【公報種別】特許法第１７条の２の規定による補正の掲載
【部門区分】第６部門第３区分
【発行日】令和1年10月17日(2019.10.17)

【公表番号】特表2019-502197(P2019-502197A)
【公表日】平成31年1月24日(2019.1.24)
【年通号数】公開・登録公報2019-003
【出願番号】特願2018-526555(P2018-526555)
【国際特許分類】
Ｇ０６Ｆ　21/56　　(2013.01)
【ＦＩ】
Ｇ０６Ｆ　21/56
Ｇ０６Ｆ　21/56　　３４０

【手続補正書】
【提出日】令和1年9月6日(2019.9.6)
【手続補正１】
【補正対象書類名】特許請求の範囲
【補正対象項目名】全文
【補正方法】変更
【補正の内容】
【特許請求の範囲】
【請求項１】
コンピュータ内で実行されるランタイム生成コード内の悪意のあるコードの検出のための方法であって、前記コンピュータのプロセッサ上で、
前記コンピュータのメモリにおけるランタイム生成コードの作成および実行のうちの少なくとも一方のインジケーションを受け取る行為、
前記ランタイム生成コードに関連する、スタティック非ハッシュ化データのシグネチャデータと、前記ランタイム生成コードを作成した、許可されたソース作成モジュールを表す複数のテンプレートの、スタティック非ハッシュ化データのテンプレートシグネチャとの間のマッチを識別する行為であって、前記テンプレートが記憶デバイス上のリポジトリに格納されている、マッチを識別する行為、および、
マッチが見つからないとき、前記ランタイム生成コード内の悪意のあるコードに対処するためにセキュリティプロセスをトリガする行為、
を実行するステップを含む、方法。
【請求項２】
前記テンプレートシグネチャが、許可されたジャストインタイム（ＪＩＴ）コンパイラを表す、請求項１に記載の方法。
【請求項３】
前記シグネチャデータと前記テンプレートシグネチャとの間の前記マッチを識別するステップが、
オペレーティングシステム機能を呼び出すために前記ランタイム生成コードによって呼び出された第１の実行可能なモジュールと、前記許可されたＪＩＴコンパイラを表す前記テンプレートとの間の関連を識別するステップ、および、
前記ランタイム生成コードを作成した第２の実行可能なモジュールと、前記許可されたＪＩＴコンパイラを表す前記テンプレートとの間の関連を識別するステップ
のうちの少なくとも一方を含む、請求項２に記載の方法。
【請求項４】
前記シグネチャデータが、前記ランタイム生成コードを格納する前記メモリにおけるエリアの、スタティック非ハッシュ化され、あらかじめ定義されたサイズを含む、請求項２

に記載の方法。
【請求項５】
前記シグネチャデータが、前記ランタイム生成コードを格納するメモリ領域の、読取り専用またはアクセス禁止としての指定を含む、請求項２に記載の方法。
【請求項６】
前記シグネチャデータが、前記ランタイム生成コード内の少なくとも１つのスタティックコードパターンを含む、請求項２に記載の方法。
【請求項７】
前記少なくとも１つのスタティックコードパターンが、前記ランタイム生成コードの少なくとも１つの関数の開始領域における少なくとも１つのあらかじめ定義されたプロローグ、少なくとも１つのエピローグ、および少なくとも１つのマジックオペランド値からなるグループから選択される少なくとも１つのメンバを含む、請求項６に記載の方法。
【請求項８】
前記シグネチャデータが、前記ランタイム生成コードの開始領域および終了領域のうちの少なくとも一方に、前記ＪＩＴコンパイラに関係するあらかじめ定義された制御構造を含む、請求項２に記載の方法。
【請求項９】
前記あらかじめ定義された制御構造が、前記ランタイム生成コードの一部分を各々格納する複数の異なるメモリ領域の各々にあるリンクリスト、ならびに前記それぞれのリンクリストの後に位置する前記それぞれのメモリ領域のサイズおよびアドレスを定義するフィールドのうちの少なくとも一方を含む、請求項８に記載の方法。
【請求項１０】
前記リンクリストが、各メモリ領域のポインタをトラバースすることによって検証され、前記フィールドが、前記フィールドの前記値をオペレーティングシステム値と相関させることによって検証される、請求項９に記載の方法。
【請求項１１】
前記シグネチャデータが、前記許可されたＪＩＴコンパイラが制限される前記ランタイム生成コードに関連するアプリケーションを含む、請求項２に記載の方法。
【請求項１２】
前記テンプレートシグネチャが、許可されたフックエンジンを表す、請求項１に記載の方法。
【請求項１３】
前記シグネチャデータが、前記ランタイム生成コードがフックエンジンによって作成されるという識別を含み、前記識別が、
フックされたモジュールの外にある外部コードに到達するために、前記フックされたモジュールのプロローグにおける既存のコードをエミュレートすること、および
前記フックをインストールした前記許可されたフックエンジン実行ファイルの前にスタックトレースに現れる前記ランタイム生成コードの位置を特定することによって前記ランタイム生成コードを識別するために、前記外部コードに関係する前記スタックトレースを分析すること、
のうちの少なくとも一方によって行われる、請求項１２に記載の方法。
【請求項１４】
前記シグネチャデータが、前記ランタイム生成コードがある前記メモリエリアのあらかじめ定義されたサイズ、少なくとも１つのコードパターン、前記ランタイム生成コードメモリ領域の開始部分および終了部分のうちの少なくとも一方におけるあらかじめ定義された制御構造、ならびに可変パラメータを除いて前記ランタイム生成コードに逆アセンブルプログラムを適用することによって取得されたアセンブリから計算されたオペコードシグネチャからなるグループから選択される少なくとも１つのメンバを含む、請求項１２に記載の方法。
【請求項１５】

前記少なくとも１つのコードパターンが、前記ランタイム生成コードの少なくとも１つの関数の開始領域における少なくとも１つのあらかじめ定義されたプロローグ、少なくとも１つのエピローグ、および少なくとも１つのマジックオペランド値からなるグループから選択される少なくとも１つのメンバを含む、請求項１４に記載の方法。

【請求項１６】

前記テンプレートシグネチャが、許可された実行可能なコンプレッサを表し、前記ランタイム生成コードは解凍されたプログラムを含む、請求項１に記載の方法。

【請求項１７】

前記シグネチャデータが、前記解凍された実行可能ファイルのフォーマットに従ったメモリ割振りのサイズ、前記実行可能ファイル構造およびコードの不変部分について計算された暗号学的ハッシュ関数、ならびに前記解凍されたプログラムがあるメモリページ上のパーミッションからなるグループから選択される少なくとも１つのメンバを含む、請求項１６に記載の方法。

【請求項１８】

前記解凍された実行可能ファイルの前記フォーマットに従って前記メモリ割振りのコンテンツをパースすることによって、前記メモリ割振りのベースでの前記メモリのコンテンツが前記解凍されたプログラムの前記フォーマットに従ったものであることを検証するステップと、フィールド値が論理的であり、前記フォーマットに従っていることをチェックするステップとをさらに含む、請求項１７に記載の方法。

【請求項１９】

悪意のあるコードを含むランタイム生成コードの検出のためのシステムであって、

コードを格納するためのメモリと、

ランタイム生成コードを作成する、許可（authorize）されたソース作成モジュールを表すテンプレートのリポジトリを格納するための記憶デバイスと、

コードを格納するプログラム記憶装置と、

前記格納されたコードを実行するために、前記メモリ、前記記憶デバイス、および前記プログラム記憶装置に結合されたプロセッサと、

を含み、

前記格納されたコードが、

前記メモリにおけるランタイム生成コードの前記作成および前記実行のうちの少なくとも一方のインジケーションを受け取り、前記ランタイム生成コードに関連する、スタティック非ハッシュ化データのシグネチャデータと前記リポジトリの、スタティック非ハッシュ化データのテンプレートシグネチャとの間のマッチを識別し、マッチが見つからないとき、前記ランタイム生成コード内の悪意のあるコードに対処するためにセキュリティプロセスをトリガするための格納されたコードを含む、

システム。

【請求項２０】

悪意のあるコードを含むランタイム生成コードの検出のためのシステムのプロセッサによって実行されるようにプログラムコードを格納した非一時的コンピュータ可読記憶媒体を含むコンピュータプログラム製品であって、前記プログラムコードが、

コンピュータのメモリにおけるランタイム生成コードの作成および実行のうちの少なくとも一方のインジケーションを受け取るための命令、

前記ランタイム生成コードに関連する、スタティック非ハッシュ化データのシグネチャデータと、ランタイム生成コードを作成する、許可（authorize）されたソース作成モジュールを表すテンプレートのセットの、スタティック非ハッシュ化データのテンプレートシグネチャとの間のマッチを識別するための命令、および、

マッチが見つからないとき、前記ランタイム生成コード内の悪意のあるコードに対処するためにセキュリティプロセスをトリガするための命令、

を含む、コンピュータプログラム製品。