

DEVICE-CODE RISK EARLY WARNING METHOD, SOFTWARE SYSTEM, AND COMPUTING DEVICE

TECHNICAL FIELD

[0001] The present invention relates to the field of code risk early warning, and in particular, to a device-code risk early warning method, a software system, and a computing device.

BACKGROUND

[0002] When performing vulnerability detection on an online website server, existing technologies need to establish a network connection to the online server, obtain source codes of a front-end web system on the online server, and compare the obtained source codes with source codes of a patch code to determine whether known or newly discovered vulnerabilities exist. The online website server (online server) itself is also a hardware device, and its operating environment executes a large number of binary codes (for example, Linux Kernel, Web Server, and the like).

[0003] When performing vulnerability detection on a back-end web system on an online server and an online hardware device, since intermediate codes, binary codes on the server, and binary codes in the hardware device are all codes unrecognizable by humans, and the server and the hardware device are already running online, the current existing technology is to send a network packet simulating an attack to the server and the hardware device via a network according to a behavior description of a specific vulnerability. After the server and the hardware device receive the network packet, whether responses of the server and the hardware device conform to the behavior description of the specific vulnerability is monitored. When they conform, it indicates that the specific vulnerability is present in the online server and the online hardware device, thereby providing a security early warning function for operators of the server and the online hardware device.

[0004] When analyzing vulnerabilities of the online server and/or the hardware device and detecting and preventing risks, since codes running in the server and/or the hardware device are mainly intermediate codes or binary codes, in the absence of source codes, it is impossible to compare source codes of patches with source codes of an online system; source code comparison

cannot be implemented without source codes. Vulnerability analysis using simulated packets requires support of a network environment. In the case of offline or abnormal network operation, simulated packets cannot be sent to the online server and/or the hardware device to be tested, and vulnerability analysis and risk monitoring using simulated packets cannot be implemented either. Moreover, when a specific vulnerability is not triggered by a simulated packet, the specific vulnerability cannot be detected by means of a simulated packet.

[0005] It can be seen based on the above description that, in the technical field of information security risk prevention and control, how to perform vulnerability analysis and security early warning on a hardware device lacking source codes and/or running online is an important technical issue. At present, there is no device-code risk prevention and control method, software system, or computing device that can provide a risk prevention and control method for vulnerability analysis and security early warning for a device and a system applying the method when source codes of the device system are lacking and/or the network is offline, so development is still urgently needed.

SUMMARY

[0006] The present invention provides a device-code risk early warning method, a software system, and a computing device, which are implemented through the following technical solutions:

[0007] Solution 1:

[0008] A device-code risk early warning method, including the following steps:

[0009] collecting an update request for a device-code, obtaining the device-code according to content of the update request, and transmitting the device-code to a device, where the device-code includes at least one software code;

[0010] obtaining a risk code when a code analysis event corresponding to the risk code is triggered, and comparing component contents of the risk code and the at least one software code; and

[0011] sending a risk early warning notification when a result of the comparison indicates that the at least one software code contains at least one security risk.

[0012] Further, sending the risk early warning notification refers to sending the risk early warning notification to at least one of a provider of the device-code, an operator of the device,

and a manager via at least one network communication channel.

[0013] Further, the device-code is provided by a first user, the collection of the update request refers to receiving for release and/or update of the device-code, and the device is operated by a second user.

[0014] Further, the at least one software code includes at least one binary code.

[0015] Further, obtaining the device-code specifically refers to obtaining the device-code from a storage medium, including: obtaining the device-code from the storage medium according to code version information, where the device-code is specified by the code version information.

[0016] Further, the code version information is directly specified in the content of the update request, or the code version information required for updating the device is determined according to a product specification, a hardware specification, and/or a firmware specification contained in the content of the update request.

[0017] Further, the update request is an online update request transmitted via a network.

[0018] Further, the content of the update request includes network location information of the device, where the device-code is transmitted to the device via the network according to the network location information.

[0019] Further, comparing the component contents of the risk code and the at least one software code includes:

[0020] extracting at least one first component feature corresponding to the risk code;

[0021] extracting at least one second component feature corresponding to the at least one software code; and

[0022] comparing the at least one first component feature with the at least one second component feature.

[0023] Further, the first component feature is obtained by analyzing code composition content and/or a code execution result corresponding to the risk code.

[0024] Further, the code composition content includes the risk code and/or a first intermediate code obtained by compiling the risk code, where the code execution result includes semantic information revealed during simulation execution of the risk code and/or the first intermediate code when performing simulation execution on the risk code and/or the first intermediate code.

[0025] Further, the at least one second component feature is obtained by analyzing at least one code composition content and/or at least one code execution result corresponding to the at least one software code.

[0026] Further, the at least one code composition content includes at least one binary code and/or at least one second intermediate code obtained by disassembling the at least one binary code, where the at least one code execution result includes semantic information revealed during simulation execution of the at least one binary code and/or the at least one second intermediate code when performing simulation execution on the at least one binary code and/or the at least one second intermediate code.

[0027] Further, a programming language structure of the intermediate code is declared by an intermediate language, an intermediate representation, a structured language, and/or a structured protocol.

[0028] Further, types of the risk code include a patch code, a malicious program code, and/or any code used for correcting or generating a software vulnerability or a software virus.

[0029] Further, the code analysis event corresponding to the risk code is triggered when the risk code is added or updated.

[0030] Further, before the code analysis event is triggered, the at least one code analysis event is detected according to a composition list of the at least one software code, where the code analysis event corresponding to the risk code is one of the at least one code analysis event.

[0031] Further, the composition list is obtained by scanning and/or analyzing a component composition of the device-code when the device-code is collected.

[0032] Further, the composition list includes metadata of the device-code itself and/or metadata of the at least one software code contained in the component composition of the device-code.

[0033] Further, the at least one software code is present in the component composition of the device-code in forms including a system core, a driver program, an application program, a component, and/or a class library.

[0034] Further, when the device-code is a system image file, the component composition contained in the system image file is scanned and/or analyzed to identify the at least one software code present in the system image file, so as to obtain the composition list of the at least one software code; or

[0035] when the device-code is a software code package, the component composition contained in the software code package is scanned and/or analyzed to identify the at least one software code present in the software code package, so as to obtain the composition list of the at least one software code.

[0036] Further, when the risk early warning notification carries an instruction, the instruction is used to instruct the device and/or a user operating the device to perform an update operation of the risk code on the device; or, when the risk early warning notification carries prompt information, the prompt information is used to provide an early warning of the at least one security risk for the provider of the device-code, the operator of the device, and/or the manager.

[0037] Solution 2:

[0038] A device-code risk early warning software system, including a collection unit, a code acquisition unit, a comparison unit, and a transmission unit; where

[0039] the collection unit is configured to receive an update request for a device-code;

[0040] the code acquisition unit is configured to obtain the device-code, where the device-code includes at least one software code;

[0041] the transmission unit is configured to transmit the device-code to a device and send a risk early warning notification; and

[0042] the comparison unit is configured to, when a code analysis event corresponding to a risk code is triggered, enable the code acquisition unit to obtain the risk code, and compare the risk code with the device-code.

[0043] Further, the comparison unit includes a first feature extraction unit, a second feature extraction unit, and a feature comparison unit; where

[0044] the first feature extraction unit is configured to extract at least one first component feature corresponding to the risk code;

[0045] the second feature extraction unit is configured to extract at least one second component feature corresponding to the at least one software code; and

[0046] the feature comparison unit is configured to compare the at least one first component feature with the at least one second component feature.

[0047] Further, the first feature extraction unit or the second feature extraction unit

includes at least one analysis unit and at least one retrieval unit; where

[0048] the at least one analysis unit is configured to obtain a first component feature by analyzing code composition content and/or a code execution result corresponding to the risk code, and/or obtain a second component feature by analyzing code composition content and/or a code execution result corresponding to the at least one software code; and

[0049] the at least one retrieval unit is configured to extract the first component feature and/or the second component feature obtained after analysis.

[0050] Further, the software system further includes at least one detection unit; where

[0051] the at least one detection unit is configured to monitor a risk code login list corresponding to each software code contained in the composition list according to the composition list of the at least one software code, so as to detect a code analysis event by monitoring abnormal changes in the risk code login list.

[0052] Solution 3:

[0053] A device-code risk early warning computing device, including a collection unit, a code acquisition unit, a comparison unit, and a transmission unit; where

[0054] the collection unit is configured to receive an update request for a device-code;

[0055] the code acquisition unit is configured to obtain the device-code from a storage medium, where the device-code includes at least one software code;

[0056] the transmission unit is configured to transmit the device-code to a device and send a risk early warning notification; and

[0057] the comparison unit is configured to, when a code analysis event corresponding to a risk code is triggered, enable the code acquisition unit to obtain the risk code from the storage medium and/or another storage medium, and compare the risk code with the device-code.

[0058] Such designs have the following advantages: by updating a device-code containing at least one software code for an online device, comparing component contents of a risk code and the device-code, and sending a risk early warning for the device, the present invention can perform risk early warning for the device regardless of whether the device is offline or the network status is normal.

BRIEF DESCRIPTION OF THE DRAWINGS

[0059] FIG. 1 is a schematic flow diagram of a device-code risk early warning method of the present invention.

[0060] FIG. 2 is a schematic structural diagram of a device-code risk early warning software system of the present invention.

[0061] FIG. 3 is a schematic flow diagram of comparing component contents of a risk code and at least one software code of the present invention.

[0062] FIG. 4 is a schematic flow diagram of a device-code risk early warning method in which a software system detects based on a composition list of at least one software code before a code analysis event is triggered.

[0063] FIG. 5 is a schematic structural diagram of a device-code risk early warning computing device of the present invention.

[0064] Reference numerals:

[0065] 1. Software system; 2. Collection unit; 3. Code acquisition unit; 4. Comparison unit; 5. Transmission unit; 6. First user; 7. Second user; 8. Device; 9. Storage medium; 80. Device-code; 11. Computing device; 21. Collection unit; 31. Code acquisition unit; 41. Comparison unit; 51. Transmission unit.

DETAILED DESCRIPTION

[0066] The present invention can be more fully understood through the following description, including the following glossary of terms and concluding examples. For the sake of brevity, publications cited in this specification, including the disclosure content of patents, are incorporated herein by reference.

[0067] The following provides detailed descriptions of embodiments. The embodiments of the present invention are non-limiting and merely represent various aspects and features of the present invention. The embodiments are only for illustrative purposes and do not limit the scope of protection intended by the present invention. In addition, unnecessary components or components that can be completed by conventional technologies are omitted from the drawings in the embodiments to clearly show the technical characteristics of the present invention.

[0068] In the most restrictive technical sense, the code composition content of the

“software code” described herein includes any binary code for loading and execution, intermediate code, and/or any computer program code written in a computer programming language. The “device-code” described herein refers to the code that runs on a device and provides logical operations and services for the device. The device-code includes binary code, intermediate code, and/or any computer program code written in a computer programming language, that is, the device-code includes at least one software code. The at least one software code is present in the component composition of the device-code in the form of components including a system core, a driver program, an application program, a component, and/or a class library. The “binary code” described herein may include any code in binary representation format. The “intermediate code” described herein may include any code represented by a structure declared by an intermediate language, an intermediate representation, a structured language, and/or a structured protocol; for example, Java Bytecode represented in accordance with the format defined by an intermediate representation, and LLVM bitcode obtained by disassembling binary code, are both types of intermediate code and can also be regarded as intermediate code in abbreviated form. The code composition content of the “risk code” described herein includes any binary code for loading and execution, intermediate code, and/or any computer program code written in a computer programming language. The types of the risk code include a patch code, a malicious program code, and/or any code used to correct or generate a software vulnerability or a software virus. The “metadata” described herein, also known as interpretive data, intermediary data, or metadata, refers to data describing an object or information related to the object. The “risk code login list” described herein may include any list logged, registered, and/or represented by a structure declared by a structured language and/or a structured protocol. The list may be stored and published in the form of a file, a database, and/or a web page. The content of the list includes login information of the risk code described herein, where the login information includes the content of the risk code itself, metadata of the risk code, software vulnerability information or software virus information corresponding to the risk code; where the CVE (Common Vulnerabilities and Exposures) list (CVE Repository) is an implementation form of the risk code login list described herein, but is not limited thereto. The “semantic information” described herein may include values retrieved from memory and/or intermediate variables during the simulation execution of the code, the execution history of each instruction during execution, output/input values of functions, and/or any retrievable numerical information. All or part of the semantic information of two codes can be used as component features for comparison between the two, and the similarity between the two codes is determined according to the degree of consistency in the comparison.

[0069] In addition, the “collection unit”, “code acquisition unit”, “transmission unit”, “comparison unit”, “first feature extraction unit”, “second feature extraction unit”, “feature comparison unit”, “analysis unit”, “retrieval unit”, and “detection unit” described herein are processing units that provide logical operations or services. The implementation forms of the processing units may include a class library, a software component, a software agent, an application program, any program block that can be built-in, installed, introduced, or extended to run in the “software system or computing device for providing device-code risk early warning” described herein, or any integrated circuit unit designed in a hardware description language and can be embedded in the “computing device for providing device-code risk early warning” described herein.

[0070] The “software system for providing device-code risk early warning” described herein may be implemented in forms including a system firmware, an application program, a software engine, an operating system, or any software platform that can be extended by programs. In a preferred embodiment, the “software system for providing device-code risk early warning” described herein can be deployed to run on the same machine, that is, the collection unit, code acquisition unit, transmission unit, comparison unit, first feature extraction unit, second feature extraction unit, feature comparison unit, analysis unit, retrieval unit, and detection unit described herein are all integrated into the same software system, and the software system is deployed to run on a machine. In another preferred embodiment, the “software system for providing device-code risk early warning” described herein can be implemented through the collaboration of multiple machines, that is, the processing units such as the collection unit, code acquisition unit, transmission unit, comparison unit, first feature extraction unit, second feature extraction unit, feature comparison unit, analysis unit, retrieval unit, and detection unit described herein are provided to different machines. The processing units in different machines collaborate and interact with each other through connection approaches including the Internet, WiFi, Bluetooth, etc., and are further jointly integrated into a software system for providing device-code risk early warning through multiple machines.

[0071] The “computing device for providing device-code risk early warning” described herein may be implemented in forms including a network communication device, a server, or any information and communication device with networking, data processing, or control capabilities. In a preferred embodiment, the “computing device for providing device-code risk early warning” described herein can be implemented by a single machine, that is, the collection unit, code acquisition unit, transmission unit, comparison unit, first feature extraction unit, second feature

extraction unit, feature comparison unit, analysis unit, retrieval unit, and detection unit described herein are all integrated into the same machine as the computing device for providing device-code risk early warning. In another preferred embodiment, the “computing device for providing device-code risk early warning” described herein can be implemented through the collaboration of multiple machines, that is, the processing units such as the collection unit, code acquisition unit, transmission unit, comparison unit, first feature extraction unit, second feature extraction unit, feature comparison unit, analysis unit, retrieval unit, and detection unit described herein are provided to different machines. The processing units in different machines can collaborate and interact with each other through connection approaches including the Internet, WiFi, Bluetooth, etc., and are further jointly integrated into a computing device for providing device-code risk early warning through multiple machines.

[0072] As mentioned above, in a preferred embodiment, the “collection unit”, “code acquisition unit”, “transmission unit”, “comparison unit”, “first feature extraction unit”, “second feature extraction unit”, “feature comparison unit”, “analysis unit”, “retrieval unit”, and “detection unit” described herein can be individually implemented in different processing units. In another preferred embodiment, at least two of the “collection unit”, “code acquisition unit”, “transmission unit”, “comparison unit”, “first feature extraction unit”, “second feature extraction unit”, “feature comparison unit”, “analysis unit”, “retrieval unit”, and “detection unit” described herein can also be implemented in the same processing unit.

[0073] The content of the present invention will be further described in detail below with reference to the accompanying drawings and specific implementations.

[0074] As shown in FIG. 1, which is a preferred flow diagram of a device-code risk early warning computing of the present invention. The device-code 80 risk early warning method includes the following steps:

[0075] S1. Collecting an update request for a device-code 80, obtaining the device-code 80 from a storage medium 9 according to the content of the update request, and transmitting the device-code 80 to a device; where the device-code 80 includes at least one software code;

[0076] S2. Obtaining, by the software system 1 when a code analysis event corresponding to a risk code is triggered, the risk code from the storage medium 9 and/or another storage medium 9, and comparing the component contents of the risk code and the at least one software code; and

[0077] S3. Sending a risk early warning notification when the comparison result

indicates that the at least one software code contains at least one security risk.

[0078] As shown in FIG. 2, which is a schematic structural diagram of a device-code 80 risk early warning software system 1 of the present invention, providing an explanation of a preferred embodiment of the device-code risk early warning computing of the present application. The first user 6 is a user of the software system 1, and the second user 7 is an operator or manager of the device 8; the collection unit 2, the code acquisition unit 3, the comparison unit 4, and the transmission unit 5 are processing units integrated into the software system 1; the storage medium 9 is a memory, an external storage device, a database, or any storage device that can store the device-code 80 or the risk code; the device-code 80 is code that runs on the device 8 and provides logical computing and services for the device 8 on the device 8.

[0079] In a preferred embodiment, the second user 7 is a user of the software system 1, and manages the device 8 through the software system 1 by registering metadata of the device 8 with the software system 1. The metadata of the device 8 includes network location information, product specifications, hardware specifications, and/or firmware specifications of the device 8. In another preferred embodiment, the second user 7 is only an operator of the device 8, and the metadata of the device 8 is registered with the software system 1 by the system administrator of the software system 1, and the system administrator of the software system 1 acts as the manager of the device 8.

[0080] In a preferred embodiment, the first user 6 is a developer of the device 8, and the first user 6 provides the device-code 80 to the software system 1 through a human-machine operation interface provided by the software system 1. When the first user 6 provides the device-code 80 to the software system 1, the first user 6 can input relevant data of the device-code 80 on the human-machine operation interface, and provide the relevant data together with the device-code 80 to the software system 1 through the human-machine operation interface; where the relevant data includes code version information of the device-code 80, as well as product specifications, hardware specifications, and/or firmware specifications of the device 8 for which the device-code 80 provides update support.

[0081] In a preferred embodiment, the software system 1 stores the device-code 80 and its relevant data (including the code version information of the device-code 80, as well as product specifications, hardware specifications, and/or firmware specifications of the device 8 for which the device-code 80 provides update support) in the storage medium 9, to complete the

login of the device-code 80 in the software system 1.

[0082] In a preferred embodiment, the collection unit 2, the code acquisition unit 3, the comparison unit 4, and the transmission unit 5 are regarded as processing units integrated into the software system 1, and each processing unit can provide at least one function to realize interaction with other processing units. When one processing unit needs to send a request to another processing unit, the content of the request can be used as a parameter, and the content of the request can be input to the other processing unit in the form of a function call. After receiving the content of the request, the other processing unit can provide services for the request according to the content of the request. In another preferred embodiment, each processing unit can also realize interaction with other processing units through network connection, Inter-process communication, or other approaches. When one processing unit needs to send a request to another processing unit, the processing unit can establish a TCP/UDP Socket connection to the other processing unit, transmit the content of the request to the other processing unit through a mutually agreed communication protocol, or send the content of the request to the other processing unit through an inter-process communication mechanism implemented by partial memory sharing, so that the other processing unit can receive the content of the request and provide services for the request according to the content of the request; where the implementation approach of how to provide a request to a processing unit and how the processing unit receives the request is not limited thereto, and those skilled in the art can make any equivalent changes and designs according to actual application requirements.

[0083] After the software system 1 completes the login of the device-code 80, in a preferred embodiment, the software system 1 can search for devices 8 registered in the software system 1 based on the product specifications, hardware specifications, and/or firmware specifications supported by the device-code 80. When the product specifications, hardware specifications, and/or firmware specifications of the device 8 match the specifications supported by the device-code 80, the software system 1 can generate an update request, which is for updating the device-code 80, for the device 8 and provide the update request to the collection unit 2. In another preferred embodiment, when the first user 6 provides the device-code 80 to the software system 1 through the human-machine operation interface of the software system 1, the update request for the device 8 can be directly input into the human-machine operation interface, and the update request can be directly provided to the software system 1. After receiving the update request, the software system 1 can provide the update request to the collection unit 2. In another preferred embodiment, the collection unit 2 can receive requests through network

connection. In this case, the first user 6 can establish a TCP/UDP Socket connection to the collection unit 2 through any other software system 1, and transmit and provide the update request for the device 8 to the collection unit 2 in the form of an online update request through the network connection. In another preferred embodiment, the software system 1 sends a notification to the device 8 through a network connection, notifying the device 8 that a new device-code 80 has been logged in the software system 1, and the code version information of the device-code 80 is included in the content of the notification. After receiving the notification, the device 8 can generate an update request containing the code version information according to the content of the notification, establish a network connection to the collection unit 2 in the form of an online update request, and provide the online update request to the collection unit 2 through network transmission. In another preferred embodiment, the software system 1 does not need to include the code version information in the content of the notification to the device 8. After receiving the notification, the device 8 can generate an update request. The content of the update request includes the product specifications, hardware specifications, and/or firmware specifications of the device 8, and the update request is transmitted to the software system 1 through a network connection. After receiving the update request, the software system 1 can determine the code version information of the device-code 80 that matches the product specifications, hardware specifications, and/or firmware specifications according to the content of the update request, add the code version information to the content of the update request, and then provide the update request to the collection unit 2.

[0084] In a preferred embodiment, after the software system 1 completes the login of the device-code 80, it does not need to send any notification to the device 8 at all. The device 8 operated by the second user 7 periodically sends online update requests to the software system 1 through network connection. The content of the update request includes the product specifications, hardware specifications, and/or firmware specifications of the device 8, and is transmitted to the software system 1 through a network connection. After receiving the update request, the software system 1 can determine the code version information of the device-code 80 that matches the product specifications, hardware specifications, and/or firmware specifications by comparing the product specifications, hardware specifications, and/or firmware specifications according to the content of the update request. When the device-code 80 is the device-code 80 that the device 8 needs to update, the software system 1 can add the code version information to the content of the update request and provide the update request to the collection unit 2. In another preferred embodiment, the second user 7, as the operator of the device 8, can also

generate an update request for the device 8 through another software system 1. The content of the update request includes the product specifications, hardware specifications, and/or firmware specifications of the device 8, and the update request is transmitted to the software system 1 through a network connection.

[0085] Based on the above description, the generation and transmission of the content of the update request, and the approach for the collection unit 2 to collect the update request are not limited thereto, and those skilled in the art can make any equivalent changes and designs according to actual application requirements.

[0086] Further, the device-code 80 is provided by the first user 6. The login of the device-code 80 in the software system 1 may be for the device-code 80 released for a new device 8, for the device-code 80 updated for an existing device 8, or for the device-code 80 updated and released for an existing device 8 just registered in the software system 1. The at least one software code included in the device-code 80 includes at least one binary code.

[0087] In a preferred embodiment, the device 8 is operated by the second user 7. The device 8 is an embedded system device, a server device, an Internet of Things sensing device, a network communication device, or any computer device with networking capabilities.

[0088] In a preferred embodiment, the content of the update request includes the network location information of the device 8. The network location information is filled into the update request by the software system 1 according to the metadata of the device 8 registered in the software system 1; or the update request is generated by the device 8, and the device 8 fills its own network location information when generating the update request.

[0089] In a preferred embodiment, after the collection unit 2 collects an update request for the device-code 80, a request can be sent to the code acquisition unit 3, to requesting the code acquisition unit 3 to obtain the device-code 80 matching the code version information from the storage medium 9 according to the code version information in the content of the update request, that is, the device-code 80 specified by the code version information. After obtaining the device-code 80, a request can be sent to the transmission unit 5, to request the transmission unit 5 to establish a TCP/UDP Socket connection to the device 8 according to the network location information of the device 8 included in the content of the update request, and transmit the device-code 80 to the device 8 through the Socket connection and a mutually agreed communication protocol, so as to complete the update of the device-code 80 on the device 8 and make the device-code 80 the code executed by the device 8 during online operation.

[0090] The device-code risk early warning computing enables the first user 6 to update the device-code 80 of the device 8 operated by the second user 7 through the software system 1, making the device-code 80 the code executed by the device 8 during online operation; when the device 8 is offline, the software system 1 implementing the device-code risk early warning computing can perform risk early warning of the device-code 80 for the device 8. With the above method, when the device 8 is online, the software system 1 can also perform risk early warning of the device-code 80 for the device 8, that is, both online and offline situations are included in the device-code risk early warning computing.

[0091] The software system 1 in S1 obtains the device-code 80 from the storage medium 9 according to the code version information corresponding to the device-code 80;

[0092] Further, the code version information is directly specified in the content of the update request, or the code version information required for updating the device 8 is determined by the software system 1 according to a product specification, a hardware specification, and/or a firmware specification included in the content of the update request.

[0093] As shown in FIG. 2, when the update request is sent by the first user 6, obviously the first user 6 can directly fill in the version information of the device-code 80 in the update request. After receiving the update request, the software system 1 can directly obtain the device-code 80 from the storage medium 9 according to the version information in the content of the update request. However, when the update request is sent by the second user 7 or the device 8, the second user 7 and the device 8 will not know the latest version information of the device-code 80 provided by the first user 6. The second user 7 only knows the device 8 operated by the second user 7, as well as the product specifications, hardware specifications, and/or firmware specifications of the operated device 8.

[0094] Therefore, when the update request sent by the second user 7 and the device 8 includes the product specifications, hardware specifications, and/or firmware specifications of the device 8 itself but does not include the latest version information of the device-code 80, after the update request from the second user 7 and the device 8 is received by the software system 1, the software system 1 can determine which version of the device-code 80 the device 8 is using based on the product specifications, hardware specifications, and/or firmware specifications sent by the second user 7 and the device 8. When the product specifications, hardware specifications, and/or firmware specifications supported by the device-code 80 match the product specifications, hardware specifications, and/or firmware specifications in the update request, the device-code 80

can be determined as the device-code 80 needed by the second user 7 and the device 8.

[0095] In a preferred embodiment, to enable the software system 1 to realize the comparison of different versions of the device-code 80 and determination of which one to adopt, the product specifications, hardware specifications, and/or firmware specifications supported by different versions of the device-code 80 can be recorded in the storage medium 9 or other databases or media accessible to the software system 1 when the device-code 80 is logged. In this way, the software system 1 can compare different versions of the device-code 80, determine the version information suitable for the device 8 of the second user 7, and then retrieve the device-code 80 specified by the version information.

[0096] The update request in S1 is an online update request transmitted via a network.

[0097] In a preferred embodiment, the online update request transmitted via a network is also referred to as an OTA online update of code in the traditional sense. Basically, the device 8 connects to a server, obtains the latest code from the server, and then performs the update. The server and the software system 1 running on the server do not need to know the network location information of the device 8. That is, the device 8 establishes a network connection to the software system 1 providing network access function according to the network location information of the software system 1, and sends an update request to the software system 1. Subsequently, the device-code 80 is directly obtained from the software system 1 via the same network connection. Alternatively, the device 8 establishes a network connection to the collection unit 2 providing network connection and access function according to the network location information of the software system 1, and after sending an update request to the collection unit 2, establishes another network connection to the transmission unit 5, and obtains the device-code 80 from the transmission unit 5 via the other network connection. The approach of implementing OTA online update of code and obtaining code through a network connection established by the device 8 is not limited thereto, and those skilled in the art can make any equivalent changes and designs according to actual application requirements.

[0098] For the above OTA online update approach of code, the device 8 must know the website location information of the server. Therefore, the software system 1 of the present application also provides the following improved approach for the OTA update of code where the online update request is transmitted via a network:

[0099] The content of the update request includes the network location information of the device 8 operated by the second user 7, and the software system 1 transmits the device-code 80

to the device 8 operated by the second user 7 via the network according to the network location information.

[0100] This improved approach can be used in the following scenarios:

[0101] The device 8 does not need to know the network location information of the software system 1 and its server. The software system 1 completes the update of the device-code 80 for the device 8 in a push manner. In a preferred embodiment, the network location information of the device 8 is written into the update request by the software system 1 according to the metadata of the device 8 registered in the software system 1 after the software system 1 receives the update request sent by the first user 6 or the second user 7, or after the update request is actively generated by the software system 1 when a new device-code 80 is logged; and the transmission unit 5 establishes a network connection to the device 8 according to the network location information in the update request and transmits the device-code 80 to the device 8 for update. In this way of completing the update of the device-code 80 in an asynchronous manner, the sending of the update request and the receiving of the device-code 80 are completed via two connections with different connection objects, which is different from the traditional way where the device 8 establishes a connection to the software system 1 and then downloads via the connection between the device 8 and the software system 1.

[0102] In fact, the above scenario makes the update of the device-code 80 unnecessary for the device 8 itself to download. The first user 6, the second user 7, or other users with permission can send an update request to the software system 1 through the above improved approach. Subsequently, the software system 1 establishes a connection to the device 8 indicated by the network location information according to the network location information of the device 8, and then actively transmits the device-code 80 to the device 8 for update. In a preferred embodiment, the content of the update request does not necessarily include the network location information of the device 8. The software system 1 queries the metadata of the device 8 registered in the software system 1 by querying a database, obtains the network location information of the device 8 from the metadata, and then can send a request to the transmission unit 5, so as to request the transmission unit 5 to transmit the device-code 80 to the device 8 according to the network location information obtained from the metadata of the device 8.

[0103] The network location information can be simply an IP Address, and there is a default communication protocol (a Port supported by the device 8 by default) between the software system 1 and the device 8 for transmitting the device-code 80, in this case, the software

system 1 only needs to know the IP Address of the device 8 to actively connect to it and then transmit the updated device-code 80. In addition, the network location information can also be an IP Address plus a Port, that is, the network location information includes the specified information of the network communication protocol. In this way, there is no need for any default network communication protocol between the software system 1 and the device 8, and the software system 1 can also establish a connection between devices according to the IP Address (location) plus the Port (communication protocol) and transmit the updated device-code 80. The approach of completing the update of the device-code 80 in an asynchronous manner is not limited thereto, and those skilled in the art can make any equivalent changes and designs according to actual application requirements.

[0104] In a preferred embodiment, the device-code 80 provided by the first user 6 is provided to the software system 1 in the form of a software code package; the form of the software code package may include a compressed file containing all software codes included in the device-code 80. In another preferred embodiment, the device-code 80 provided by the first user 6 is provided to the software system 1 in the form of a system image file, and the system image file contains all software codes included in the device-code 80.

[0105] In a preferred embodiment, regardless of whether in the form of a system image file or a software code package, when the software system 1 receives the device-code 80, it can scan all software codes included in the device-code 80. First, mount the system image file containing the device-code 80 to a memory or a sector divided from the hard disk memory (when it is a zImage, its content can be decompressed into a folder by means of decompression), or decompress the software code package containing the device-code 80 into a specified folder of the hard disk memory. Then, the file system can be used to traverse the files and directories of the mounted sector or the folder, that is, scan the component composition contained in the system image file and the software code package. After grasping the file and directory structure of the device-code 80 according to the traversal result, it can be compared with the file and directory structures of known operating system cores, driver programs, application programs, common software components, and/or class libraries to find out the operating system cores, driver programs, application programs, common software components, and/or class libraries with matching file and directory structures. According to the above procedure, after collecting the device-code 80, the software system 1 can scan and analyze a composition list of the software codes included in the device-code 80. The composition list may include metadata of each software code existing in forms such as an operating system core, a driver program, an

application program, a common software component, and/or a class library (the metadata is data describing information related to the software code, for example, descriptive information such as the name of each software code can be a form of metadata), and the content of the composition list may also include metadata of the device-code 80 itself (the metadata is data describing information related to the device-code 80).

[0106] In a preferred embodiment, according to the composition list of the at least one software code included in the device-code 80, the software system 1 can monitor the CVE list corresponding to each software code contained in the composition list (for example, for the Linux Kernel, monitor the CVE list corresponding to the Linux Kernel), and detect code analysis events by monitoring abnormal changes in the CVE list. When a vulnerability is added to the list, a patch code corresponding to the vulnerability is provided and added to the vulnerability record, or the patch code of the vulnerability is updated, leading to abnormal changes in the content of the list, a code analysis event can be triggered; where the patch code is a form included in the risk code, but is not limited thereto. For the malicious software code which is one form of the risk code, those skilled in the art can make any equivalent changes and designs according to actual application requirements. The code analysis event corresponding to the risk code is triggered when the risk code is added or updated, and the triggered event can be received by the comparison unit 4. The CVE list is an implementation form of the risk code login list described herein, but is not limited thereto, and those skilled in the art can make any equivalent changes and designs for different forms of risk code login lists according to actual application requirements.

[0107] In a preferred embodiment, after the device 8 completes the update of the device-code 80, when a patch code is added or updated, resulting in the triggering of a code analysis event corresponding to the patch code, the software system 1 transmits the code analysis event to the comparison unit 4. When the comparison unit 4 receives the code analysis event, the comparison unit 4 can request the code acquisition unit 3 to retrieve the patch code from the storage medium 9 storing the patch code; where the storage medium 9 may be the same as the storage medium 9 storing the device-code 80 or another storage medium 9. In another preferred embodiment, when the software system 1 monitors that a patch code is added or updated, it can first retrieve and compile the source code of the patch code stored in another storage medium 9, and store the compiled result (binary code or intermediate code) in the storage medium 9. In this case, the pre-compiled result can be regarded as part of the entire patch code, and the comparison unit 4 retrieves the entire patch code (including source code, binary code, and/or intermediate

code) from the storage medium 9 and the other storage medium 9. The above approach for monitoring the risk code login list and retrieving codes is not limited thereto, and those skilled in the art can make any equivalent changes and designs according to actual application requirements.

[0108] In a preferred embodiment, after the comparison unit 4 retrieves the patch code, it can compare the component contents of the patch code and the corresponding at least one software code in the device-code 80. When the comparison result indicates that the at least one software code contains at least one security risk, the transmission unit 5 can be requested to send a risk early warning notification regarding the security risk.

[0109] In a preferred embodiment, sending the risk early warning notification refers to sending a risk early warning notification to at least one of the provider of the device-code 80 (the first user 6), the operator of the device 8 (the second user 7), and the manager (the second user 7 or the manager of the software system 1) via at least one network communication channel, such as email, mobile phone push notification, mobile phone short message notification, and other network communication channels. The function of the risk early warning is that the patch code is one form included in the risk code. The existence of the patch code itself indicates that there is a corresponding known or newly discovered vulnerability in the software code included in the device-code 80, and the patch code is the code for correcting the vulnerability. When the comparison result of the component contents of the patch code and the software code shows that the component content of the software code does not include the patch code, it means that the vulnerability is still present in the device-code 80. Unscrupulous persons can use the vulnerability to attack the device 8 that executes the device-code 80 and is running online. That is, the existence of the vulnerability is a risk to the first user 6, the second user 7, or any other third party. The risk early warning notification is an advance warning notification to relevant persons regarding this risk, thereby realizing risk prevention and control for the device-code 80.

[0110] In a preferred embodiment, when the form of the risk code is a malicious software code, the malicious software code represents the code of a known vulnerability. That is, when the comparison result of the component contents of the malicious software code and the software code shows that the component content of the software code still includes the malicious software code, it means that the vulnerability is still present in the device-code 80, and a risk early warning notification needs to be sent.

[0111] Regarding the “component content of the software code” and the “component

content of the risk code” described herein, the “component content” refers to including “program instructions contained in the code composition content of the code”, “storage structure of the code in the memory”, and “environment settings and configuration data required for executing the code”; where the code composition content of the code includes any binary code for loading and execution, intermediate code, and/or any computer program code written in a computer programming language. In addition, the “security risk” described herein includes the risk that the code contains vulnerabilities, viruses, and/or malicious software, and/or any risk that causes the code to perform unexpected logical computing. The above approach of scanning for a security risk by comparing the risk code with the software code is not limited thereto, and those skilled in the art can make any equivalent changes and designs according to actual application requirements.

[0112] As shown in FIG. 3, the comparison of the component contents of the risk code and the at least one software code in S2 includes the following steps:

[0113] S21. Extracting at least one first component feature corresponding to the risk code;

[0114] S22. Extracting at least one second component feature corresponding to the at least one software code; and

[0115] S23. Comparing the at least one first component feature with the at least one second component feature.

[0116] Further, the first component feature in S21 is obtained by analyzing code composition content and/or code execution result corresponding to the risk code; where the code composition content includes the risk code and/or a first intermediate code obtained by compiling the risk code, and the code execution result includes semantic information revealed during simulation execution of the risk code and/or the first intermediate code when performing simulation execution on the risk code and/or the first intermediate code.

[0117] Further, the at least one second component feature in S22 is obtained by analyzing at least one code composition content and/or at least one code execution result corresponding to the at least one software code; where the at least one code composition content includes at least one binary code and/or at least one second intermediate code obtained by disassembling the at least one binary code, and the at least one code execution result includes semantic information revealed during simulation execution of the at least one binary code and/or the at least one second intermediate code when performing simulation execution on the at least one binary code

and/or the at least one second intermediate code.

[0118] Further, a programming language structure of the first intermediate code and/or the at least one second intermediate code is declared by an intermediate language, an intermediate presentation, a structured language, and/or a structured protocol.

[0119] Types of the risk code in S2 include a patch code associated with the at least one software code, a malicious program code, and/or any code used for correcting or generating a software vulnerability or a software virus.

[0120] The code analysis event of the risk code in S2 is triggered when the risk code is added or updated to a risk code login list.

[0121] As shown in FIG. 4, before the code analysis event in S2 is triggered, the method further includes: detecting at least one code analysis event by the software system 1 according to a composition list of the at least one software code. In S2, the code analysis event corresponding to the risk code is one of the at least one code analysis event.

[0122] As shown in FIG. 2, the component composition of the device-code 80 of the device 8 may include software codes existing in different forms such as a system core, a driver program, an application program, a component, and/or a class library. The code composition content of each software code may include some Scripts, binary codes, and/or intermediate codes. Therefore, the at least one software code can be understood as part of the device-code 80. The at least one code composition content can be understood as part of the at least one software code.

[0123] Generally speaking, the at least one software code contained in a device 8 includes one list, for example, which may include components such as Linux Kernel, OpenSSL, sqlite, and the like. As these components are continuously inspected, vulnerabilities are discovered and corresponding patch codes are released, which are then added or updated to online CVE lists (CVE repositories) and GIT Hub.

[0124] When these vulnerabilities and patch codes are added or updated to online CVE lists and GIT Hub by the respective entities or organizations, the software system 1 described herein will not be notified. Therefore, for the code analysis event to be triggered, it means that before the trigger, the software system 1 should monitor CVE lists and GIT Hub such as those for Linux Kernel, OpenSSL, sqlite, etc., according to the composition list of the device-code 80, so as to detect these code analysis events.

[0125] The technical effect achieved thereby is that the software system 1 can immediately complete the code security detection of the device-code 80 when a vulnerability and a patch code are added or updated to an online risk code login list (for example, a CVE list).

[0126] Further, the composition list is obtained by the software system 1 scanning and/or analyzing the component composition of the device-code 80 when the device-code 80 is provided to the software system 1.

[0127] The device-code 80 may be a single .o file (object file), .img file (system image file), .so file (program library), or a .zip or .gz compressed file (the content includes all .o and .so files required by the entire device, and the path structure of these files is also retained in the compressed file).

[0128] For the overall device-code 80 of a device 8, there are many files of software codes included therein. Each software code is combined in the device-code 80 in the form of different components (for example, Linux Kernel, OpenSSL, sqlite, etc.). The files of each software code are located in different path positions. There are different ways to obtain the composition list of the software codes included in the device-code 80. One way is to scan the entire device-code 80 for file path information. As software codes of different components, their related files are placed in different positions, so it can be basically determined which components are contained in the device-code 80. In addition, the content of each software code file can also be read to check whether there are certain characteristics of specific components in the file content to grasp the component list. For example, when reading the code composition content in a code file included in a software code, it can be compared with known code segments of known software such as Linux Kernel, OpenSSL, sqlite, etc. According to the comparison result, it can be determined which components are contained in the device-code 80. The approach of scanning and/or analyzing the component composition of the device-code 80 to obtain the composition list of each software code in the device-code 80 is not limited thereto, and those skilled in the art can make any equivalent changes and designs according to actual application requirements.

[0129] In a preferred embodiment, the code composition content of the patch code includes a file written in Javascript and containing Javascript code. The function of the patch code is to correct a vulnerability of a software code included in the device-code 80. The comparison unit in the software system 1 can first locate the position of the Javascript file corresponding to the file name in the code composition content of the software code by means of the file name of the Javascript file. After reading the contents of the Javascript file of the patch

code and the corresponding Javascript file of the software code respectively, the component contents contained in the two files are compared, that is, it is compared whether the contents of the two files include the corrected code segment. The corrected code segment is used as the component feature for comparison. A consistent comparison result indicates that the software code does not contain the vulnerability, and an inconsistent comparison result indicates that the software code contains the vulnerability.

[0130] In a preferred embodiment, in addition to segments contained in the code as features, the component features for comparison can also be the storage structure of the patch code and the software code in the memory and/or the environment settings and configuration information of the patch code and the software code. For example: a new file is added to the code composition content of a patch code, and the file name of this file does not exist in the code composition content of the corresponding software code; and/or a new parameter is added to the environment settings and configuration of the patch code, and this parameter does not exist in the environment settings and configuration of the corresponding software code. It can be said that the component content of the software code does not include the patch code, and thus the vulnerability corresponding to the patch code is still present in the software code.

[0131] As in the above example, the code segments, file name structure, and/or configuration content of the patch code used for comparison are the first component features in the component content comparison. The code segments, file name structure, and/or configuration content of the software code used for comparison are the second component features in the component content comparison. The approach of comparing component features obtained by analyzing code composition content is not limited thereto, and those skilled in the art can make any equivalent changes and designs according to actual application requirements.

[0132] In a preferred embodiment, the code composition content of a software code includes a first file containing binary code, and the file name of the first file and its relative position in the storage structure can correspond to a second file contained in a patch code and written in C/C++ source code. The second file contains code for correcting a vulnerability. In this case, the first file is first disassembled into a first intermediate code (code in the form of LLVM IR as the intermediate code representation), and the second file is also compiled and then disassembled into a second intermediate code (code in the form of LLVM IR as the intermediate code representation). Furthermore, simulation execution can be performed on the first intermediate code and the second intermediate code, that is, by reading the program instructions in the first intermediate code and the second intermediate code, feeding in simulated parameters,

and performing logical computing according to the read program instructions. After all program instructions are executed, the semantic information revealed during the simulation execution process can be obtained according to the execution history of each instruction during the execution of the program instructions, the output and input of function calls, and/or the execution order; where the “semantic information” described herein includes the values of memory and/or intermediate variables during program operation, and/or the instructions, functions called during the simulation execution process, and their output and input values. A similarity is obtained by comparing the semantic information of the first intermediate code with the semantic information of the second intermediate code, and it can be determined whether the code segments of the first intermediate code and the second intermediate code are consistent according to the similarity, thereby identifying whether the first file contains the code for correcting the vulnerability in the second file. When the component features formed by all the contents of the semantic information of the first intermediate code and the semantic information of the second intermediate code are completely included in semantic information of each other, the similarity can be said to be 100%. When the component features formed by only a part of the contents are included in the semantic information of each other, the proportion of the part to the total content is the similarity. Based on this, a similarity result ranging from 0% to 100% can be obtained. When the similarity exceeds a set threshold, it indicates that the first file contains the code of the second file. The threshold is a reference value for determining whether the code of the second file is included in the first file. For example, when the threshold is set to 90%, it means that when the similarity is higher than 90%, the first file contains the code of the second file; otherwise, the code of the second file is not contained in the first file.

[0133] In a preferred embodiment, the code composition content of a software code includes a first file containing intermediate code (Java bytecode), and the file name of the first file and its relative position in the storage structure can correspond to a second file contained in a patch code and written in Java source code. The second file contains code for correcting a vulnerability. In this case, after compiling the second file into intermediate code (Java bytecode), its semantic information is obtained through simulation execution, and then it can be compared whether the first file contains the code for correcting the vulnerability in the second file through the component features such as values retrieved from memory and/or intermediate variables contained in the semantic information, and/or instructions, functions called during the simulation execution process, and their output and input values.

[0134] The above approach of comparing component features of semantic information

obtained through simulation execution is not limited thereto, and those skilled in the art can make any equivalent changes and designs according to actual application requirements.

[0135] Sending the risk early warning in S3 carries an instruction, and the instruction is used to instruct the device 8 and/or a user operating the device 8 to perform an update operation of the risk code on the device 8; or the risk early warning notification carries prompt information, and the prompt information is used to provide an early warning of the at least one security risk for the first user 6, the second user 7, and/or the system administrator.

[0136] The technical effect lies in that the risk early warning notification can carry an instruction. When a security risk of the device 8 occurs, the device 8 can be directly instructed to perform an immediate update of the risk code through the instruction to immediately fix the vulnerability. The reason why the technical solution of the present invention can achieve this effect is that it targets the component analysis of binary codes, that is, when a vulnerability is detected in the device-code 80, the binary code (risk code) of the patch for the vulnerability is already available, so the device 8 can be notified to update the risk code immediately.

[0137] As shown in FIG. 5, the present embodiment further provides a device-code risk early warning computing device 11, including a collection unit 21, a code acquisition unit 31, a comparison unit 41, and a transmission unit 51; where

[0138] the collection unit 21 is configured to receive an update request for a device-code 80;

[0139] the code acquisition unit 31 is configured to obtain the device-code 80 from a storage medium 9, where the device-code 80 includes at least one software code;

[0140] the transmission unit 51 is configured to transmit the device-code 80 to a device 8 and send a risk early warning notification; and

[0141] the comparison unit 41 is configured to, when a code analysis event corresponding to a risk code is triggered, enable the code acquisition unit 31 to obtain the risk code from the storage medium 9 and/or another storage medium 9, and compare the risk code with the device-code 80.

[0142] When the comparison result of the above comparison indicates that the at least one software code contains at least one security risk, the transmission unit 51 is further configured to send an early warning notification. The transmission unit 51 sends a risk early warning notification to at least one of a provider of the device-code 80, an operator of the device

8, and a manager via at least one network communication channel.

[0143] In a preferred embodiment, the computing device 11 is an embedded device. The collection unit 21, the code acquisition unit 31, the comparison unit 41, and/or the transmission unit 51 are designed in a hardware description language and can be integrated into an integrated circuit unit of the computing device 11 or into an embedded system of the computing device 11 in the form of software code. The collection unit 21, the code acquisition unit 31, the comparison unit 41, and the transmission unit 51 are driven by the embedded system running in the computing device 11.

[0144] In another preferred embodiment, the computing device 11 is a server. The collection unit 21, the code acquisition unit 31, the comparison unit 41, and/or the transmission unit 51 are integrated into a software system 1 in the form of software code. The software system 1 is provided to the computing device 11 for operation through a built-in or installation manner, and the collection unit 21, the code acquisition unit 31, the comparison unit 41, and the transmission unit 51 are driven by the software system 1 running in the computing device 11.

[0145] In another preferred embodiment, the computing device 11 includes distributed servers. The collection unit 21, the code acquisition unit 31, the comparison unit 41, and/or the transmission unit 51 are integrated into a distributed software system 1 in the form of software code. The distributed software system 1 is deployed to the distributed servers for operation to form a computing device 11. The collection unit 21, the code acquisition unit 31, the comparison unit 41, and the transmission unit 51 are driven by the distributed software system 1 running in the computing device 11.

[0146] The above manner of composing the computing device 11 through combinations of different forms of hardware and software is not limited thereto, and those skilled in the art can make any equivalent changes and designs according to actual application requirements.

[0147] Further, with reference to FIG. 3, the comparison unit 41 includes a first feature extraction unit, a second feature extraction unit, and a feature comparison unit; where

[0148] the first feature extraction unit is configured to extract at least one first component feature corresponding to the risk code;

[0149] the second feature extraction unit is configured to extract at least one second component feature corresponding to the at least one software code; and

[0150] the feature comparison unit is configured to compare the at least one first

component feature with the at least one second component feature.

[0151] Still further, the first feature extraction unit or the second feature extraction unit includes at least one analysis unit and at least one retrieval unit; where

[0152] the at least one analysis unit is configured to obtain a first component feature by analyzing code composition content and/or code execution result corresponding to the risk code, and/or obtain a second component feature by analyzing code composition content and/or code execution result corresponding to the at least one software code; and

[0153] the at least one retrieval unit is configured to extract the first component feature or the second component feature obtained after analysis.

[0154] With reference to FIG. 4, the device-code risk early warning computing device further includes at least one detection unit; where

[0155] the at least one detection unit is configured to monitor a risk code login list (for example, a CVE list) corresponding to each software code contained in the composition list according to the composition list of the at least one software code included in the device-code 80, so as to detect a code analysis event by monitoring abnormal changes in the risk code login list.

[0156] In summary, the present invention discloses a device-code risk early warning method, a software system 1, and a computing device 11. By updating a device-code 80 containing at least one software code for an online device 8, comparing component contents of a risk code and the device-code 80, and sending a risk early warning for the device 8, the present invention can perform risk early warning for the device 8 regardless of whether the device 8 is offline or the network status is normal, thereby realizing risk prevention and control for the device-code 80.

[0157] The above detailed descriptions are specific explanations of feasible embodiments of the present invention. The embodiments are not intended to limit the scope of protection of the present invention. All equivalent implementations or changes made without departing from the present invention shall be included in the scope of protection of the present invention.

WHAT IS CLAIMED IS:

1. A device-code risk early warning method, characterized by comprising following steps:

collecting an update request for a device-code, obtaining the device-code according to content of the update request, and transmitting the device-code to a device, wherein the device-code comprises at least one software code;

obtaining a risk code when a code analysis event corresponding to the risk code is triggered, and comparing component contents of the risk code and the at least one software code; and

sending a risk early warning notification when a result of the comparison indicates that the at least one software code contains at least one security risk.
2. The device-code risk early warning method according to claim 1, characterized in that sending the risk early warning notification refers to sending the risk early warning notification to at least one of a provider of the device-code, an operator of the device, and a manager via at least one network communication channel.
3. The device-code risk early warning method according to claim 1, characterized in that the device-code is provided by a first user, the collection of the update request refers to receiving for release and/or update of the device-code, and the device is operated by a second user.
4. The device-code risk early warning method according to claim 1, characterized in that the at least one software code comprises at least one binary code.
5. The device-code risk early warning method according to claim 1, characterized in that obtaining the device-code specifically refers to obtaining the device-code from a storage medium, comprising:

obtaining the device-code from the storage medium according to code version information, wherein the device-code is specified by the code version information.
6. The device-code risk early warning method according to claim 5, characterized in that the code version information is directly specified in the content of the update request, or the code version information required for updating the device is determined according to a product specification, a hardware specification, and/or a firmware specification contained in the content of the update request.
7. The device-code risk early warning method according to claim 1, characterized in that the update request is an online update request transmitted via a network.

8. The device-code risk early warning method according to claim 1, characterized in that the content of the update request comprises network location information of the device, wherein the device-code is transmitted to the device via the network according to the network location information.

9. The device-code risk early warning method according to claim 1, characterized in that comparing the component contents of the risk code and the at least one software code comprises:

extracting at least one first component feature corresponding to the risk code;

extracting at least one second component feature corresponding to the at least one software code; and

comparing the at least one first component feature with the at least one second component feature.

10. The device-code risk early warning method according to claim 9, characterized in that the first component feature is obtained by analyzing code composition content and/or a code execution result corresponding to the risk code.

11. The device-code risk early warning method according to claim 10, characterized in that the code composition content comprises the risk code and/or a first intermediate code obtained by compiling the risk code, wherein the code execution result comprises semantic information revealed during simulation execution of the risk code and/or the first intermediate code when performing simulation execution on the risk code and/or the first intermediate code.

12. The device-code risk early warning method according to claim 9, characterized in that the at least one second component feature is obtained by analyzing at least one code composition content and/or at least one code execution result corresponding to the at least one software code.

13. The device-code risk early warning method according to claim 12, characterized in that the at least one code composition content comprises at least one binary code and/or at least one second intermediate code obtained by disassembling the at least one binary code, wherein the at least one code execution result comprises semantic information revealed during simulation execution of the at least one binary code and/or the at least one second intermediate code when performing simulation execution on the at least one binary code and/or the at least one second intermediate code.

14. The device-code risk early warning method according to claim 11 or claim 13, characterized in that a programming language structure of the intermediate code is declared by

an intermediate language, an intermediate representation, a structured language, and/or a structured protocol.

15. The device-code risk early warning method according to claim 1, characterized in that types of the risk code comprise a patch code, a malicious program code, and/or any code used for correcting or generating a software vulnerability or a software virus.

16. The device-code risk early warning method according to claim 1, characterized in that the code analysis event corresponding to the risk code is triggered when the risk code is added or updated.

17. The device-code risk early warning method according to claim 1, characterized by comprising detecting, before the code analysis event is triggered, the at least one code analysis event according to a composition list of the at least one software code, wherein the code analysis event corresponding to the risk code is one of the at least one code analysis event.

18. The device-code risk early warning method according to claim 17, characterized in that the composition list is obtained by scanning and/or analyzing a component composition of the device-code when the device-code is collected.

19. The device-code risk early warning method according to claim 18, characterized in that the composition list comprises metadata of the device-code itself and/or metadata of the at least one software code contained in the component composition of the device-code.

20. The device-code risk early warning method according to claim 19, characterized in that the at least one software code is present in the component composition of the device-code in forms including a system core, a driver program, an application program, a component, and/or a class library.

21. The device-code risk early warning method according to claim 20, characterized by comprising scanning and/or analyzing, when the device-code is a system image file, the component composition contained in the system image file to identify the at least one software code present in the system image file, so as to obtain the composition list of the at least one software code; or

scanning and/or analyzing, when the device-code is a software code package, the component composition contained in the software code package to identify the at least one software code present in the software code package, so as to obtain the composition list of the at least one software code.

22. The device-code risk early warning method according to claim 2, characterized by comprising instructing, when the risk early warning notification carries an instruction, the device and/or a user operating the device by means of the instruction to perform an update operation of the risk code on the device; or providing, when the risk early warning notification carries prompt information, an early warning of the at least one security risk for the provider of the device-code, the operator of the device, and/or the manager by means of the prompt information.

23. A device-code risk early warning software system, characterized by comprising a collection unit, a code acquisition unit, a comparison unit, and a transmission unit; wherein

the collection unit is configured to receive an update request for a device-code;

the code acquisition unit is configured to obtain the device-code, wherein the device-code comprises at least one software code;

the transmission unit is configured to transmit the device-code to a device and send a risk early warning notification; and

the comparison unit is configured to, when a code analysis event corresponding to a risk code is triggered, enable the code acquisition unit to obtain the risk code, and compare the risk code with the device-code.

24. The device-code risk early warning software system according to claim 23, characterized in that the comparison unit comprises a first feature extraction unit, a second feature extraction unit, and a feature comparison unit; wherein

the first feature extraction unit is configured to extract at least one first component feature corresponding to the risk code;

the second feature extraction unit is configured to extract at least one second component feature corresponding to the at least one software code; and

the feature comparison unit is configured to compare the at least one first component feature with the at least one second component feature.

25. The device-code risk early warning software system according to claim 24, characterized in that the first feature extraction unit or the second feature extraction unit comprises at least one analysis unit and at least one retrieval unit; wherein

the at least one analysis unit is configured to obtain a first component feature by analyzing code composition content and/or a code execution result corresponding to the risk code, and/or

obtain a second component feature by analyzing code composition content and/or a code execution result corresponding to the at least one software code; and

the at least one retrieval unit is configured to extract the first component feature and/or the second component feature obtained after analysis.

26. The device-code risk early warning software system according to claim 23, characterized by further comprising at least one detection unit; wherein

the at least one detection unit is configured to monitor a risk code login list corresponding to each software code contained in the composition list according to the composition list of the at least one software code, so as to detect a code analysis event by monitoring abnormal changes in the risk code login list.

27. A device-code risk early warning computing device, characterized by comprising a collection unit, a code acquisition unit, a comparison unit, and a transmission unit; wherein

the collection unit is configured to receive an update request for a device-code;

the code acquisition unit is configured to obtain the device-code from a storage medium, wherein the device-code comprises at least one software code;

the transmission unit is configured to transmit the device-code to a device and send a risk early warning notification; and

the comparison unit is configured to, when a code analysis event corresponding to a risk code is triggered, enable the code acquisition unit to obtain the risk code from the storage medium and/or another storage medium, and compare the risk code with the device-code.