

SECURED SYSTEM, METHOD AND A COMPUTER READABLE MEDIUM FOR DEPLOYING MESSAGE-ORIENTED MIDDLEWARE FOR ONE OR MORE MICROSERVICES

TECHNICAL FIELD

[0001] The present disclosure relates generally to a system, a method and a computer readable medium for deploying a message-oriented middleware under one or more microservices. More particularly, embodiments of the disclosure are related to the system and the method that support decoupling of the application with the middleware and isolation of data in the system to enhance the security of data.

BACKGROUND

[0002] Modern approaches to integrating distributed systems almost invariably rely on middleware. Middleware take many forms, ranging from distributed object brokers, to message-oriented middleware and enterprise service buses. Message-oriented middleware refers to a programmatic infrastructure that supports the exchange of messages between distributed computing systems. At present, most of the systems heavily depend on one specific middleware to fulfil its function(s). This results in a high degree of coupling between the system and the middleware as the coders pay more emphasis on coding with respect to a particular middleware. However, should there be a need to use or incorporate a different middleware, in view of technology update and so on, adaptation or upgradation to a new or a different middleware may incur huge labor and time cost, as well as other potential risks. For instance, a system that wishes to change from one middleware to another, for example from originally Kafka message-oriented middleware to RabbitMQ, would require massive changes in the code.

[0003] There are many kinds of message-oriented middleware in the market, and the ways of using different middleware are quite varied, which is bound to bring huge training and maintenance costs. The dependence of a system on a specific type of middleware results in huge labor cost, when a new middleware is required to be

implemented.

[0004] In addition, with significant increase in system concurrency, microservices architecture is becoming more and more popular. Multiple sets of message-oriented middleware clusters need to be deployed in a typical microservices architecture, so as to achieve message and authority isolation between different microservices. However, in many cases, messages between microservices with different business functions and responsibilities often share the same message-oriented middleware cluster resulting in messages between different microservices being integrated or deisolated and thereby, greatly increasing security risks in communications.

[0005] A need, therefore, exists for an improved system, method and a computer readable medium that support decoupling of the application with the middleware and isolation of data in the system to enhance the security of data.

SUMMARY

[0006] The specification describes a system, a method and a computer readable medium for deploying message-oriented middleware under one or more microservices architecture securely. The methods, systems and computer readable mediums disclosed in the present application are advantageous in several ways, including but not limited to, deploying a message communication model supported by all middleware and easily adaptable to different kinds of middleware, reducing the degree of coupling between the system and message-oriented middleware, reducing the labor and time cost in the event of technology upgradation, reducing the complexity of system deployment and the number of services required, message and authority isolation between different microservices resulting in enhanced communication security of the microservices system and reducing the training costs and risk during the project development stage.

[0007] It is, therefore, one object of the present invention to provide a system, a method and a computer readable medium that supports switching between different message-oriented middleware with minimum modification and isolates messages

between microservices of different business modules and function modules.

[0008] It is further an object of the present invention, to provide a system, a method and a computer readable medium for deploying message-oriented middleware under one or more microservices that reduce the coupling degree between the system and message-oriented middleware under one or more microservices, thus reducing the labor and time cost in the event of technology upgradation and reducing the training costs during the project development stage.

[0009] It is further an object of the present invention to provide a system, a method and a computer readable medium for deploying message-oriented middleware under one or more microservices that reduce the complexity of system deployment and the number of servers required.

[0010] It is further an object of the present invention to provide a system, a method and a computer readable medium for deploying message-oriented middleware under one or more microservices that separate different business modules at the level of message-oriented middleware to improve system security.

[0011] In a first aspect of the present disclosure, a system for deploying message-oriented middleware under one or more microservices is disclosed. The system comprises a microservice having one or more of: a business module, a functional module or a combination thereof. The system further comprises a middleware server cluster for one or more of: a business module, a functional module or a combination thereof, and said middleware server cluster comprises one or more creator for creating at least one specific theme formed under a big theme for messaging of each microservice, and one or more consumers subscribed to one or more specific themes. The system further comprises an internal service cluster including one or more internal services for accessing one or more of: the business module, the functional module or the combination thereof, an external service cluster including one or more external components for accessing one or more of: the business module, the functional module or the combination thereof, and a connectivity cluster including one or more

connectivity component for connecting the internal services and the external components, and wherein the internal services of one or more microservices send a message to the external components through the connectivity component, using the specific theme of the big theme.

[0012] According to an embodiment in conjunction to the first aspect of the present disclosure, each of the big theme is at least one business topic.

[0013] According to an embodiment in conjunction to the first aspect of the present disclosure, each of the specific theme is a special business function.

[0014] According to an embodiment in conjunction to the first aspect of the present disclosure, the creator in the business modules or the functional modules creates one specific theme.

[0015] According to an embodiment in conjunction to the first aspect of the present disclosure, each of the specific theme has at least one channel for communication.

[0016] According to an embodiment in conjunction to the first aspect of the present disclosure, the channel for communication is defined in a configuration file.

[0017] According to an embodiment in conjunction to the first aspect of the present disclosure, each specific theme and the big theme are defined in the configuration file.

[0018] In a second aspect of the present disclosure, a method for deploying message-oriented middleware under one or more microservices is disclosed. The method comprises splitting each microservice into one or more of: a business module, a functional module or a combination thereof; implementing a functional level middleware cluster, including: creating, three middleware clusters or virtual hosts for at least one of an internal service, a connectivity component and an external component, creating, at least one user on each cluster or virtual host, and sending, a message from the internal services to the connectivity component and the connectivity component redirects the messages to the specific external components; and implementing middleware server clusters under business level which includes the

following steps: creating at least one specific theme formed under a big theme for messaging of one or more microservices using one or more creator; and allowing one or more consumers to subscribe in one or more specific themes.

[0019] According to an embodiment in conjunction to the second aspect of the present disclosure, implementing functional level middleware clusters includes sending a message from the external components to the connectivity component and the connectivity component redirecting the message to the internal services.

[0020] According to an embodiment in conjunction to the second aspect of the present disclosure, each of the big theme is at least one business topic.

[0021] According to an embodiment in conjunction to the second aspect of the present disclosure, each of the specific theme is a special business function.

[0022] According to an embodiment in conjunction to the second aspect of the present disclosure, the creator in the business or functional modules creates one specific theme.

[0023] According to an embodiment in conjunction to the second aspect of the present disclosure, each of the specific theme has at least one channel for communication.

[0024] According to an embodiment in conjunction to the second aspect of the present disclosure, the channel for communication is defined in a configuration file.

[0025] According to an embodiment in conjunction to the second aspect of the present disclosure, each specific theme and the big theme are defined in the configuration file.

[0026] In a third aspect of the present disclosure, a computer readable medium storing a program comprising instructions that, when executed by a system, cause the system to perform the method for deploying message-oriented middleware under one or more microservices is disclosed. The computer readable medium comprises

splitting each microservice into one or more of: a business module, a functional module or a combination thereof; implementing a functional level middleware cluster, including: creating three middleware clusters or virtual hosts for at least one of an internal service, a connectivity component and an external component, creating at least one user on each cluster or virtual host, and sending a message from the internal services to the connectivity component and the connectivity component redirects the messages to the specific external components; and implementing middleware server clusters under business level which includes the following steps: creating at least one specific theme formed under a big theme for messaging of one or more microservices using one or more creator, and allowing one or more consumers to subscribe in one or more specific themes.

[0027] According to an embodiment in conjunction to the third aspect of the present disclosure, implementing functional level middleware clusters includes sending messages from the external components to the connectivity component and the connectivity component redirecting the messages to the internal services.

[0028] According to an embodiment in conjunction to the third aspect of the present disclosure, each of the big theme is at least one business topic.

[0029] According to an embodiment in conjunction to the third aspect of the present disclosure, each of the specific theme is a special business function.

[0030] According to an embodiment in conjunction to the third aspect of the present disclosure, the creator in the business or functional modules creates one specific theme.

[0031] According to an embodiment in conjunction to the third aspect of the present disclosure, each of the specific theme has at least one channel for communication.

[0032] According to an embodiment in conjunction to the third aspect of the present disclosure, the channel for communication is defined in a configuration file.

[0033] According to an embodiment in conjunction to the third aspect of the present disclosure, each specific theme and the big theme are defined in the configuration file.

[0034] The following disclosure is provided to facilitate an understanding of some of the innovative features unique to the disclosed embodiment in order to achieve the above-mentioned objectives and is not intended to be a full description. A full appreciation of the various aspects of the embodiments disclosed herein can be gained by taking into consideration the entire specification, claims, drawings, and abstract as a whole.

BRIEF DESCRIPTION OF THE DRAWINGS

[0035] **FIG. 1** is a block diagram depicting an example microservices system **10**, in accordance with the disclosed embodiments;

[0036] **FIG. 2** is a schematic diagram depicting an example concept **100** of a big theme **114** and specific themes **108**, **110**, **112**, **116**, **118**, **120**, in accordance with the disclosed embodiments;

[0037] **FIG. 3** is a flow diagram **170** that illustrates an example process to achieve the big and specific theme concept **100** of **FIG. 2**, in accordance with the disclosed embodiments;

[0038] **FIG. 4** is a schematic diagram disclosing big themes **214**, **216**, forming a middleware server cluster **32**, in accordance with the disclosed embodiments;

[0039] **FIG. 5** is a schematic diagram disclosing an example messaging communication model **400** used with a message-oriented middleware, in accordance with the disclosed embodiments;

[0040] **FIG. 6** is a schematic diagram of a functional level system **440** showing

isolation of data at a functional level to maintain data security, in accordance with another embodiment of the present disclosure; and

[0041] FIG. 7 is a flow diagram 700 that illustrates an example process of deploying a message-oriented middleware under one or more microservices, in accordance with the disclosed embodiments.

[0042] FIG. 8 is a flow diagram 800 that illustrates an example process of implementing a middleware server cluster, in accordance with the disclosed embodiments.

[0043] FIG. 9 is a flow diagram 900 that illustrates an example process of implementing functional level middleware clusters, in accordance with the disclosed embodiments.

[0044] The summary above, as well as the following detailed description of illustrative embodiments, is better understood when read in conjunction with the appended drawings. For the purpose of illustrating the present disclosure, exemplary constructions of the disclosure are shown in the drawings. However, the disclosure is not limited to specific methods and instrumentalities disclosed herein. Moreover, those in the art will understand that the drawings are not to scale. Wherever possible, like elements have been indicated by identical numbers.

DETAILED DESCRIPTION OF ILLUSTRATIVE EMBODIMENTS

[0045] The particular configurations discussed in the following description are non-limiting examples that can be varied and are cited merely to illustrate at least one embodiment and are not intended to limit the scope thereof.

[0046] Microservices architecture is a service-oriented architecture in which applications are built as loosely packed service modules. In the present invention each microservice (also referred as application) has one or more business services, referred as one or more business module. For example, the business modules in an enterprise

system include but are not limited to, order management, billing management, and so on. Each microservice also has one or more functional module that defines set of activities, roles and responsibilities of various users of the system. Business services of an application are defined in terms of the business modules, while the internal behavior of the application, that is, application functions that realize these business services are defined in terms of function modules. In an example, in a financial services application, business modules may include accounting, billing and so on, and function modules may include functions such as displaying accounts status, creating invoices, sending invoices and so on. Each business/functional module is referred using a business topic. The business topic is generally referred as big theme. Each business module has one or more sub modules. The sub modules are referred as one or more specific business functions or special business function. The specific business functions are under one business topic or big theme. The specific business functions are generally referred as specific themes. In different middleware, business topics and the specific business functions are referred using different names. For example, in RabbitMQ middleware, an “exchange” is configured to route messages to different message “queues” and makes the forwarding decision based on “routing_key” and “binding_key”. Thus, if middleware RabbitMQ is used in the present invention, the “exchange” (big theme) is the business topic and the “queue” (specific theme) is the special business function of the business topic.

[0047] FIG. 1 is a block diagram depicting an example microservices system **10**, in accordance with the disclosed embodiments. In the microservices system/ ecosystem **10**, microservices **21, 22** (also referred to as applications) are split as business or function modules. For example, as disclosed in FIG. 1, the service **21** has business modules **12, 14** and functional modules **16, 18**. The service **22** is split as business modules **24, 26** and functional modules **28, 30**. The system **10** also has middleware server clusters **32**. The cluster **32** is the middleware server cluster, which is the carrier of message elements. The cluster **32** contains one or more big themes or one or more specific themes or a combination thereof. In the middleware server clusters **32**, the business topic of an application will be the big theme. The independent application will focus on same business topic, that is, the big theme. The application has a set of some specific business functions under the same topic and each of the business functions will have a specific theme. The big theme receives messages from message

producers or senders or creators and then routes the messages to the specific themes. The specific theme is the special business function of the business topic. It receives messages from the business theme and then delivers said messages to message listeners or consumers.

[0048] The system **10** also has functional level middleware clusters **34** for communicating with functional level systems. A configuration file **33** is used by the system **10** to read the middleware connection information, big theme and specific theme information. This information is already defined in the configuration file **33** and is used at the time of messaging. It should be noted that the big themes and specific themes are under same middleware server clusters **32**. The big theme consists of several specific themes and is a set of said specific themes. Thus, a big theme pertains to each service **21** and **22** and specific themes pertains to the business or function modules **12, 14, 16, 18, 24, 26, 28, 30**.

[0049] **FIG. 2** is a schematic diagram depicting an example concept **100** of a big theme **114** and specific themes **108, 110, 112, 116, 118, 120**, in accordance with the disclosed embodiments. As explained above, in a microservices ecosystem, services are split by business or function modules, which means each microservice is a set of business or function. Each service has a big theme, while each business or function module has their own specific theme. The big theme and the specific themes are in same middleware server cluster and the big theme is a parent/ set of the specific themes. Each specific theme **108, 110, 112** has a message producer or sender or creator **102, 104, 106** and each specific theme **116, 118, 120** has a message listener or consumer **122, 124, 126**. Message producers or senders or creators **102, 104, 106** create a message pertaining to specific theme 1, 2 and 3, respectively, and send the message to the big theme **114**. The message listeners or consumers **122, 124, 126** subscribe from the big theme **114** for one or more specific themes, and the big theme routes the message relating to the relevant specific theme to that message listener or consumer.

[0050] As disclosed in **FIG. 2**, consumer **122** subscribes to specific theme (1), **116**; consumer **124** subscribes to specific themes (1 and 2) **118**; and consumer **126** subscribes to all the specific themes (1, 2 and 3) **120**.

[0051] For example, there is a service by the name of “order management service”. Said service has two main functions: “order workflow execution” and “order information synchronization”. In this service, big theme may be OMS (“order management service”). Further, the specific themes for the above-mentioned functions may be “OMS.order_workflow_execution” and “OMS.order_information_synchronization”. Both the two specific themes are under and subsets of “OMS” that is, the big theme. In this regard, the message producers or senders or creators generates a message with a specific theme that is, “OMS.order_workflow_execution” or “OMS.order_information_synchronization” while listeners or consumers may subscribe to one or both of said themes depending on their interest. Thus, at the coding level, time is not spent on coding to a specific middleware feature, but the themes of the messages are the only focus. Thus, the present embodiment decouples the middleware with the system at the code level, thereby enabling said system to support different middleware, as need be. In this regard, the present embodiment can switch between different message-oriented middleware with minimum modification in the configuration file and the messages between the microservices of different business and function modules in the system are completely isolated.

[0052] Thus, applying the above example in **FIG. 2**, “Order management System (OMS) may be the big theme **114** with two main functions of ‘order workflow execution’ and ‘order information synchronization’. Accordingly, the specific theme, for example **108**, under OMS may be, “OMS.order_workflow_execution” and the specific theme **110** may be “OMS.order_information_synchronization”. The above-mentioned specific themes **108** and **110** are subsets of the big theme OMS. In this regard, message producers or senders or creators **102** and **104** generate a message with a specific theme **108** and **110**, while consumers **122**, **124** and **126** may subscribe to one or more of said themes based on their interest.

[0053] It should be noted that in the present embodiment, in code, the specific feature of the middleware itself is ignored, and communication only takes place in “channel”, where one specific theme will have one channel. In this regard, the channel is defined in the configuration file in, for example, YAML Ain't Markup Language (YAML). In the

present embodiment, the middleware does not deal with the themes. The present embodiment uses the specific feature of the middleware to implement the theme structure. For example, for RabbitMQ middleware, the “exchange” is considered as big theme **114** and the “queue” is considered as the specific theme **108, 110 or 112**.

[0054] For example, in Kafka middleware, “topics” represent the logical collection of messages that belong to a group and each “topic” is split into one or more “partitions”. Thus, in an example if Kafka middleware is used, the “topic” will be considered as the big theme **114**, and the “partition” is considered as specific theme **108, 110 or 112**.

[0055] **FIG. 3** is a flow diagram **170** that illustrates an example process to achieve the big and specific theme concept, in accordance with the disclosed embodiments. As explained above, in the present embodiment, the middleware does not deal with the themes. The present embodiment uses the specific feature of the middleware to implement the theme structure. Specifically, as illustrated in **FIG. 3**, initially, spring cloud stream dependency is added in a project system at step **171**. Spring Cloud Stream is a framework for building highly scalable event-driven microservices connected with shared messaging systems. Then, the middleware connection information is defined at step **172** in the configuration file and the big and specific themes are defined at step **173** in configuration file. Thereafter, the message producers or senders or creators create a message of the specific theme and send the message to the big theme at step **174**. At step **175**, the message listeners or consumers subscribe from the big theme for one or more specific themes and receive messages from the relevant one or more specific themes. Thus, by the above steps, the message producers or senders or creators and message listeners or consumers are decoupled from each other.

[0056] Each microservice in a microservices architecture has a big theme, while each business or function module has its own specific theme. **FIG. 4** is a schematic diagram disclosing big themes **214, 216**, forming a middleware server cluster **32**, in accordance with the disclosed embodiments. As disclosed above, the cluster **32** is the middleware server cluster, which is the carrier of message elements. The cluster contains one or more big themes, for example big themes **214, 216** in **FIG. 4**. Further,

the cluster **32** contains one or more specific themes, for example specific themes **202, 204, 206, 208, 210, 212** in **FIG. 4**. Specific themes are subsets of the big theme on the cluster. For example, in **FIG. 4**, the specific themes **202, 204** and **206** are under (subsets of) the big theme **214**, similarly, the specific themes **208, 210** and **212** are under (subsets of) the big theme **216**. The big theme **214, 216** is the business topic of each application. The big theme **214, 216** receives messages from message producers or senders or creators and then routes the messages to the specific themes **202, 204, 206, 208, 210, 212**. The specific theme **202, 204, 206, 208, 210, 212** is the special business function of the business topic. The big theme receives messages created by message producers or senders or creators (for example, **102, 104** or **106** in **FIG. 2**) from the specific business theme and then delivers or routes said messages to message listeners or consumers based on their subscription of one or more specific themes (shown as Specific theme 1, 2, 3 in **FIG. 4**). The message model disclosed in the schematic diagram of **FIG. 4** can be adapted for use with different middleware as said model does not rely on the middleware but focuses on the themes of the message only.

[0057] FIG. 5 is a schematic diagram disclosing an example messaging communication model **400** used with a message-oriented middleware, in accordance with the disclosed embodiments. The middleware exemplified in **FIG. 5** is RabbitMQ. As disclosed earlier, according to the present disclosure, the coding will focus on the features corresponding to the “theme” of the middleware, herein RabbitMQ. Thus, in **FIG. 5**, for middleware RabbitMQ, the big theme is “exchange” **468** and the specific themes are “queue” **472, 474, 476**. For middleware RabbitMQ, the big theme (“exchange” **468**) is the business topic and the specific theme (“queue” **472, 474, 476**) is the special business function of the business topic.

[0058] As disclosed earlier, the present disclosure focuses on the themes of the message and not the middleware, and accordingly, “exchange” **468** and “queue” **472, 474, 476** are the themes taken in view of the middleware used. For example, “Routing key” and “binding key” are features found only in RabbitMQ, and both have the same value as the “specific themes”. “Routing key” **462, 464, 466** is used by message producer or sender or creator **452, 454, 456** and “binding key” **471, 473, 475** is used

when message is received by the message consumer **482, 484, 486**, wherein one routing key matches at least one binding key.

[0059] As disclosed in **FIG. 5**, message producers or senders or creators **452, 454, 456** will code a message to exchange **468** with a specific theme that is “RoutingKey” **462, 464, 466**. Exchange **468** in this regard corresponds to the “big theme”, which routes messages to queues **472, 474, 476**. The queues **472, 474, 476** bind to exchange **468** with the “Binding Key” **471, 473, 475**. Binding key is an expression often used to refer to themes and it describes the kind of messages that are subscribed. The messages will be routed to the queues **472, 474, 476**, if the RoutingKey matches the BindingKey.

[0060] In another example, the middleware Kafka may be used wherein the “topic” may be considered as the big theme, and the “partition” may be considered as the specific theme. In another example, a middleware following Java Message Service (JMS) standard may be used wherein, the “topic” may be considered as the big theme and “queue” may be considered as the specific theme. The messages in queues **472, 474, 476** are sent to message consumers **482, 484, 486** depending on their subscriptions.

[0061] **FIG. 6** is a schematic diagram of a functional level system **440** showing isolation of data at a functional level to maintain data security, in accordance with another embodiment of the present disclosure. As discussed above, with the increase in system concurrency, multiple sets of messaging middleware clusters may be needed to be deployed in a typical microservices architecture, in order to achieve message and authority isolation between different microservices. In reality, in many cases, messages between microservices with different business functions and responsibilities often share the same message-oriented middleware cluster, messages between different microservices may not be in isolation and this greatly increases communications security risks.

[0062] The present invention as disclosed in the embodiment of **FIG. 6** makes use of the characteristics of different message-oriented middleware to physically or

logically isolate business modules even on the same message-oriented middleware cluster, so as to achieve communication security of the microservices system.

[0063] As disclosed in **FIG. 6**, three roles are defined in the function level, for example, internal **402**, external **424**, and connectivity component **410**. The internal role **402** is the business owner's system and may have several internal systems, for example, Internal A **404**, Internal B **406** and Internal C **408**. External component or external role **424**, for example, External A **412**, External B **414** and External C **416** are client systems and the connectivity component **410** acts as a bridge between the internal and external systems or services or roles. The different roles use different middleware connections and each of them are physically isolated. In this regard, each cluster may be given separate accounts with different permission levels. For example, in **FIG. 6** the External roles **424** may connect to the middleware with a read-only account or a particular permission level and thereby only read data which has been allowed. The cluster **418** refers to a cluster for internals **402**, the cluster **420** refers to a cluster for connectivity component **410** and the cluster **422** refers to a cluster for externals **424**. The clusters **418**, **420**, **422** are together referred as functional level middleware clusters **34** in **FIG. 1**. The connectivity or connectivity component **410** refers to without limitation, a wired or wireless communication system that is used to communicate between the internal **402** and the external **424** roles. The cluster for connectivity **420** is used for communication between the cluster for internals **418** and cluster for externals **422**.

[0064] It should be noted that the functional level system **440** disclosed in **FIG. 6** can be modified to make use of the characteristics of the different message-oriented middleware to physically or logically isolate business modules even on the same message-oriented middleware clusters so as to achieve communication security of the microservices system. The present invention provides a common communication design for a microservices structured system, so that when the messaging structure is designed, the time spent on the middleware is reduced. By way of the above disclosure, the present invention is able to support different message-oriented middleware, including but not limited to RabbitMQ and Kafka.

[0065] **FIG. 7** is a flow diagram **700** that illustrates an example process of deploying a message-oriented middleware under one or more microservices, in accordance with the disclosed embodiments. As disclosed in **FIG. 7**, the process **700** can be implemented, for example, by splitting each microservice into one or more of a business module, a functional module or a combination thereof at step **701**. In the microservices system or ecosystem, services are split as business or function modules. Then, as disclosed in **FIG. 7**, the process **700** can be implemented, for example, by implementing functional level middleware clusters at step **702**. The system also has functional level middleware clusters for communicating with functional level systems.

[0066] Thereafter, as disclosed in **FIG. 7**, the process **700** can be implemented, for example, at step **703**, by implementing middleware server clusters. The middleware server cluster is the carrier of message elements. A configuration file is used by the system to read the information related to middleware connection information, big theme and specific theme. This information is already defined in the configuration file and is used at the time of messaging. The big theme consists of several specific themes and is a set of said specific themes.

[0067] **FIG. 8** is a flow diagram **800** that illustrates an example process of implementing a middleware server cluster, in accordance with the disclosed embodiments. As disclosed in **FIG. 8**, the process **800** can be implemented, for example, by creating at least one specific theme under a big theme at step **801**. As disclosed above, the cluster contains one or more big themes or one or more specific themes or a combination thereof. In the middleware server clusters, the business topic will be the big theme. The independent application will focus on same business topic that is the big theme. The application has a set of some specific business functions under the same topic and each of the function will be the specific theme. The big theme receives messages from message producers or senders or creators and then routes the messages to the specific themes. The specific theme is the special business function of the business topic. It receives messages from the business theme and then delivers said messages to message listeners or consumers.

[0068] Thereafter, the example process of **FIG. 8** can be implemented by allowing

one or more message listeners or consumers to subscribe to one or more specific themes at step **802**. As explained above, in a microservices ecosystem, services are split by business or function modules, which means each microservice is a set of business or function. Each service has a big theme, while each business or function module has their own specific theme. The big theme and the specific themes are in same middleware server cluster and the big theme is a parent or a set of the specific themes. Each specific theme has a message producer or sender or creator and each specific theme has a message listener or consumer. Message producers or senders or creators create a message pertaining to a specific theme and send the message to the big theme. The message listeners or consumers subscribe from the big theme for one or more specific themes, and the big theme routes the message relating to the relevant specific theme to that listener or consumer.

[0069] **FIG. 9** is a flow diagram that illustrates an example process of implementing functional level middleware clusters **900** in accordance with the disclosed embodiments. As disclosed in **FIG. 9**, the process **900** can be implemented, for example, by creating three middleware clusters or virtual hosts for internal services or internal or internal components, connectivity or connectivity components and external services or external components at step **901**. Then, the users on each in clusters of internal, connectivity component and external are created at step **902**. Thereafter, at step **903**, the internal, connectivity component and external connect to the clusters for internal, connectivity component and external, respectively. In some implementations, optionally, the internal services may send messages to other internal services. Thereafter, as disclosed in **FIG. 9**, the process **900** can be implemented, for example, by the internal services sending messages to connectivity component and then connectivity component redirecting the messages to the specific external components and vice versa at step **904**.

[0070] It should be noted that, generating messages with specific themes, as depicted in **FIG. 2** is used to send to internal **402** or connectivity component **410**. Connectivity component **410** will redirect the messages to external components **412**, **414** and **416**.

[0071] As disclosed above, the present invention supports different messaging middleware thereby saving time at the stage of system design by not according importance to the specific theory of a middleware. The present invention can be modified and used in different message-oriented middleware to physically or logically isolate business modules even on the same message-oriented middleware clusters, so as to achieve communication security of the microservices system.

[0072] Thus, the example process disclosed in **FIG. 9** isolates data at the function level to maintain data security that is internal, connectivity component and external.

[0073] The present invention reduces the coupling degree between the system and message-oriented middleware, thereby reducing the labor cost, training costs and time wastage in the event of technology upgradation. Further, the present invention also reduces the complexity of system deployment and the number of servers required. The present invention also implements separate business modules at the level of message-oriented middleware, thereby improving the security of the system.

[0074] Thus, the present invention is able to provide a solution to design a message structure which can decouple from message-oriented middleware and also isolate data at function level.

[0075] It will be appreciated that variations of the above disclosed and other features and functions, or alternatives thereof, may be desirably combined into many other different systems or applications. Also, various presently unforeseen or unanticipated alternatives, modifications, variations or improvements therein may be subsequently made by those skilled in the art which are also intended to be encompassed by the following claims.

[0076] Although embodiments of the current disclosure have been described comprehensively in considerable detail to cover the possible aspects, those skilled in the art would recognize that other versions of the disclosure are also possible.

CLAIMS

What is claimed is:

1. A system for deploying message-oriented middleware under one or more microservices, comprising:
 - the microservice having one or more of: a business module, a functional module or a combination thereof;
 - a middleware server cluster for one or more of: a business module, a functional module or a combination thereof, wherein the middleware server cluster comprises:
 - one or more creator for creating at least one specific theme formed under a big theme for messaging of each microservice; and
 - one or more consumers subscribed to one or more specific themes;
 - an internal service cluster including one or more internal services for accessing one or more of: the business module, the functional module or the combination thereof;
 - an external service cluster including one or more external components for accessing one or more of: the business module, the functional module or the combination thereof; and
 - a connectivity cluster including one or more connectivity component for connecting the internal services and the external components;

wherein the internal services of one or more microservices send a message to the external components through the connectivity component, using the specific theme of the big theme.
2. The system of claim 1, wherein each of the big theme is at least one business topic.
3. The system of any one of claims 1 or 2, wherein each of the specific theme is a special business function.
4. The system of any one of claims 1 to 3, wherein each of the specific theme has at least one channel for communication.

5. The system of claim 4, wherein the channel for communication is defined in a configuration file.
6. The system of claim 5, wherein each specific theme and the big theme are defined in the configuration file.
7. A method for deploying message-oriented middleware under one or more microservices, comprising:
 - splitting, each microservice into one or more of: a business module, a functional module or a combination thereof;
 - implementing, a functional level middleware cluster, including:
 - creating, three middleware clusters or virtual hosts for at least one of an internal service, a connectivity component and an external component;
 - creating, at least one user on each cluster or virtual host; and
 - sending, a message from the internal services to the connectivity component and the connectivity component redirects the messages to the specific external components.
 - implementing middleware server clusters under business level includes following steps:
 - creating at least one specific theme formed under a big theme for messaging of one or more microservices using one or more creator;
 - and
 - allowing one or more consumers to subscribe in one or more specific themes.
8. The method of claim 7, wherein implementing functional level middleware clusters includes sending a message from the external components to the connectivity component and the connectivity component redirecting the message to the internal services.
9. The method of any one of claim 7 or 8, wherein each of the big theme is at least one business topic.

10. The method of any one of claims 7 to 9, wherein each of the specific theme is a special business function.
11. The method of claim 7, wherein the creator in the business or functional modules creates one specific theme.
12. The method of claim 7, wherein each of the specific theme has at least one channel for communication.
13. The method of claim 12, wherein the channel for communication is defined in a configuration file.
14. The method of claim 13, wherein each specific theme and the big theme are defined in the configuration file.
15. A computer readable medium storing a program comprising instructions that, when executed by a system, cause the system to perform the method for deploying message-oriented middleware under one or more microservices, comprising:
 - splitting, each microservice into one or more of: a business module, a functional module or a combination thereof;
 - implementing, a functional level middleware cluster, including:
 - creating, three middleware clusters or virtual hosts for at least one of an internal service, a connectivity component and an external component;
 - creating, at least one user on each cluster or virtual host; and
 - sending, a message from the internal services to the connectivity component and the connectivity component redirects the messages to the specific external components;
 - implementing middleware server clusters under business level includes following steps:
 - creating at least one specific theme formed under a big theme for messaging of one or more microservices using one or more creator; and

allowing one or more consumers to subscribe in one or more specific themes.

16. The computer readable medium of claim 15, wherein implementing functional level middleware clusters includes sending messages from the external components to the connectivity component and the connectivity component redirecting the messages to the internal services.
17. The computer readable medium of any one of claim 15 or 16, wherein each of the big theme is at least one business topic and each of the specific theme is a special business function.
18. The computer readable medium of claim 15, wherein the creator in the business or functional modules creates one specific theme.
19. The computer readable medium of claim 15, wherein each of the specific theme has at least one channel for communication.
20. The computer readable medium of claim 19, wherein the channel for communication, each specific theme and the big theme are defined in the configuration file.