

METHOD AND APPARATUS FOR DEBUGGING CLOUD TERMINAL, AND COMPUTER PROGRAM

TECHNICAL FIELD

5 **[0001]** The present disclosure relates to the field of computer technologies, and in particular, to the fields of cloud computing and virtualization technologies, and more particularly, to a method and an apparatus for debugging a cloud terminal, an electronic device, a storage medium and a computer program product, which may be applied to cloud terminal debugging scenarios.

BACKGROUND

10 **[0002]** ADB (Android Debug Bridge) is a debugging tool provided by Android (a mobile operating system) for developers. In the traditional ADB debugging mode, debugging personnel connect a debugging terminal to a mobile phone through a data
15 cable to perform ADB debugging on the mobile phone. However, cloud mobile phones are deployed on the cloud, and the traditional data cable method cannot achieve connection to cloud phones, thereby making ADB debugging of cloud phones impossible.

20 SUMMARY

[0003] The present disclosure provides a method, an apparatus, an electronic device, a storage medium and a computer program product for debugging a cloud terminal.

[0004] According to a first aspect, there is provided a method for debugging a cloud terminal, including:

25 **[0005]** establishing a secure interaction channel between a target cloud terminal and a debugging terminal through an agent service container corresponding to the target cloud terminal, where the agent service container is deployed in a server data center where the target cloud terminal is located;

[0006] establishing a debugging channel between the target cloud terminal and the

debugging terminal on the basis of the secure interaction channel; and

[0007] transmitting debugging data between the target cloud terminal and the debugging terminal based on the debugging channel.

[0008] According to a second aspect, there is provided an apparatus for debugging a cloud terminal, including:

[0009] a secure channel establishment unit configured to establish a secure interaction channel between a target cloud terminal and a debugging terminal through an agent service container corresponding to the target cloud terminal, where the agent service container is deployed in a server data center where the target cloud terminal is located;

[0010] a debugging channel establishment unit configured to establish a debugging channel between the target cloud terminal and the debugging terminal on the basis of the secure interaction channel; and

[0011] a debugging unit configured to transmit debugging data between the target cloud terminal and the debugging terminal based on the debugging channel.

[0012] According to a third aspect, there is provided an electronic device, including:

[0013] at least one processor; and

[0014] a memory communicatively connected to the at least one processor;

[0015] where the memory stores instructions executable by the at least one processor, and the instructions, when executed by the at least one processor, enable the at least one processor to perform the method as described in any implementation of the first aspect.

[0016] According to a fourth aspect, there is provided a non-transitory computer-readable storage medium storing computer instructions, where the computer instructions are used to cause a computer to perform the method as described in any implementation of the first aspect.

[0017] According to a fifth aspect, there is provided a computer program product, including a computer program which, when executed by a processor, implements the method as described in any implementation of the first aspect.

[0018] According to the technology of the present disclosure, a method and an apparatus for debugging a cloud terminal are provided. A secure interaction channel

between a target cloud terminal and a debugging terminal is established through an agent service container corresponding to the target cloud terminal, so as to establish a debugging channel on the basis of the secure interaction channel for debugging the cloud terminal, thereby providing a way for debugging cloud terminals and improving the convenience and security of the cloud terminal debugging process.

[0019] It should be understood that the content described in this section is not intended to identify key or important features of the embodiments of the present disclosure, nor is it used to limit the scope of the present disclosure. Other features of the present disclosure will become easily understood through the following description.

BRIEF DESCRIPTION OF THE DRAWINGS

[0020] The drawings are used to better understand the solution of the present disclosure and do not constitute a limitation to the present disclosure, where:

[0021] FIG. 1 is a schematic diagram of an exemplary system architecture to which an embodiment of the present disclosure may be applied;

[0022] FIG. 2 is a flowchart of an embodiment of a method for debugging a cloud terminal according to the present disclosure;

[0023] FIG. 3 is a schematic diagram of an application scenario of the method for debugging a cloud terminal according to the embodiment;

[0024] FIG. 4 is a flowchart of another embodiment of the method for debugging a cloud terminal according to the present disclosure;

[0025] FIG. 5 is a structural diagram of an embodiment of an apparatus for debugging a cloud terminal according to the present disclosure;

[0026] FIG. 6 is a structural schematic diagram of a computer system adapted for implementing the embodiments of the present disclosure.

DETAILED DESCRIPTION OF EMBODIMENTS

[0027] The following describes exemplary embodiments of the present disclosure with reference to the accompanying drawings, which include various details of the

embodiments of the present disclosure to facilitate understanding, and they should be regarded as merely exemplary. Therefore, those of ordinary skill in the art should recognize that various changes and modifications may be made to the embodiments described herein without departing from the scope and spirit of the present disclosure. Similarly, descriptions of well-known functions and structures are omitted in the following for clarity and conciseness.

[0028] In the technical solution of the present disclosure, the collection, storage, use, processing, transmission, provision and disclosure of user personal information all comply with the provisions of relevant laws and regulations and do not violate public order and good customs.

[0029] FIG. 1 shows an exemplary architecture 100 to which the method and apparatus for debugging a cloud terminal of the present disclosure may be applied.

[0030] As shown in FIG. 1, the system architecture 100 may include terminal devices 101, 102, 103, a network 104 and a server 105. The terminal devices 101, 102, 103 are communicatively connected to form a topological network, and the network 104 is a medium for providing a communication link between the terminal devices 101, 102, 103 and the server 105. The network 104 may include various types of connections, such as wired, wireless communication links or optical fiber cables, and the like.

[0031] The terminal devices 101, 102, 103 may be hardware devices or software that support network connection for data interaction and data processing. When the terminal devices 101, 102, 103 are hardware, they may be various electronic devices supporting network connection and functions such as information acquisition, interaction, display and processing, including but not limited to smart phones, tablet computers, e-book readers, laptop portable computers, desktop computers and the like. When the terminal devices 101, 102, 103 are software, they can be installed in the above-listed electronic devices. The terminal devices may be implemented as multiple pieces of software or software modules for providing distributed services, or as a single piece of software or software module, which is not specifically limited herein.

[0032] The server 105 may be a server that provides various services, for example, a background processing server that obtains cloud terminal debugging requests from the terminal devices 101, 102, 103 and debugs the deployed cloud terminals. Optionally, the

server may feed back the debugging result to the terminal device. As an example, the server 105 may be a cloud server.

[0033] It should be noted that the server may be hardware or software. When the server is hardware, it can be implemented as a distributed server cluster composed of multiple servers, or as a single server. When the server is software, it can be implemented as multiple pieces of software or software modules (such as software or software modules for providing distributed services), or as a single piece of software or software module, which is not specifically limited herein.

[0034] It should also be noted that the method for debugging a cloud terminal provided by the embodiments of the present disclosure is generally executed by a server, or by a server and a terminal device in cooperation with each other. Correspondingly, the parts (for example, the units) included in the apparatus for debugging a cloud terminal may be all disposed in the server, or may be respectively disposed in the server and the terminal device.

[0035] It should be understood that the numbers of terminal devices, networks and servers in FIG. 1 are merely examples. According to implementation requirements, there may be any number of terminal devices, networks and servers. When the electronic device on which the method for debugging a cloud terminal runs does not need to perform data transmission with other electronic devices, the system architecture may only include the electronic device (for example, a server) on which the method for debugging a cloud terminal runs.

[0036] Please refer to FIG. 2, which is a flowchart of a method for debugging a cloud terminal provided by an embodiment of the present disclosure. The process 200 includes the following steps.

[0037] Step 201 includes: establishing a secure interaction channel between a target cloud terminal and a debugging terminal through an agent service container corresponding to the target cloud terminal.

[0038] In this embodiment, an execution subject of the method for debugging a cloud terminal (for example, the server in FIG. 1) may establish a secure interaction channel between a target cloud terminal and a debugging terminal through an agent service container corresponding to the target cloud terminal, where the agent service container is

deployed in a server data center where the target cloud terminal is located. The agent service in the agent service container acts as an intermediate layer, enabling the debugging terminal to remotely access and debug the cloud terminal through the Internet without directly connecting to the internal network where the cloud terminal is located.

5 **[0039]** A cloud terminal is a virtualized terminal device based on cloud computing technology, which transfers the traditional local computing, storage and display functions to a cloud server, and provides cloud resources to users in the form of virtual terminals through a network. These virtual terminals run on the cloud, and users can access and operate these cloud terminals through terminal devices such as thin clients,
10 ordinary computers and mobile devices by using a remote desktop protocol or specific client software, so as to achieve an experience similar to using local devices. The cloud terminal is, for example, a cloud mobile phone or a cloud computer.

15 **[0040]** In this embodiment, the execution subject may establish a first sub-channel between the agent service container and the debugging terminal through the agent service container corresponding to the target cloud terminal; establish a second sub-channel between the agent service container and the cloud terminal through the agent service container; and generate a secure interaction channel by combining the first sub-channel and the second sub-channel.

20 **[0041]** As an example, the first sub-channel may be created through various communication protocols, such as SSH (Secure Shell) protocol, HTTP (Hypertext Transfer Protocol), HTTPS (Hypertext Transfer Protocol Secure), WebSocket protocol, SSL (Secure Sockets Layer), TLS (Transport Layer Security), FTP (File Transfer Protocol), SFTP (SSH File Transfer Protocol), VPN (Virtual Private Network).

25 **[0042]** Taking the SSH protocol as an example, SSH server software such as OpenSSH is installed and configured in the agent service container. On the debugging terminal, SSH client software such as an OpenSSH client, PuTTY and the like is used to specify information such as the IP (Internet Protocol) address or domain name of the agent service container and the SSH service port. A connection request is initiated to the SSH service of the agent service container through the SSH client. The SSH server of the
30 agent service container authenticates the identity of the debugging terminal, and establishes an SSH connection if the authentication is passed.

[0043] Since the agent service container is deployed in the server data center where the target cloud terminal is located, a second sub-channel between the agent service container and the cloud terminal may be established through the intranet of the data center.

5 **[0044]** In some optional implementations of this embodiment, the execution subject may perform the above step 201 in the following manner: first, determine a network port corresponding to the agent service container, where the network port characterizes a network connection port for a cloud terminal corresponding to the agent service container; then, establish the secure interaction channel between the target cloud terminal
10 and the debugging terminal through the agent service container according to the network port.

[0045] The network port is used to monitor requests from the debugging terminal in the external network, and is an entrance for the agent service container to communicate with the outside world. For example, the port may be set to 11515, and it is ensured that
15 the mapping relationship of the port is specified when creating a docker container, so that the external network can access the agent service container through this port. A port is allocated to each container, so that different target objects (for example, users who purchase cloud terminals) can access the containers allocated to the target objects through different network ports, realizing customization of the agent service container
20 for the target objects. All available network ports form a port pool, and for the unallocated network ports in the port pool, they may be allocated to newly created agent service containers.

[0046] In this implementation, first, a first sub-channel between the debugging terminal and the agent service container is established according to the network port.
25 Specifically, on the debugging terminal, the external network IP address and network port information of the agent service container are obtained. On the debugging terminal, client software supporting a tunnel function is used to configure connection parameters, including the external network IP address and network port of the agent service container and other information, according to the requirements of a corresponding protocol (such as the SSH protocol). A connection request is initiated through the client software, and
30 an initial connection between the debugging terminal and the agent service container is established by using the network port of the agent service container. After receiving the

connection request, the agent service container verifies the connection request according to a preset security policy (such as an authentication method based on a key or a certificate), and if the verification is passed, a secure interaction channel is established between the debugging terminal and the agent service container, laying a foundation for establishing a debugging channel subsequently and ensuring the security of communication between the debugging terminal and the target cloud terminal. Through this secure interaction channel, the requests of the debugging terminal are forwarded to the target cloud terminal, and the responses of the target cloud terminal are returned to the debugging terminal.

[0047] Then, a second sub-channel between the agent service container and the target cloud terminal is established based on the intranet of the data center; finally, a secure interaction channel is generated by combining the first sub-channel and the second sub-channel.

[0048] This implementation provides a specific implementation of the secure interaction channel. Establishing the secure interaction channel based on the network port corresponding to the agent service container can hide the actual network location of the target cloud terminal and reduce the risk of its direct exposure. At the same time, strict security policies such as authentication and firewall rules are implemented on the agent service container, which can effectively prevent unauthorized access and attacks and ensure debugging security. In addition, the agent service container can centrally manage connection requests of multiple cloud terminals, and uniformly control and monitor network access, which improves resource utilization, simplifies the network architecture, and facilitates maintenance and expansion.

[0049] In some optional implementations of this embodiment, the agent service container is deployed in a server where the target cloud terminal is located, and multiple cloud terminals are set in the server. At this time, the network port is used to characterize network connection ports for multiple cloud terminals deployed in the server where the agent service container is located.

[0050] In this implementation, the execution subject, in response to determining that the multiple cloud terminals include the target cloud terminal, establishes the secure interaction channel between the target cloud terminal and the debugging terminal

through the agent service container according to the network port.

[0051] As an example, the multiple cloud terminals have respective terminal identifiers, and the execution subject may match the terminal identifier of the target cloud terminal with the respective terminal identifiers of the multiple cloud terminals, and determine that the multiple cloud terminals include the target cloud terminal in response to determining that the multiple terminal identifiers corresponding to the multiple cloud terminals include the terminal identifier of the target cloud terminal.

[0052] In this implementation, before establishing the secure interaction channel, it is necessary to confirm that the target cloud terminal is a cloud terminal corresponding to the agent service container, so as to accurately connect to the target cloud terminal, which avoids resource waste and security risks caused by mistakenly connecting to other cloud terminals; facilitates reasonable planning and allocation of network resources, and performs refined management of network bandwidth and the like according to the requirements of cloud terminals, thereby improving resource utilization.

[0053] In some optional implementations of this embodiment, the execution subject may perform the establishment process of the secure interaction channel in the following manner: first, determine a cloud terminal whitelist corresponding to a target object to which the debugging terminal belongs; then, in response to determining that the cloud terminal whitelist includes the target cloud terminal, establish the secure interaction channel between the target cloud terminal and the debugging terminal through the agent service container according to the network port.

[0054] The target object to which the debugging terminal belongs is generally a purchasing user of the cloud terminal. For each target object, a corresponding cloud terminal whitelist may be set, for example, the cloud terminal whitelist includes multiple cloud terminals purchased by the user.

[0055] As an example, the multiple cloud terminals in the cloud terminal whitelist have respective terminal identifiers, and the execution subject may match the terminal identifier of the target cloud terminal with the terminal identifiers of the cloud terminals in the cloud terminal whitelist, and determine that the cloud terminal whitelist includes the target cloud terminal in response to determining that the terminal identifiers of the cloud terminals in the cloud terminal whitelist include the terminal identifier of the target

cloud terminal.

[0056] In this implementation, the cloud terminal whitelist mechanism restricts the debugging terminal to access only authorized cloud terminals, thereby preventing unauthorized access and enhancing security. This mechanism ensures accurate connection between the debugging terminal and the correct cloud terminal, avoiding misoperations and improving debugging accuracy. Furthermore, centralized maintenance of the whitelist simplifies permission management and facilitates access control.

[0057] In some optional implementations of this embodiment, the execution subject may perform the establishment process of the secure interaction channel in the following manner: first, establish an external network interaction channel between the agent service container and the debugging terminal through the agent service container according to the network port; then, establish the secure interaction channel by combining the external network interaction channel and an internal network interaction channel between the agent service container and the target cloud terminal.

[0058] As an example, the debugging terminal sends a connection request to the agent service container, specifying the network port of the agent service container. After receiving the request, the agent service container verifies the request according to a preset security policy (such as an authentication method based on a key or a certificate), and if the verification is passed, an external network interaction channel is established between the debugging terminal and the agent service container.

[0059] The agent service container searches for the intranet IP address of the corresponding target cloud terminal in the cloud terminal whitelist according to the target cloud terminal identifier information carried in the request of the debugging terminal. Then, the agent service container takes advantage of being in the same intranet as the target cloud terminal to actively establish an internal network interaction channel with the target cloud terminal.

[0060] Finally, the agent service container coordinates the external network interaction channel and the internal network interaction channel to forward the requests of the debugging terminal to the target cloud terminal and return the responses of the target cloud terminal to the debugging terminal, thereby establishing a secure interaction channel between the debugging terminal and the target cloud terminal.

[0061] This implementation provides a specific way of establishing a secure interaction channel, which skillfully integrates the connection advantages of the external network and the internal network, can not only ensure the safe and stable communication between the debugging terminal and the target cloud terminal, but also effectively hide the intranet information of the cloud terminal and reduce security risks. At the same time, this way simplifies network configuration, makes remote debugging more convenient and efficient, improves user experience, and ensures the security and reliability of data transmission.

[0062] Step 202 includes: establishing a debugging channel between the target cloud terminal and the debugging terminal on the basis of the secure interaction channel.

[0063] In this embodiment, the execution subject may establish a debugging channel between the target cloud terminal and the debugging terminal on the basis of the secure interaction channel. The debugging channel is, for example, an ADB channel, a VNC (Virtual Network Computing) channel, an RDP (Remote Desktop Protocol) channel.

[0064] Taking the ADB channel as an example, the establishment includes: first, ensuring that the ADB service is enabled on the target cloud terminal, and obtaining its ADB port number (typically 5555 by default) and intranet IP address; installing the ADB client tool on the debugging terminal, verifying that the client tool operates normally and supports network connections to ADB devices; and next, opening a command-line terminal on the debugging terminal and execute the command "adb connect localhost:local_port", where local_port represents the local listening port on the debugging terminal previously configured in the SSH channel. Upon successful connection, the ADB client on the debugging terminal displays device information of the target cloud terminal, indicating successful establishment of the ADB debugging channel. Debugging operations may then be performed on the cloud terminal using ADB commands from the debugging terminal.

[0065] Step 203 includes: transmitting debugging data between the target cloud terminal and the debugging terminal based on the debugging channel.

[0066] In this embodiment, the execution subject may transmit debugging data between the target cloud terminal and the debugging terminal based on the debugging channel, where the debugging data is, for example, log data such as system logs and

application logs, application program data, debugging instructions and response data, and various other data that may be involved in the debugging process.

[0067] In an example scenario, a developer deploys a mobile application on a cloud mobile phone and needs to debug its functions, and the debugging process is as follows.

5 **[0068]** A developer establishes an ADB debugging channel between a debugging terminal and a cloud phone through a secure interaction channel. The developer uses an ADB command to install an application onto the cloud phone and starts the application. On the debugging terminal, the developer sends debugging instructions to the application through an ADB command or a debugging tool (such as Android Studio), for example, 10 setting breakpoints, executing code step by step, viewing variable values, and the like. The cloud phone runs the application and executes corresponding operations according to the debugging instructions, and sends debugging data (such as log information, application status, and the like) back to the debugging terminal through the debugging channel. The developer analyzes, according to the returned debugging data, whether a 15 function of the application is normal, and troubleshoots and fixes problems. Finally, the developer discovers that a logic error exists in a specific function of the application, and after debugging and fixing, the function of the application runs normally on the cloud phone.

20 **[0069]** In another exemplary scenario, an enterprise user uses a cloud computer for graphic design work, feels that performance of the cloud computer is unstable, and needs to perform debugging. The debugging process is as follows:

25 **[0070]** A debugging channel is established between the debugging terminal and the cloud computer through a secure interaction channel (such as using an SSH protocol). A performance monitoring tool is run on the cloud computer to collect performance data, including CPU (Central Processing Unit) usage, memory occupation, disk I/O (Input/Output), network bandwidth, and the like. The collected performance data is transmitted to the debugging terminal through the debugging channel. The debugging terminal analyzes the performance data and discovers that when the cloud computer runs large design software, memory occupation is excessively high, causing system response to slow down. According to an analysis result, resource allocation of the cloud computer 30 is adjusted (such as increasing memory), and performance data is collected and analyzed

again to verify a performance optimization effect. Finally, through optimizing memory configuration, performance of the cloud computer is significantly improved, and the cloud computer can smoothly run graphic design software to meet work requirements of the user.

5 **[0071]** In some optional implementations of this embodiment, the above-mentioned execution subject may execute the foregoing step 202 in the following manner: first, encrypting debugging data through a symmetric key corresponding to the agent service container to obtain encrypted data; and then, transmitting the encrypted data between the target cloud terminal and the debugging terminal based on the debugging channel.

10 **[0072]** After establishing the secure interaction channel, a symmetric key corresponding to the agent service container may be negotiated based on a communication protocol adopted by the secure interaction channel. Taking an SSH secure interaction channel as an example, the SSH protocol automatically negotiates and generates a symmetric key.

15 **[0073]** For a communication process between the debugging terminal and the agent service container, the debugging data is encrypted and decrypted through the symmetric key. Specifically, for debugging data sent from the debugging terminal to the agent service container, the debugging terminal encrypts the debugging data through the symmetric key, the agent service container decrypts the received encrypted data through
20 the symmetric key, and transmits the decrypted debugging data to the target cloud terminal. For debugging data sent from the agent service container to the debugging terminal, after receiving the debugging data of the target cloud terminal, the agent service container encrypts the debugging data through the symmetric key, and transmits the obtained encrypted data to the debugging terminal. After receiving the encrypted
25 data, the debugging terminal decrypts the encrypted data through the symmetric key.

30 **[0074]** In this implementation, symmetric encryption of debugging data can efficiently ensure confidentiality and integrity of data during transmission, prevent information leakage or tampering, enhance security of a debugging process, and since symmetric encryption has small computational overhead, debugging efficiency can be maintained.

[0075] In some optional implementations of this embodiment, the execution subject

may execute an encryption process based on the symmetric key in the following manner: first, determining a target symmetric key corresponding to a target object to which the debugging terminal belongs from multiple symmetric keys corresponding to the agent service container; and then, encrypting the debugging data through the target symmetric key to obtain encrypted data.

[0076] In some cases, cloud terminals of the target objects may be deployed in a same server, and the agent service container in the server includes symmetric keys respectively corresponding to the target objects. The agent service container stores a correspondence relationship between target objects and symmetric keys, and may determine a target symmetric key corresponding to a target object through the correspondence relationship, and implement encryption and decryption of data based on the target symmetric key.

[0077] This implementation can accurately encrypt debugging data for different target objects, enhance debugging security in a multi-user environment, and facilitate unified management and maintenance to ensure confidentiality and integrity of data transmission.

[0078] With continued reference to FIG. 3, which is a schematic diagram 300 of an application scenario of the method for debugging a cloud terminal according to the embodiment. A data center includes multiple servers, where a server 301 is deployed with multiple cloud terminals and an agent service container 3011. A target cloud terminal 3012 among the multiple cloud terminals belongs to a target object 302. When the target object has a debugging requirement for the target cloud terminal, a secure interaction channel 304 between the target cloud terminal and a debugging terminal 303 of the target object is established through the agent service container corresponding to the target cloud terminal; a debugging channel 305 between the target cloud terminal and the debugging terminal is established on the basis of the secure interaction channel; and debugging data is transmitted between the target cloud terminal and the debugging terminal based on the debugging channel.

[0079] In this embodiment, a secure interaction channel between a target cloud terminal and a debugging terminal is established through an agent service container corresponding to the target cloud terminal, so as to establish a debugging channel on the basis of the secure interaction channel for debugging the cloud terminal. This provides a

way for debugging the cloud terminal and improves the convenience and security of the cloud terminal debugging process.

[0080] In some optional implementations of this embodiment, before performing the above step 201, the execution subject may also perform the following operations:

5 **[0081]** first, determining parameter data of created containers in an agent service container set; then, creating the agent service container in the server data center where the target cloud terminal is located according to the parameter data.

10 **[0082]** The parameter data of the created containers is, for example, data such as names and network ports of the created containers. According to the parameter data of the created containers in the agent service container set, the execution subject may clarify the adopted container names and network ports, and when creating a new agent service container in the server data center where the target cloud terminal is located, it is necessary to avoid adopting parameter data consistent with the parameter data of the created containers.

15 **[0083]** This implementation can effectively avoid container parameter conflicts, ensure the uniqueness and independence of each agent service container, ensure smooth communication between the target cloud terminal and the debugging terminal, and improve system stability and management efficiency.

20 **[0084]** In some optional implementations of this embodiment, before performing the above step 201, the execution subject may also perform the following operations: first, determining an asymmetric key pair corresponding to the agent service container, where the asymmetric key pair includes a private key and a public key; then, sending the private key to the debugging terminal and sending the public key to the agent service container; and performing identity authentication on the debugging terminal through the agent service container based on the asymmetric key pair.

25 **[0085]** As an example, after creating the agent service container, a certificate authority or a trusted key management system generates an asymmetric key pair for the agent service container, including a private key and a corresponding public key. Then, the generated private key is sent to the debugging terminal, which needs to keep the private key properly to prevent leakage. The private key is used to sign or encrypt data sent to the agent service container. The public key is sent to the agent service container,

which uses the public key to verify or decrypt data sent by the debugging terminal.

[0086] When the debugging terminal attempts to establish a connection with the agent service container, the debugging terminal uses the private key to sign the authentication request data sent by the debugging terminal to generate a digital signature, and sends the signed data to the agent service container.

[0087] After receiving the request from the debugging terminal, the agent service container verifies the digital signature by using the pre-obtained public key to check the integrity of the data and the authenticity of the source.

[0088] If the verification is passed, it indicates that the identity of the debugging terminal is legal, and the agent service container allows the establishment of a connection and further performs subsequent operations according to service requirements; if the verification fails, the connection request is rejected to ensure that only a legal debugging terminal can communicate with the agent service container.

[0089] In this implementation, the asymmetric key pair is used to ensure the communication security between the debugging terminal and the agent service container. The private key signs or encrypts data locally on the debugging terminal, and the public key verifies the data on the agent service container, which prevents identity forgery, ensures the confidentiality and integrity of data, and improves the security and reliability of identity authentication.

[0090] With continued reference to FIG. 4, a schematic flowchart 400 of another embodiment of the method for debugging a cloud terminal according to the present disclosure is shown. The process 400 includes the following steps.

[0091] Step 401 includes: determining parameter data of created containers in an agent service container set.

[0092] Step 402 includes: creating an agent service container in a server data center where a target cloud terminal is located according to the parameter data.

[0093] Step 403 includes: determining an asymmetric key pair corresponding to the agent service container, where the asymmetric key pair includes a private key and a public key.

[0094] Step 404 includes: sending the private key to a debugging terminal and

sending the public key to the agent service container.

[0095] Step 405 includes: in response to performing identity authentication on the debugging terminal through the agent service container based on the asymmetric key pair and determining that multiple cloud terminals corresponding to the agent service container include the target cloud terminal, determining whether a cloud terminal whitelist corresponding to a target object to which the debugging terminal belongs includes the target cloud terminal.

[0096] Step 406 includes: in response to determining that the multiple cloud terminals include the target cloud terminal, establish a secure interaction channel between the target cloud terminal and the debugging terminal through the agent service container according to a network port corresponding to the agent service container, where the network port is used to characterize network connection ports for the multiple cloud terminals corresponding to the agent service container.

[0097] Step 407 includes: establishing a debugging channel between the target cloud terminal and the debugging terminal on the basis of the secure interaction channel.

[0098] Step 408 includes: encrypting debugging data transmitted between the target cloud terminal and the debugging terminal through a symmetric key corresponding to the agent service container to obtain encrypted data.

[0099] Step 409 includes: transmitting the encrypted data between the target cloud terminal and the debugging terminal based on the debugging channel.

[0100] The process 400 of the method for debugging a cloud terminal in this embodiment specifically describes the establishment process of the secure interaction channel, which helps to further improve the convenience and security of the cloud terminal debugging process.

[0101] With continued reference to FIG. 5, as an implementation of the methods shown in the above figures, the present disclosure provides an embodiment of an apparatus for debugging a cloud terminal. The system embodiment corresponds to the method embodiment shown in FIG. 2, and the system can be specifically applied to various electronic devices.

[0102] As shown in FIG. 5, an apparatus 500 for debugging a cloud terminal

includes:

[0103] a secure channel establishment unit 501 configured to establish a secure interaction channel between a target cloud terminal and a debugging terminal through an agent service container corresponding to the target cloud terminal, where the agent service container is deployed in a server data center where the target cloud terminal is located;

[0104] a debugging channel establishment unit 502 configured to establish a debugging channel between the target cloud terminal and the debugging terminal on the basis of the secure interaction channel; and

[0105] a debugging unit 503 configured to transmit debugging data between the target cloud terminal and the debugging terminal based on the debugging channel.

[0106] In some optional implementations of this embodiment, the secure channel establishment unit 501 is further configured to: determine a network port corresponding to the agent service container, where the network port characterizes a network connection port for a cloud terminal corresponding to the agent service container; and establish the secure interaction channel between the target cloud terminal and the debugging terminal through the agent service container according to the network port.

[0107] In some optional implementations of this embodiment, the network port is used to characterize network connection ports for multiple cloud terminals corresponding to the agent service container, and the secure channel establishment unit 501 is further configured to: in response to determining that the multiple cloud terminals include the target cloud terminal, establish the secure interaction channel between the target cloud terminal and the debugging terminal through the agent service container according to the network port.

[0108] In some optional implementations of this embodiment, the secure channel establishment unit 501 is further configured to: determine a cloud terminal whitelist corresponding to a target object to which the debugging terminal belongs; and in response to determining that the cloud terminal whitelist includes the target cloud terminal, establish the secure interaction channel between the target cloud terminal and the debugging terminal through the agent service container according to the network port.

[0109] In some optional implementations of this embodiment, the secure channel establishment unit 501 is further configured to: establish an external network interaction channel between the agent service container and the debugging terminal through the agent service container according to the network port; and establish the secure interaction channel by combining the external network interaction channel and an internal network interaction channel between the agent service container and the target cloud terminal.

[0110] In some optional implementations of this embodiment, the debugging unit 503 is further configured to: encrypt the debugging data through a symmetric key corresponding to the agent service container to obtain encrypted data; and transmit the encrypted data between the target cloud terminal and the debugging terminal based on the debugging channel.

[0111] In some optional implementations of this embodiment, the debugging unit 503 is further configured to: determine a target symmetric key corresponding to a target object to which the debugging terminal belongs from multiple symmetric keys corresponding to the agent service container; and encrypt the debugging data through the target symmetric key to obtain the encrypted data.

[0112] In some optional implementations of this embodiment, the apparatus further includes a container creation unit (not shown in the figure) configured to: determine parameter data of created containers in an agent service container set; and create the agent service container in the server data center where the target cloud terminal is located according to the parameter data.

[0113] In some optional implementations of this embodiment, the apparatus further includes an identity authentication unit (not shown in the figure) configured to: determine an asymmetric key pair corresponding to the agent service container, where the asymmetric key pair includes a private key and a public key; send the private key to the debugging terminal and send the public key to the agent service container; and perform identity authentication on the debugging terminal through the agent service container based on the asymmetric key pair.

[0114] In this embodiment, an apparatus for debugging a cloud terminal is provided. A secure interaction channel between a target cloud terminal and a debugging terminal is

established through an agent service container corresponding to the target cloud terminal, so as to establish a debugging channel on the basis of the secure interaction channel for debugging the cloud terminal. This provides a way for debugging the cloud terminal and improves the convenience and security of the cloud terminal debugging process.

5 **[0115]** According to the embodiments of the present disclosure, the present disclosure further provides an electronic device, which includes: at least one processor; and a memory communicatively connected to the at least one processor; where the memory stores instructions executable by the at least one processor, and the instructions, when executed by the at least one processor, enable the at least one processor to implement the method for debugging a cloud terminal described in any of the above embodiments.

10 **[0116]** According to the embodiments of the present disclosure, the present disclosure further provides a readable storage medium storing computer instructions, where the computer instructions are used to cause a computer to implement the method for debugging a cloud terminal described in any of the above embodiments when executed.

15 **[0117]** An embodiment of the present disclosure provides a computer program product, which, when executed by a processor, can implement the method for debugging a cloud terminal described in any of the above embodiments.

20 **[0118]** FIG. 6 shows a schematic block diagram of an example electronic device 600 that can be used to implement the embodiments of the present disclosure. The electronic device is intended to represent various forms of digital computers, such as laptop computers, desktop computers, workstations, personal digital assistants, servers, blade servers, mainframe computers, and other suitable computers. The electronic device may also represent various forms of mobile devices, such as personal digital processors, cellular phones, smart phones, wearable devices, and other similar computing devices.

25 The components shown herein, their connections and relationships, and their functions are merely examples, and are not intended to limit the implementation of the present disclosure described and/or claimed herein.

30 **[0119]** As shown in FIG. 6, the device 600 includes a computing unit 601, which can perform various appropriate actions and processes according to a computer program stored in a read-only memory (ROM) 602 or a computer program loaded from a storage unit 608 into a random access memory (RAM) 603. Various programs and data required

for the operation of the device 600 can also be stored in the RAM 603. The computing unit 601, the ROM 602 and the RAM 603 are connected to each other through a bus 604. An input/output (I/O) interface 605 is also connected to the bus 604.

[0120] Multiple components in the device 600 are connected to the I/O interface 605, including: an input unit 606 such as a keyboard, a mouse and the like; an output unit 607 such as various types of displays, speakers and the like; a storage unit 608 such as a magnetic disk, an optical disk and the like; and a communication unit 609 such as a network card, a modem, a wireless communication transceiver and the like. The communication unit 609 allows the device 600 to exchange information/data with other devices through a computer network such as the Internet and/or various telecommunication networks.

[0121] The computing unit 601 may be various general-purpose and/or special-purpose processing components with processing and computing capabilities. Some examples of the computing unit 601 include, but are not limited to, a central processing unit (CPU), a graphics processing unit (GPU), various dedicated artificial intelligence (AI) computing chips, various computing units running machine learning model algorithms, a digital signal processor (DSP), and any appropriate processor, controller, microcontroller, and the like. The computing unit 601 executes the various methods and processes described above, such as the method for debugging a cloud terminal. For example, in some embodiments, the method for debugging a cloud terminal may be implemented as a computer software program tangibly embodied in a machine-readable medium, such as the storage unit 608. In some embodiments, part or all of the computer program may be loaded and/or installed on the device 600 via the ROM 602 and/or the communication unit 609. When the computer program is loaded into the RAM 603 and executed by the computing unit 601, one or more steps of the method for debugging a cloud terminal described above may be executed. Alternatively, in other embodiments, the computing unit 601 may be configured to perform the method for debugging a cloud terminal by any other appropriate means (for example, by means of firmware).

[0122] Various implementations of the systems and technologies described herein may be implemented in digital electronic circuit systems, integrated circuit systems, field programmable gate arrays (FPGAs), application specific integrated circuits (ASICs), application specific standard products (ASSPs), systems on chips (SOCs), complex

programmable logic devices (CPLDs), computer hardware, firmware, software, and/or combinations thereof. These various implementations may include: being implemented in one or more computer programs that can be executed and/or interpreted on a programmable system including at least one programmable processor, which may be a special-purpose or general-purpose programmable processor that can receive data and instructions from a storage system, at least one input device, and at least one output device, and transmit data and instructions to the storage system, the at least one input device, and the at least one output device.

[0123] Program code for implementing the methods of the present disclosure may be written in any combination of one or more programming languages. These program codes may be provided to a processor or controller of a general-purpose computer, a special-purpose computer, or other programmable apparatus for debugging a cloud terminal, such that the program codes, when executed by the processor or controller, cause the functions/operations specified in the flowcharts and/or block diagrams to be implemented. The program code may be executed entirely on a machine, partially on a machine, as a stand-alone software package partially on a machine and partially on a remote machine, or entirely on a remote machine or server.

[0124] In the context of the present disclosure, a machine-readable medium may be a tangible medium that can contain or store a program for use by or in connection with an instruction execution system, apparatus or device. The machine-readable medium may be a machine-readable signal medium or a machine-readable storage medium. The machine-readable medium may include, but is not limited to, an electronic, magnetic, optical, electromagnetic, infrared, or semiconductor system, apparatus or device, or any suitable combination of the above. More specific examples of the machine-readable storage medium will include an electrical connection based on one or more wires, a portable computer disk, a hard disk, a random access memory (RAM), a read-only memory (ROM), an erasable programmable read-only memory (EPROM or flash memory), an optical fiber, a portable compact disk read-only memory (CD-ROM), an optical storage device, a magnetic storage device, or any suitable combination of the above.

[0125] To provide interaction with a user, the systems and technologies described herein may be implemented on a computer having: a display device (for example, a CRT

(cathode ray tube) or LCD (liquid crystal display) monitor) for displaying information to the user; and a keyboard and a pointing device (for example, a mouse or a trackball) through which the user can provide input to the computer. Other types of devices may also be used to provide interaction with the user; for example, the feedback provided to the user may be any form of sensory feedback (for example, visual feedback, auditory feedback, or tactile feedback); and input from the user may be received in any form, including acoustic input, speech input, or tactile input.

[0126] The systems and technologies described herein may be implemented in a computing system including a back-end component (for example, as a data server), or a computing system including a middleware component (for example, an application server), or a computing system including a front-end component (for example, a user computer with a graphical user interface or a web browser through which the user can interact with the implementation of the systems and technologies described herein), or a computing system including any combination of such back-end components, middleware components, or front-end components. The components of the system may be interconnected by any form or medium of digital data communication (for example, a communication network). Examples of communication networks include: a local area network (LAN), a wide area network (WAN) and the Internet.

[0127] The computer system may include a client and a server. The client and server are generally remote from each other and typically interact through a communication network. The relationship of client and server arises by virtue of computer programs running on the respective computers and having a client-server relationship to each other. The server may be a cloud server, also known as a cloud computing server or a cloud host, which is a host product in the cloud computing service system to solve the defects of high management difficulty and weak business scalability in traditional physical hosts and virtual private server (VPS) services; it may also be a server of a distributed system, or a server combined with a blockchain.

[0128] According to the technical solution of the embodiments of the present disclosure, a method and an apparatus for debugging a cloud terminal are provided. A secure interaction channel between a target cloud terminal and a debugging terminal is established through an agent service container corresponding to the target cloud terminal, so as to establish a debugging channel on the basis of the secure interaction channel for

debugging the cloud terminal. This provides a mode for debugging the cloud terminal and improves the convenience and security of the cloud terminal debugging process.

[0129] It should be understood that the steps of reordering, adding or deleting may be performed using the various forms shown above. For example, the steps described in the present disclosure may be performed in parallel or sequentially or in a different order, so long as the desired results of the technical solution disclosed in the present disclosure can be realized, and no limitation is imposed herein.

[0130] The foregoing detailed description is not intended to limit the scope of the present disclosure. It will be appreciated by those skilled in the art that various modifications, combinations, sub-combinations, and substitutions may be made depending on design requirements and other factors. Any modifications, equivalents, and modifications that fall within the spirit and principles of the disclosure are intended to be included within the scope of protection of the disclosure.

WHAT IS CLAIMED IS:

1. A method for debugging a cloud terminal, comprising:

5 establishing a secure interaction channel between a target cloud terminal and a debugging terminal through an agent service container corresponding to the target cloud terminal, wherein the agent service container is deployed in a server data center where the target cloud terminal is located;

establishing a debugging channel between the target cloud terminal and the debugging terminal based on the secure interaction channel; and

10 transmitting debugging data between the target cloud terminal and the debugging terminal based on the debugging channel.

2. The method according to claim 1, wherein the establishing the secure interaction channel between the target cloud terminal and the debugging terminal through the agent service container corresponding to the target cloud terminal comprises:

15 determining a network port corresponding to the agent service container, wherein the network port characterizes a network connection port for a cloud terminal corresponding to the agent service container; and

establishing the secure interaction channel between the target cloud terminal and the debugging terminal through the agent service container according to the network port.

3. The method according to claim 2, wherein the network port is used to characterize network connection ports for a plurality of cloud terminals corresponding to the agent service container, and

25 the establishing the secure interaction channel between the target cloud terminal and the debugging terminal through the agent service container according to the network port comprises:

in response to determining that the plurality of cloud terminals include the target cloud terminal, establishing the secure interaction channel between the target cloud terminal and the debugging terminal through the agent service container according to the network port.

5 4. The method according to claim 2, wherein the establishing the secure interaction channel between the target cloud terminal and the debugging terminal through the agent service container according to the network port comprises:

 determining a cloud terminal whitelist corresponding to a target object to which the debugging terminal belongs; and

10 in response to determining that the cloud terminal whitelist includes the target cloud terminal, establishing the secure interaction channel between the target cloud terminal and the debugging terminal through the agent service container according to the network port.

15 5. The method according to any one of claims 2-4, wherein the establishing the secure interaction channel between the target cloud terminal and the debugging terminal through the agent service container according to the network port comprises:

 establishing an external network interaction channel between the agent service container and the debugging terminal through the agent service container according to the network port; and

20 establishing the secure interaction channel by combining the external network interaction channel and an internal network interaction channel between the agent service container and the target cloud terminal.

25 6. The method according to claim 1, wherein the transmitting debugging data between the target cloud terminal and the debugging terminal based on the debugging channel comprises:

encrypting the debugging data through a symmetric key corresponding to the agent service container to obtain encrypted data; and

transmitting the encrypted data between the target cloud terminal and the debugging terminal based on the debugging channel.

5

7. The method according to claim 6, wherein the encrypting the debugging data through the symmetric key corresponding to the agent service container to obtain encrypted data comprises:

10 determining a target symmetric key corresponding to a target object to which the debugging terminal belongs from a plurality of symmetric keys corresponding to the agent service container; and

encrypting the debugging data through the target symmetric key to obtain the encrypted data.

15

8. The method according to claim 1, wherein before establishing the secure interaction channel between the target cloud terminal and the debugging terminal through the agent service container corresponding to the target cloud terminal, the method further comprises:

determining parameter data of created containers in an agent service container set; and

20

creating the agent service container in the server data center where the target cloud terminal is located according to the parameter data.

25

9. The method according to claim 1 or 8, wherein before establishing the secure interaction channel between the target cloud terminal and the debugging terminal through the agent service container corresponding to the target cloud terminal, the method further comprises:

determining an asymmetric key pair corresponding to the agent service container,

wherein the asymmetric key pair comprises a private key and a public key;

sending the private key to the debugging terminal and sending the public key to the agent service container; and

performing identity authentication on the debugging terminal through the agent service container based on the asymmetric key pair.

10. An apparatus for debugging a cloud terminal, comprising:

a secure channel establishment unit configured to establish a secure interaction channel between a target cloud terminal and a debugging terminal through an agent service container corresponding to the target cloud terminal, wherein the agent service container is deployed in a server data center where the target cloud terminal is located;

a debugging channel establishment unit configured to establish a debugging channel between the target cloud terminal and the debugging terminal based on the secure interaction channel; and

a debugging unit configured to transmit debugging data between the target cloud terminal and the debugging terminal based on the debugging channel.

11. An electronic device, comprising:

at least one processor; and

a memory communicatively connected to the at least one processor;

wherein the memory stores instructions executable by the at least one processor, and the instructions, when executed by the at least one processor, cause the at least one processor to perform the method according to any one of claims 1-9.

12. A non-transitory computer-readable storage medium storing computer instructions,

wherein the computer instructions are used to cause a computer to perform the method according to any one of claims 1-9.

5 13. A computer program product, comprising a computer program, wherein the computer program, when executed by a processor, implements the method according to any one of claims 1-9.