジナルコードを含むことができる。このわずかな変異は、静的に検出不能なエラーを生じ、わずかな変異の数はユーザが指定することができる。デコイコードは、不明瞭な述語スキームの一部とすることができる。このようなスキームは、設定及び実行は容易であるが分析が困難な述語である不明瞭な述語に依存する標的文を作成することによって関数の静的分析を阻止する。不明瞭な述語はアレイエイリアシングを活用することができ、これらの述語には、常に真、常に偽及び時々真という３つのタイプの述語が存在する。常に真の述語は、「ｉｆ」分岐内でオリジナルコードを実行し、「ｅｌｓｅ」分岐内でデコイコードを実行する。常に偽の述語は、「ｉｆ」分岐内でデコイコードを実行し、「ｅｌｓｅ」分岐内でオリジナルコードを実行する。時々真の述語は、２つの分岐上でオリジナルコードとオリジナルコードの難読化バージョンとを実行する。

【００６０】

上述したように、図１４～図１７には、Ｐａｉｌｌｉｅｒ暗号及びＥｌ　Ｇａｍａｌ暗号を用いた実施形態を示す。図１４～図１７の実施形態では、特に明記しない限り以下の想定を適用する。Ｐａｉｌｌｉｅｒにおけるメッセージ「ｍ」の公開鍵「ｎ」による暗号

化は、ＥＰ（ｍ）＝（ｎ＋１）$^{m}$ ｒ$^{n}$ ｍｏｄ ｎ$^{2}$として定められ、ここでの「ｒ」は、公開鍵「ｎ」よりも小さく「ｎ」と互いに素であるランダムな非ゼロの整数である。Ｐａｉｌｌｉｅｒにおける暗号「ｃ」の秘密鍵（ｂ、ｕ）による暗号解読は、ＤＰ（ｃ）＝ｕ（（ｃ$^{b}$ ｍｏｄ ｎ$^{2}$－１）／ｎ）ｍｏｄ ｎとして定められる。Ｅｌ Ｇａｍａｌにおけるメッセージ「ｍ」の公開鍵（ｎ、ｇ、ｑ、ｈ）による暗号化は、ＥＧ（ｍ）＝（ｇ$^{r}$ ｍｏｄ ｎ、ｍ ｈ$^{r}$ ｍｏｄ ｎ）として定められ、ここでの「ｒ」は、「ｎ」よりも小さいランダムな非ゼロの整数である。Ｅｌ Ｇａｍａｌにおける暗号（ｅ，ｃ）の秘密鍵「ｘ」による暗号解読は、ＤＧ（ｅ、ｃ）＝ｅ$^{q-x}$ ｃ ｍｏｄ ｎとして定められる。

【００６１】

図１４には、Ｐａｉｌｌｉｅｒ暗号からＥｌ Ｇａｍａｌ暗号へのシュレッディングされた遷移暗号関数の実施形態を示す。暗号文「ｃ」をＰａｉｌｌｉｅｒから暗号解読してＥｌ Ｇａｍａｌに暗号化することは、以下のように定められる。

ＰＧ（ｃ）＝
ｌｅｔ ｍ＝ｕ（（ｃ$^{b}$ ｍｏｄ ｎ$^{2}$－１）／ｎ）ｍｏｄ ｎ
ｉｎ（ｇ$^{r}$ ｍｏｄ ｎ，ｍ ｈ$^{r}$ ｍｏｄ ｎ）

関数「ＰＧ（ｃ）」は、以下のプロセスを用いて「ＰＧ１（ｃ）」、「ＰＧ２（ａ、ｃ）」及び「ＰＧ３（ｗ）」にシュレッディングすることができる。値ｂ１及びｂ２は、ｂ１とｂ２の和が、ｂ（ｂ１＋ｂ２＝ｂ）に等しくなるようにランダムに選択される。値ｕ１及びｕ２は、ｕ１とｕ２の積がｕ ｍｏｄ ｎ（ｕ１＊ｕ２＝ｕ ｍｏｄ ｎ）に等しくなるようにランダムに選択される。これらの関数は、以下のように定められる。

ＰＧ１（ｃ）＝ｃ$^{b1}$ ｍｏｄ ｎ$^{2}$
ＰＧ２（ａ，ｃ）＝
ｌｅｔ ｂ＝ｕ２（（ａ ｃｂ$^{2}$ ｍｏｄ ｎ$^{2}$）－１）／ｎ）ｍｏｄ ｎ
ｉｎ（ｇ$^{r}$ ｍｏｄ ｎ，ｂ ｈ$^{r}$ ｍｏｄ ｎ）
ＰＧ３（ｗ）＝ｕ１ ｗ ｍｏｄ ｎ

これらの関数を組み合わせると、以下の関数「ＰＧ＿ｓｈｒｅｄ（ｃ）」が得られる。

ＰＧ＿ｓｈｒｅｄ（ｃ）＝
ｌｅｔ ａ＝ＰＧ１（ｃ）
ｌｅｔ（ｖ，ｗ）＝ＰＧ２（ａ，ｃ）
ｌｅｔ ｚ＝ＰＧ３（ｗ）
ｉｎ（ｖ，ｚ）

【００６２】

図１４では、信頼できるコンピュータ１（ＴＣ１）１４００が、公開鍵「ｐｋ」を用いてメッセージ「ｍ」をＰａｉｌｌｉｅｒ暗号文「ｃ」に暗号化した後に、これをクラウドコンピュータ２（ＣＣ２）１４１０に送信する。ＣＣ２ １４１０は、「ＰＧ１（ｃ）」を計算して結果を「ａ」に割り当てた後に、「ａ」及び「ｃ」をクラウドコンピュータ３（ＣＣ３）１４２０に送信する。ＣＣ３ １４２０は、「ＰＧ２（ａ，ｃ）」を計算して結果を（ｖ，ｗ）に割り当た後に、これをＣＣ２ １４１０に返送する。ＣＣ２ １４１０は、「ＰＧ３（ｗ）」を計算して結果を「ｚ」に割り当た後に、（ｖ，ｚ）を信頼できるコンピュータ４（ＴＣ４）１４３０に送信する。ＴＣ４ １４３０は、この時点でＥｌ Ｇａｍａｌ暗号である暗号文（ｖ，ｚ）を受け取り、秘密鍵「ｓｋ」を用いてこれを暗号解読してメッセージ「ｍ」を明らかにする。全てのランダムに選択された値「ｂ１」、「ｂ２」、「ｕ１」及び「ｕ２」は、ＣＣ２ １４１０にもＣＣ３ １４２０にも分からない。図１４では、公開鍵「ｐｋ」が「ｎ」と置き換わるように意図され、秘密鍵「ｓｋ」が「ｘ」と置き換わるように意図される。

【００６３】

図１５に、Ｅｌ ＧａｍａｌからＰａｉｌｌｉｅｒへのシュレッディングされた遷移暗号関数の実施形態を示す。暗号文（ｅ，ｃ）をＥｌ Ｇａｍａｌから暗号解読してＰａｉ

ｌｌｉｅｒに暗号化することは、以下のように定められる。

$$GP(e, c) = \mathrm{let}\ m = e^{q-x}\ c \bmod n\ \mathrm{in}\ (n+1)^{m}\ r^{n} \bmod n^{2}$$

関数「ＧＰ（ｅ，ｃ）」は、以下のプロセスを用いて「ＧＰ（ｅ）」、「ＧＰ（ｆ）」及び「ＧＰ（ｖ，ｃ）」にシュレッディングすることができる。値ｘ１及びｘ２は、ｘ１とｘ２の積が「ｑ－ｘ」（ｘ１＊ｘ２＝ｑ－ｘ）に等しくなるようにランダムに選択される。これらの関数は、以下のように定められる。

$$GP1(e) = e^{x1} \bmod n$$

$$GP2(f) = \mathrm{let}\ a = f^{x2} \bmod n\ \mathrm{in}\ (n+1)^{a} \bmod n^{2}$$

$$GP3(v, c) = \mathrm{let}\ w = v^{c} \bmod n^{2}\ \mathrm{in}\ w\ r^{n} \bmod n^{2}$$

これらの関数を組み合わせると、以下の関数「ＧＰ＿ｓｈｒｅｄ（ｅ，ｃ）」が得られる。

$$GP\_shred(e, c) = \mathrm{let}\ f = GP1(e)\ \mathrm{let}\ v = GP2(f)\ \mathrm{in}\ GP3(v, c)$$

【００６４】

図１５では、信頼できるコンピュータ１（ＴＣ１）１５００が、公開鍵「ｐｋ」を用いてメッセージ「ｍ」をＥｌ　Ｇａｍａｌ暗号文（ｅ，ｃ）に暗号化した後に、これをクラウドコンピュータ２（ＣＣ２）１５１０に送信する。ＣＣ２　１５１０は、「ＧＰ１（ｅ）」を計算して結果を「ｆ」に割り当てた後に、「ｆ」をクラウドコンピュータ３（ＣＣ３）１５２０に送信する。ＣＣ３　１５２０は、「ＧＰ２（ｆ）」を計算して結果を「ｖ」に割り当てた後に、これをＣＣ２　１５１０に返送する。ＣＣ２　１５１０は、「ＧＰ３（ｖ，ｃ）」を計算して結果を「ｚ」に割り当てた後に、「ｚ」を信頼できるコンピュータ４（ＴＣ４）１５３０に送信する。ＴＣ４　１５３０は、この時点でＰａｉｌｌｉｅｒ暗号である暗号文「ｚ」を受け取り、秘密鍵「ｓｋ」を用いてこれを暗号解読してメッセージ「ｍ」を明らかにする。ランダムに選択された値「ｘ１」及び「ｘ２」は、ＣＣ２　１５１０にもＣＣ３　１５２０にも分からない。図１５では、公開鍵「ｐｋ」が（ｎ，ｇ，ｑ，ｈ）と置き換わるように意図され、秘密鍵「ｓｋ」が（ｂ，ｕ）と置き換わるように意図される。

【００６５】

図１６に、Ｐａｉｌｌｉｅｒ暗号を使用するシュレッディングされた比較関数の実施形態を示す。Ｐａｉｌｌｉｅｒスキームにおいて２つの整数（いずれもｎ／２未満であり、ここでのｎは公開鍵である）を暗号化する場合、これらの間の順序比較は以下のように定めることができ、

$$CP(c_1, c_2) = \mathrm{let}\ c_3 = \mathrm{invert}\ c_2 \bmod n^{2}\ \mathrm{let}\ c = c_1\ c_3 \bmod n^{2}\ \mathrm{let}\ d = u((c^{b} \bmod n^{2} - 1)/n) \bmod n\ \mathrm{in}\ \mathrm{if}\ d = 0\ \mathrm{then}\ EQ\ \mathrm{else\ if}\ d < n/2\ \mathrm{then}\ GT\ \mathrm{else}\ LT$$

ここでの「ＥＱ」は「$c_1=c_2$」を意味し、「ＧＴ」は「$c_1>c_2$」を意味し、「ＬＴ」は「$c_1<c_2$」を意味する。なお、「ｃ」は、「$c_1$」と「$c_2$」との間の暗号化差分であり、「ｄ」は、「$c_1$」と「$c_2$」との間の暗号解読差分である。関数「ＣＰ（$c_1$，$c_2$）」は、以下のプロセスを用いて「ＣＰ１（$c_1$，$c_2$）」、「ＣＰ２（ａ，ｃ）」及び「ＣＰ３（ｂ）」にシュレッディングすることができる。値ｂ１及びｂ２は、ｂ１とｂ２の和がｂ（ｂ１＋ｂ２＝ｂ）に等しくなるようにランダムに選択される。値ｕ１及びｕ２は、ｕ１とｕ２の積がｕ　ｍｏｄ　ｎ（ｕ１＊ｕ２＝ｕ　ｍｏｄ　ｎ）に等しくなるようにランダムに選択される。これらの関数は、以下のように定められる。

ＣＰ１（$c_1$，$c_2$）＝
　　ｌｅｔ　$c_3$＝ｉｎｖｅｒｔ　$c_2$　ｍｏｄ　$n^2$
　　ｌｅｔ　ｃ＝$c_1$　$c_3$　ｍｏｄ　$n^2$
　　ｌｅｔ　ａ＝$c^{b1}$　ｍｏｄ　$n^2$
　　ｉｎ（ａ，ｃ）
ＣＰ２（ａ，ｃ）＝
　　ｌｅｔ　ｂ＝ｕ２（（ａ　$c^{b2}$　ｍｏｄ　$n^2$）－１）／ｎ）ｍｏｄ　ｎ
　　ｉｎ　ＣＰ３（ｂ）
ＣＰ３（ｂ）＝
　　ｌｅｔ　ｄ＝ｕ１　ｂ　ｍｏｄ　ｎ
　　ｉｎ
　　　　ｉｆ　ｄ＝０　ｔｈｅｎ　ＥＱ
　　　　ｅｌｓｅ　ｉｆ　ｄ＜ｎ／２　ｔｈｅｎ　ＧＴ
　　　　ｅｌｓｅ　ＬＴ

これらの関数を組み合わせると、以下の関数「ＣＰ＿ｓｈｒｅｄ（ｃ１，ｃ２）」が得られる。

ＣＰ＿ｓｈｒｅｄ（$c_1$，$c_2$）＝
　　ｌｅｔ（ａ，ｃ）＝ＣＰ１（$c_1$，$c_2$）
　　ｌｅｔ　ｂ＝ＣＰ２（ａ，ｃ）
　　ｉｎ　ＣＰ３（ｂ）

図１６では、信頼できるコンピュータ１（ＴＣ１）１６００が、公開鍵「ｐｋ」を用いてメッセージ「ｍ１」をＰａｉｌｌｉｅｒ暗号文「ｃ１」に暗号化し、公開鍵「ｐｋ」を用いてメッセージ「ｍ２」をＰａｉｌｌｉｅｒ暗号文「ｃ２」に暗号化する。その後、ＴＣ１　１６００は、（$c_1$，$c_2$）をクラウドコンピュータ２（ＣＣ２）１６１０に送信する。ＣＣ２　１６１０は、「ＣＰ１（$c_1$，$c_2$）」を計算して結果を（ａ，ｃ）に割り当てた後に、（ａ，ｃ）をクラウドコンピュータ３（ＣＣ３）１６２０に送信する。ＣＣ３　１６２０は、「ＣＰ２（ａ，ｃ）」を計算して結果を「ｂ」に割り当てた後に、これをクラウドコンピュータ４（ＣＣ４）１６３０に送信する。ＣＣ４　１６３０は、「ＣＰ３（ｂ）」を計算して結果を「ｚ」に割り当てた後に、「ｚ」をＣＣ２　１６１０に送信する。ＣＣ２　１６１０は「ｚ」を受け取り、条件文を実行して「ＥＱ」、「ＧＴ」又は「ＬＴ」を判断し、これらをさらなる計算に使用することができる。全てのランダムに選択された値「ｂ１」、「ｂ２」、「ｕ１」及び「ｕ２」は、クラウドコンピュータに分からない。図１６では、公開鍵「ｐｋ」が「ｎ」と置き換わるように意図される。

【００６６】

図１７に、Ｅｌ　Ｇａｍａｌ暗号を使用するシュレッディングされた関数の実施形態を示す。Ｅｌ　Ｇａｍａｌスキームにおいて２つの整数（いずれもｎ／２未満であり、ここでのｎは公開鍵である）を暗号化する場合、これらの間の順序比較は以下のように定めることができ、

ＣＧ（（$e_1$，$c_1$），（$e_2$，$c_2$））＝
　　ｌｅｔ　$p_1$＝ＧＰ（$e_1$，$c_1$）
　　ｌｅｔ　$p_2$＝ＧＰ（$e_2$，$c_2$）

in CP($p_1$, $p_2$)

ここでの「GP()」は、図１５で定められるＥｌ GamalからＰａｉｌｌｉｅｒへの遷移関数であり、「ＣＰ()」は、図１６で定められるＰａｉｌｌｉｅｒ比較関数である。関数ＣＧ((ｅ１, ｃ１), (ｅ２, ｃ２))は、以下のようにシュレッディングすることができる。

CG_shred((e1, c1), (e2, c2)) =
let p1=GP_shred(e1, c1)
let p2=GP_shred(e2, c2)
in CP_shred(p1, p2)

図１７では、信頼できるコンピュータ１（ＴＣ１）１７００が、公開鍵「ｐｋ」を用いてメッセージ「ｍ１」をＥｌ Gamal暗号文（ｅ１, ｃ１）に暗号化し、公開鍵「ｐｋ」を用いてメッセージ「ｍ２」をＥｌ Gamal暗号文（ｅ２, ｃ２）に暗号化する。その後、ＴＣ１ １７００は、（ｅ１, ｃ１）及び（ｅ２, ｃ２）をクラウドコンピュータ２（ＣＣ２）１７１０に送信する。ＣＣ２ １７１０は、「ＧＰ１（ｅ１）」を計算して結果を「ｆ１」に割り当て、「ＧＰ１（ｅ２）」を計算して結果を「ｆ２」に割り当てた後に、「ｆ１」及び「ｆ２」をクラウドコンピュータ３（ＣＣ３）１７２０に送信する。ＣＣ３ １７２０は、「ＧＰ２（ｆ１）」を計算して結果を「ｖ１」に割り当て、「ＧＰ２（ｆ２）」を計算して結果を「ｖ２」に割り当てた後に、「ｖ１」及び「ｖ２」をＣＣ２ １７１０に返送する。ＣＣ２ １７１０は、「ＧＰ３（ｖ１, ｃ１）」を計算して結果を「ｐ１」に割り当て、「ＧＰ３（ｖ２, ｃ２）」を計算して結果を「ｐ２」に割り当てる。ＣＣ２ １７１０は、「ＣＰ１（ｐ１, ｐ２）」を計算して結果を（ａ, ｃ）に割り当てた後に、（ａ, ｃ）をクラウドコンピュータ３（ＣＣ３）１７２０に送信する。ＣＣ３ １７２０は、「ＣＰ２（ａ, ｃ）」を計算して結果を「ｂ」に割り当て、これをクラウドコンピュータ４（ＣＣ４）１７３０に送信する。ＣＣ４ １７３０は、「ＣＰ３（ｂ）」を計算して結果を「ｚ」に割り当てた後に、「ｚ」をＣＣ２ １７１０に送信する。ＣＣ２ １７１０は「ｚ」を受け取り、条件文を実行して「ＥＱ」、「ＧＴ」、又は「ＬＴ」を判断し、これらをさらなる計算に使用することができる。全てのランダムに選択された値は、クラウドコンピュータに分からない。図１７では、公開鍵「ｐｋ」が（ｎ, ｇ, ｑ, ｈ）と置き換わるように意図される。

【００６７】

上述したように、数学演算のシュレッディング及び難読化を行ういくつかの実施形態は、演算を乗法難読化スキームから加法難読化スキームに、及びこの逆に変換することを必要とすることもできる。図１８及び図１９に、これらの遷移関数の２つの実施形態例を示す。

【００６８】

図１８には、乗法難読化スキームから加法難読化スキームに変換する実施形態を示す。この実施形態例では、プログラムが、乱数「ｒ」による「ａ」の乗法ブラインディングを実行するが、乱数「ｓ」による「ａ」の加法ブラインディングを必要とする。変換プロセスは、セキュリティのためにシュレッディングされる。最初に、信頼できるコンピュータ１（ＴＣ１）１８００及び信頼できるコンピュータ５（ＴＣ５）１８４０は、「ａ」をブラインディングするための２つの乱数「ｒ」及び「ｓ」を選択する。ＴＣ１ １８００は、「ａ＊ｒ」を計算して結果を変数「ｃ」に割り当てた後に、これをクラウドコンピュータ２（ＣＣ２）１８１０に送信する。ＣＣ２ １８１０は、乗法から加法へのブラインディングをクラウドコンピュータ３（ＣＣ３）１８２０に要求する。ＣＣ３ １８２０は、乱数値「ｒ」及び「ｓ」の知識を与えられ、「ｒ＊ｓ」を計算し、この積を変数「ｈ」に割り当てて「ｈ」をＣＣ２ １８１０に送信する。ＣＣ２ １８１０は、ＣＣ３ １８２０から値「ｈ」を受け取り、「ｃ＋ｈ」を計算し、結果を変数「ｅ」に割り当ててこれをクラウドコンピュータ４（ＣＣ４）１８３０に送信する。次に、ＣＣ４ １８３０は、「

ｅ／ｒ」を計算することによって積をアンブラインディングし、これを変数「ｆ」に割り当てて「ｆ」をＴＣ５　１８４０に送信する。最後に、ＴＣ５　１８４０は、「ａ＋ｓ」に等しいａの加法ブラインディング値を受け取る。変数「ａ」は、「ｆ」から「ｓ」を減算することによって復元することができる。

【００６９】

図１９には、加法難読化スキームから乗法難読化スキームに変換する実施形態を示す。このプロセスは、基本的に図１８に示すものとは逆のプロセスである。なお、乱数シードが必要なコンピュータ以上に分散されるのを防ぐために、矢印は、図１８の経路とは逆の経路をたどる。このように、いずれの変換についても、ＴＣ（１８００、１８４０、１９００及び１９４０）及びＣＣ３（１８２０及び１９２０）のみが両シードを知っており、ＣＣ２（１８１０及び１９１０）はいずれのシードも知らず、ＣＣ４（１８３０及び１９３０）は乗法シードのみを知っている。

【００７０】

図１９の実施形態例では、プログラムが、乱数「ｓ」による「ａ」の加法ブラインディングを実行するが、乱数「ｒ」による「ａ」の乗法ブラインディングを必要とする。変換プロセスは、セキュリティのためにシュレッディングされる。最初に、信頼できるコンピュータ１（ＴＣ１）１９００及び信頼できるコンピュータ５（ＴＣ５）１９４０は、「ａ」をブラインディングするための２つの乱数「ｒ」及び「ｓ」を選択する。ＴＣ５　１９４０は、「ａ＋ｓ」を計算して結果を変数「ｃ」に割り当てた後に、これをクラウドコンピュータ４（ＣＣ４）１９３０に送信する。次に、ＣＣ４　１９３０は、「ｃ＊ｒ」を計算することによってこの和をブラインディングし、これを変数「ｅ」に割り当てて「ｅ」をクラウドコンピュータ２（ＣＣ２）１９１０に送信する。ＣＣ２　１９１０は、加法から乗法へのブラインディングをクラウドコンピュータ３（ＣＣ３）１９２０に要求する。ＣＣ３　１９２０は、乱数値「ｒ」及び「ｓ」の知識を与えられ、「ｒ＊ｓ」を計算し、この積を変数「ｈ」に割り当てて「ｈ」をＣＣ２　１９１０に送信する。ＣＣ２　１９１０は、ＣＣ３　１９２０から値「ｈ」を受け取り、「ｅ－ｈ」を計算し、結果を変数「ｆ」に割り当ててこれをＴＣ１　１９００に送信する。最後に、ＴＣ１　１９００は、「ａ＊ｒ」に等しいａの乗法ブラインディング値を受け取る。変数「ａ」は、「ｆ／ｒ」を計算することによって復元することができる。

【００７１】

図１８及び図１９に示すような１つのブラインディングスキームから別のブラインディングスキームへの変換のシュレッディングは、いずれか１つのクラウドコンピュータを制御している攻撃者が「ａ」の平文値を計算できないように使用される。シュレッディング変換中には、ＣＣ４（１８３０及び１９３０）に乗法鍵「ｒ」が公開されるが、このコンピュータにおける攻撃者は、「ａ」の加法ブラインディング値しか分からない。同様に、ＣＣ２（１８１０及び１９１０）における攻撃者は、「ａ」の乗法ブラインディング値は分かるが、乗法ブラインディング鍵「ｒ」にアクセスできないため「ａ」の計算が防がれる。ＣＣ３（１８２０及び１９２０）では両鍵「ｒ」及び「ｓ」が明らかになるが、このコンピュータは積「ｒ＊ｓ」しか供給せず、従ってＣＣ３（１８２０及び１９２０）における攻撃者に「ａ」の加法ブラインディング値又は乗法ブラインディング値が分かってしまうことは決してない。

【００７２】

上記の方法及びプロセス（方法Ｉ）は、他の単一のコンピュータへのルートアクセス権を有し、実行プログラム、メモリ内データ及びディスク上のファイルの記録及び分析を行って機密情報を観察する攻撃者（脅威レベルＩ）に対して効果的である。しかしながら、方法Ｉは、他の単一のコンピュータ上の実行プログラム、メモリ及びファイルを修正して実行を邪魔することができる攻撃者（脅威レベルＩＩ）に対しては効果的でない場合もある。このことを考慮して、上記の方法及びプロセスを、全てのシュレッドが他の複数のコンピュータ上で実行されるように拡張することができる（方法ＩＩ）。他の複数のコンピュータからの中間データ結果を整合性についてチェックし、不整合が検出された場合には

計算を中断する。方法ＩＩは、特定のシュレッドを実行する他の全てのコンピュータの制御を攻撃者が獲得しない限り脅威レベルＩＩの攻撃者からの攻撃を阻止することができる。さらに、ネットワーク内の他の全てのコンピュータを制御する攻撃者に対しては、方法Ｉも方法ＩＩも効果的でない（脅威レベルＩＩＩ）。方法ＩＩは、単一のプログラムを実行するように選択された他のコンピュータが複数の管理領域又は複数の商業的に異なるインフラに及ぶように拡張することができる（方法ＩＩＩ）。例えば、他のコンピュータは、Ｇｏｏｇｌｅ、Ａｍａｚｏｎ及び／又はＭｉｃｒｏｓｏｆｔなどの様々なパブリッククラウドプロバイダから選択することができる。方法ＩＩＩは、あるプログラムのために選択された全ての領域又はインフラの制御を攻撃者が獲得しない限り脅威レベルＩＩＩの攻撃者からの攻撃を阻止することができる。

【００７３】

上記の方法及びプロセスによって提供されるプライバシーは、計算及びデータのシュレッディングに由来する。難読化及び暗号化は、実行待ち時間に多少のオーバヘッドを加えるが、性能悪化の主な原因は、コンピュータ間のネットワークの通信待ち時間に起因する。従って、関連する様々なコンピュータ間で送信されるメッセージの数及びメッセージのサイズを最小化することが重要である。

【００７４】

本文書で引用又は説明した各特許、特許出願及び公表文献の開示は、その全体が引用により本明細書に組み入れられる。

【００７５】

当業者であれば、本発明の好ましい実施形態に数多くの変更及び修正を行うことができ、このような変更及び修正は、本発明の趣旨から逸脱することなく行うことができると理解するであろう。従って、添付の特許請求の範囲は、本発明の真の趣旨及び範囲に含まれる全てのこのような同等の変形例も対象とするように意図される。

【符号の説明】

【００７６】

１００　プログラム
１１０　信頼できるコンピュータ１
１２０　クラウドコンピュータ２
１３０　クラウドコンピュータ３
１４０　信頼できるコンピュータ４

【図１】

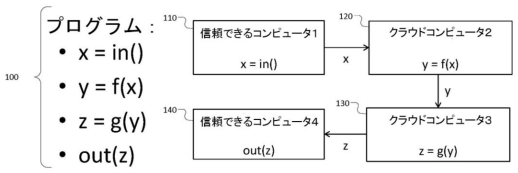


FIG. 1

【図２】

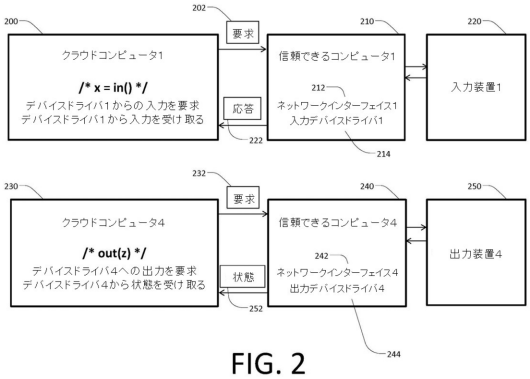


FIG. 2

【図３】

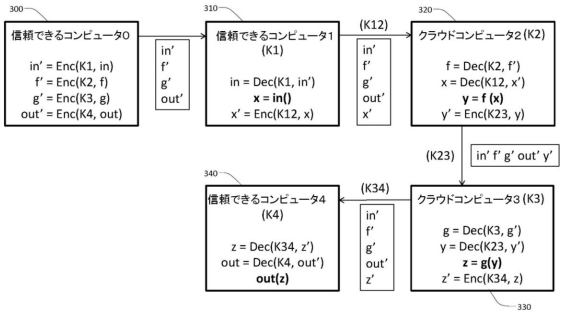


FIG. 3

【図４】

• ワンタイムパッド
• ランダム　r, s, u

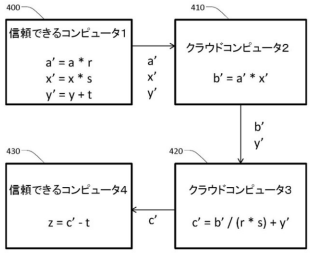


FIG. 4

【図５】

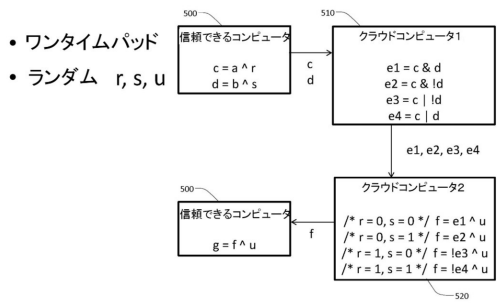


FIG. 5

【図６】

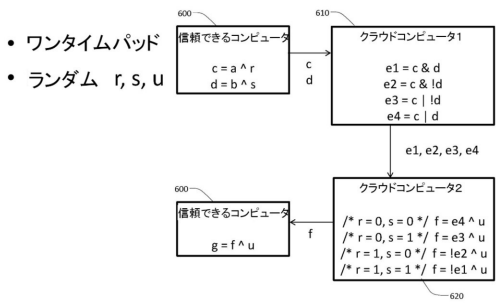


FIG. 6

【図７】

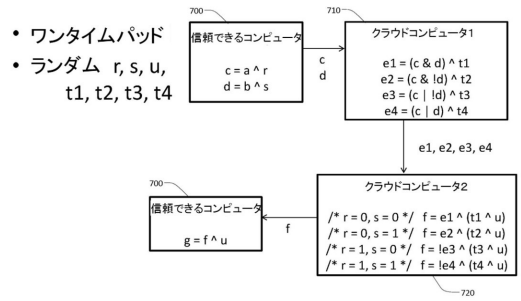


FIG. 7

【図８】

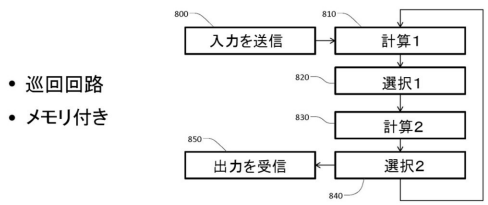


FIG. 8

【図９】

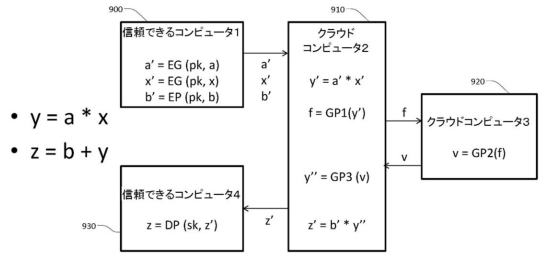


FIG. 9

【図１０】

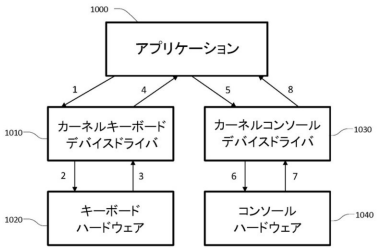


FIG. 10

【図１１】

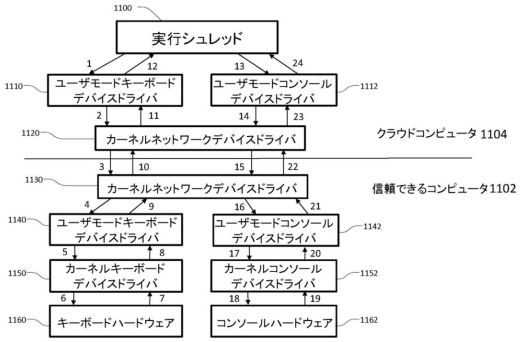


FIG. 11

【図１２】

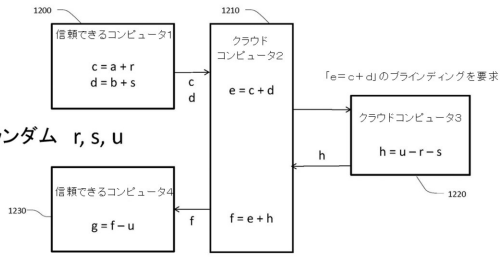


FIG. 12

【図１３】

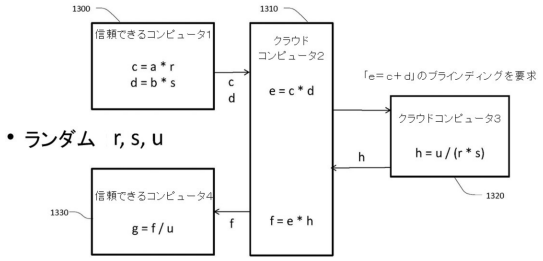


FIG. 13

【図１４】

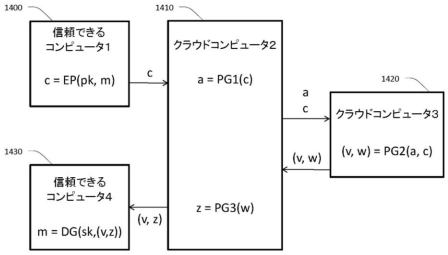


FIG. 14

【図１５】

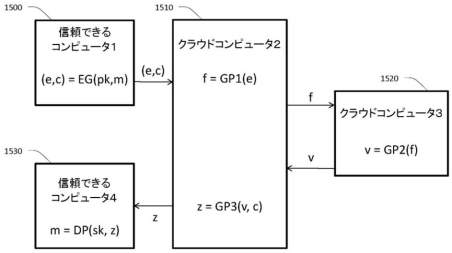


FIG. 15

【図１６】

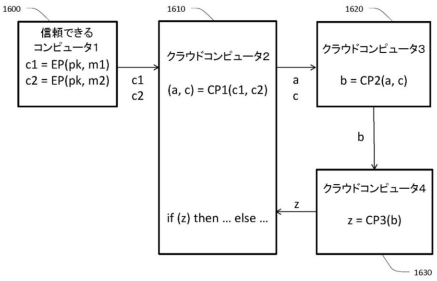


FIG. 16

【図１７】

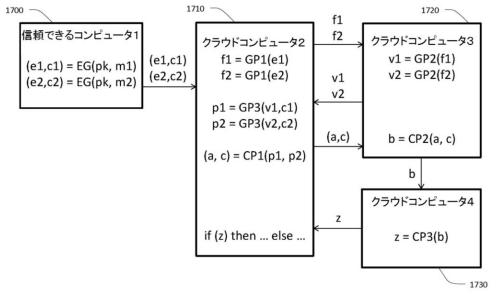


FIG. 17

【図１８】

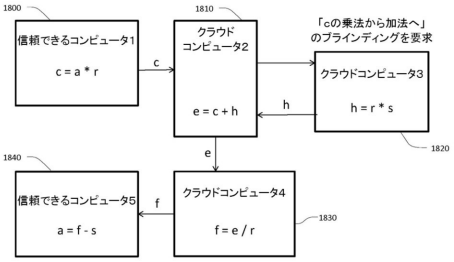


FIG. 18

【図１９】

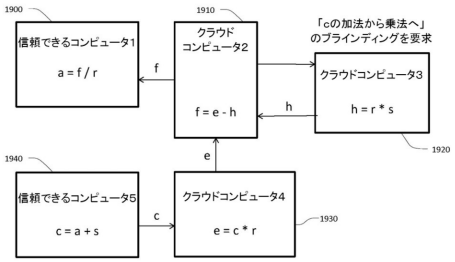


FIG. 19

フロントページの続き

(74)代理人 100109070
弁理士 須田 洋之
(74)代理人 100109335
弁理士 上杉 浩
(74)代理人 100120525
弁理士 近藤 直樹
(74)代理人 100139712
弁理士 那須 威夫
(74)代理人 100176418
弁理士 工藤 嘉晃
(72)発明者 シダナ アシュミート
アメリカ合衆国 カリフォルニア州 ９４３０３ パロ アルト スタンリー ウェイ １１７５
(72)発明者 コルテ プリヤダーシャン
アメリカ合衆国 テキサス州 ７８７５９ オースティン ヴィラ マリア レーン ７２３９
(72)発明者 リン カルヴァン
アメリカ合衆国 テキサス州 ７８７５９ オースティン ローン メサ ８４０８

審査官 金沢 史明

(56)参考文献 米国特許出願公開第２０１２／００６６５１０（ＵＳ，Ａ１）
特開２００８－２８３６７２（ＪＰ，Ａ）
特表２０１５－５０１９４６（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０９Ｃ １／００
Ｈ０４Ｌ ９／１０
Ｇ０６Ｆ ２１／１０－２１／１２